Solving Vehicle Routing Problems Using Multiple Ant Colonies and Deterministic Approaches

Samer Sa'adah

A thesis submitted in partial fulfilment of the
requirements of Napier University
for the degree of Doctor of Philosophy

October 27, 2007

Abstract.

In the vehicle routing problem with time windows VRPTW, there are two main objectives. The primary objective is to reduce the number of vehicles, the secondary one is to minimise the total distance travelled by all vehicles. This thesis describes some experiments with multiple ant colony and deterministic approaches. For that, it starts explaining how a double ant colony system called DACS 01 with two colonies has the advantage of using the pheromone trails and the XCHNG local search and the ability of tackling multiple objectives problems like VRPTW. Also, it shows how such DACS system lacks vital components that make the performance as comparable and competitive with that of the well-known VRPTW algorithms. Therefore, the inclusions of components, like a triple move local search, a push-forward and push-backward strategy PFPBS, a hybrid local search HLS and a variant of a 2-Opt move, improve the results very significantly, compared to not using them. Furthermore, it draws the attention to an interesting discovery, which suggests that if a DACS system uses ants that are more deterministic, then that system has the ability to bring performance that is better than that of another DACS system with pheromone ants. Consequently, the interesting discovery has motivated the author to investigate a number of SI1-Like deterministic approaches, which most of them depend on capturing under-constrained tours and removing them using a removing heuristic that uses the hybrid local search HLS. Some of these SI1-Like approaches show signs of the ability of improving the average, best and worst case performances of DACS systems on some problem set cases, if they are merged with such systems. Of course, this casts some doubt on whether the use of pheromone trails, which is a distinctive feature of multiple ant-colony systems in the research literature, is really so advantageous as is sometimes claimed. Experiments are conducted on the 176 problem instances with 100, 200 and 400 customers of Solomon [1], Ghering and Homberger [2] and [3]. The results shown in this thesis are comparable and competitive to those obtained by other state-of-the-art approaches.

Acknowledgments.

Thanks to my family and supervisors for the encouragement and support that I have received throughout my research.

Declaration.

I herby declare that I have composed this thesis entirely myself, and that it describes my own research unless indicated.

Samer Sa'adah.

Edinburgh.

October 27, 2007.

Contents

Chapter 1

Introduction

The vehicle routing problem with time windows VRPTW is an NP-hard problem in which several vehicles are to be used to deliver quantities of goods to a number of customers. In vehicle routing problems, some companies in the real world pursue the objective of reducing the number of vehicles that are used to pick up items or to deliver products while other companies seek to minimise the travelled distances could be done by the vehicles.

Of course, the first example, where the objective is to reduce the number of vehicles, is an indication that the companies are interested in saving the labour, in this case the vehicles, in order to use such labour in other activities that would return more profits to the companies involved. However when the objective is to minimise the travelled distances as in the second example mentioned above, then the companies involved are more interested in saving money in the form of fuel costs and driver wages. Also, making the objective of minimising the travelled distances, as a primary goal, rather than the objective of reducing the number of vehicles may lead into having extra used vehicles with low fuel costs and driver wages that service the customers.

In operations research OR, researchers in vehicle routing problems try to do both objectives but the major goal is to bring the number of vehicles down and then to look afterwards for minimising the travelled distances. For that, the general theme in this thesis in all the algorithms or approaches used is to consider the reduction of the number of vehicles as a primary goal and then the minimisation of the travelled distances comes as a secondary goal.

This thesis starts reviewing the literature of the VRPTW problem as in Chap-

ter 2. Then, it begins this research in Chapter 3 by describing critically how very-well known techniques such as Ant Colony Optimisation ACO and local search and their results are presented in the VRPTW literature and how they should be introduced. In Chapter 4, it studies and investigates different kinds of multiple ant colony systems after discovering that the success and the ability of a double ant colony system, called DACS 01, rely on the usage of the pheromone trails and the XCHNG local search and its handling to multiple-objective problems like VRPTW. For that, the thesis tries, here, to know which ingredients are behind that success. Are they the pheromone trails, the local searches or both of them? Also, it explores more things about the scalability of such systems in solving problem groups with customers more than 100 - like 200 and 400 customers. Later, it discovers that if a double ant colony system lacks a number of significant components like a triple move local search, a push-forward and push-backward strategy PFPBS, a hybrid local search HLS and a variant of a 2-Opt move, then that thing is going to make the performance a lot worse and result in finding very poor quality solutions.

At a later stage, the thesis finds out that a DACS system that uses more deterministic ants and does not consider the pheromone trails in the computation could bring very good quality performance and solutions compared with the performance and the solutions computed by another DACS system that depends mainly on the pheromone ants. For that in Chapter 5, it looks into a number of deterministic approaches that are named 'SI1-Like' and derived originally from one of Solomon's insertion heuristics [1]. Most of these SI1-like deterministic approaches try to remove under-constrained tours in order to see if such SI1-Like approaches at some stage can be merged with DACS systems that use the pheromone ants in a way that results in improving the performance of such DACS systems. Afterwards on some problem set cases, this thesis recognises that some of these SI1-Like approaches show signs of the ability of improving the average, best and worst case performances of DACS systems that are using the pheromone ants if they are hybridised together.

In this introductory chapter, Section 1.1 starts talking about routing problems in general and how they are related together. Then in Section 1.2, the VRPTW problem is defined and the motivations from this research are described. Later, the contributions of this thesis are summarized in Section 1.3 and the thesis itself in Section 1.4 is outlined.

1.1 Routing Problems

Routing problems are studied massively in the field of operations research OR and are considered as very hard problems to tackle. Like many other types of real world problems, routing problems vary from those that involve single objectives to those that have two or more conflicting objectives that make them even harder. Also, the more constraints and limitations are provided for such problems the more difficult and challenging they become.

The most simplistic form of a single-objective routing problem is the travelling salesman problem TSP. In this problem, a travelling salesman tries to find the shortest path in terms of distance to visit a number of cities and each city must be visited just once. In the literature, the TSP problem is where the distance between any two cities, like c_i and c_j, is the same in two directions, c_i to c_j and c_j to c_i. For that in the literature, TSP is called the symmetric TSP as well. In the case where the distance between any two cities in its two directions is not the same, the problem is called ATSP, asymmetric travelling salesman problem.

Now, if there are a number of m salesmen to visit a number of cities, the routing problem becomes m-TSP. Of course in the case of m-TSP, an extra care has to be made into how to distribute the m salesmen between the cities in order to serve them in the shortest possible way. Now if a city or a customer is attached with a demand then the problem will be a vehicle routing problem VRP with a limitation on the capacity of each of the vehicles available to solve such a problem. The VRP problem is a perfect example of the problem that has two conflicting objectives.

When the following two limitations are added to the VRP problem, then in this case VRP is considered as the vehicle routing problem with time windows, which is going to be defined in Section 1.2.

L1- The customers should be served within the time limits of their ready times and due dates.

L2- The vehicles should start servicing in a way that depends on the ready time of a depot and such vehicles should return on time to the depot according to the depot's due date.

1.2 VRPTW and Motivations

The vehicle routing problem with time windows VRPTW is the problem where a collection of customers must be visited by a number of vehicles such that a number of constraints are observed. The time-windows of customers in vehicle routing problems are of genuine practical interest and not just an academic complication because customers, like groceries and supermarkets for example, often arrange to hire staff for just two hours in order to help unload a large vehicle with items. So, the vehicle must arrive near the start of the time window of a customer or else the customer will have to pay those extra staff more money.

In VRPTW, each customer has a demand of goods and a time window in which to be serviced by one of the vehicles available at the depot. The vehicles have a limited capacity, and cannot be loaded beyond this capacity. Furthermore, there are time limits concerning the window in which a customer can be serviced. The earliest time the servicing may start is known as the "ready time"; the latest time is known as the "due date". A vehicle can arrive at a customer before that customer's ready time but in this case the vehicle has to wait before starting to serve the customer. The vehicle might arrive at any time within the time limits of the time window of a customer but in this case the customer has to wait for some time before being serviced. Also, a vehicle should return back to the depot on time without violating the due date limit of the depot.

The problem is to minimise the number of vehicles used to service all the customers, and also to minimise the total distance travelled by those vehicles. There are many aims out of this research, which are summarized as follows:

A1- First, to have the results and the performance of the multiple ant colony and deterministic approaches used in a way to be comparable, similar as or competitive with the results and performance published of the MACS-VRPTW and RVNS systems in [4] and [5] respectively on a classical set of 56 Solomon benchmark problem instances that has 100 customer nodes each.

A2- Second, to study and investigate about research ideas of our own or mentioned in existing methods and heuristics (like ACS+NN [6], ACS+I1 [6], RVNS [5], ES [2], ES4 [2], ES4C [2], TLOL [7], VGA1 [7], VGA2 [7] and many others) and whether using them would let the DACS system scale up in solving larger

problem instances with a number of customers greater than 100 like problem instances with 200 and 400 customers.

A3- Third, to discover and explore the effects of using different types of route construction and improvement heuristics in the decision making of the multiple ant colony and deterministic algorithms when searching for very good solutions.

A4- Finally, to improve on the multiple ant colony and deterministic approaches by getting better results, being faster in terms of elapsed CPU time or being more reliable.

1.3 Summary of Contributions

The work reported in this thesis demonstrates a number of points, summarised briefly here:

P1- Any metaheuristic algorithm that uses only static heuristics in building the solutions and tackles a multiple-objective problem like VRPTW as a single-objective problem and does not differentiate between attractive and unattractive edges and lacks a local search component does not lead necessarily into having a good outcome.

P2- A multiple ant colony system, like DACS 01 in Section 4.5, that is using and updating the pheromone trails to differentiate between attractive and unattractive edges and tackles VRPTW as a multiple objective problem and uses the local search is able to get a good outcome but it does not mean that the system is able to get the competitive or even the comparable outcome that is expected.

P3- The local search ingredients of a multiple ant colony system, like DACS 02 in Section 4.6, rather than the way of using and updating the pheromone trails, seem to be very important parts to get right in order to get a better outcome but it does not mean that such ingredients are enough in making the system output the comparable or the competitive outcome wanted.

P4- Evolutionary algorithms, multiple ant colony systems and local searches without a push forward and push backward strategy PFPBS as in Section 4.7.1 that stores, uses and updates information necessary to the nodes visited in solutions mean that the metaheuristic systems and the local searches involved are not going to have the ability of getting out the comparable outcome wished for during the amount of CPU time allocated.

P5- Modifying the local search that uses the push-forward and push backward strategy PFPBS with new ingredients, as in the hybrid local search HLS in Section 4.7.3 or as the variant of 2-Opt in Section 4.7.4, rather than the way of using and updating the pheromone trails leads into making multiple ant colony systems produce the competitive performance in the CPU time amount defined.

P6- A multiple ant colony system that uses ants, which are more deterministic as in Section 4.7.8, rather than the pheromone ants are able to get good results during the CPU time amount assigned. Therefore, pheromone trails are not always necessary for having good outcomes.

P7- A deterministic approach that captures and removes under-constrained tours as in Chapter 5 could have the ability of improving on some problem set cases the average, best and worst case performances of DACS systems that use the pheromone ants.

P8- An experimental work using 64 multiple ant colony systems and 20 deterministic approaches on 176 problem instances with 100, 200 and 400 customers.

1.4 Outline of the thesis

The rest of the thesis is divided into five chapters and they are as follows:

Chapter 2: Literature Review. This chapter provides a review to the literature of VRPTW that is related to the research in this thesis.

Chapter 3: Presentations of techniques and results. This chapter describes critically how Ant Colony Optimisation ACO techniques and

local searches and their results are presented in the VRPTW literature and how they should be introduced.

Chapter 4: Multiple Ant Colonies. This chapter investigates many different variants of multiple ant colony approaches.

Chapter 5: Deterministic Approaches. This chapter studies different types of deterministic approaches. .

Chapter 6: Conclusions and Future Research. This chapter ends up talking about the conclusions and future research. .

Chapter 2

Literature Review

Chapter 2 talks about the problem formulation of vehicle routing problems with time windows VRPTW, the problem groups with their instances and the complexity issues related to VRPTW. Also, this chapter explains in detail about the kind of exact and approximate approaches used in the literature in trying to solve such problems. Exact approaches such as dynamic programming, Lagrange relaxation-based and column generation methods are the classical heuristics used in trying to find optimality or optimal solutions for the Solomon problem instances [1] [8] of VRPTW. While the classical heuristics have managed to find optimal solutions for a number of problem instances, however they have failed to solve many other kinds of problem instances to optimality. For that, the researchers in the field of vehicle routing have found that approximate methods such as route construction heuristics, local searches and many other meta-heuristics are much so better in trying to solve such problem instances than the classical methods.

2.1 Problem formulation

Tables 2.1 and 2.2 represent the meanings of the constant and variable terms related to VRPTW. The VRPTW problem is formulated of a set of customers C that includes also the depot c_0, a set of arcs A and a set of vehicles V in a directed graph G. The set of arcs represents connections between the depot and the customers and among the customers. Each arc between two customers c_i and c_j, where $c_i \neq c_j$, has a value equal to the travel time $t_{ci,cj}$, which has the same value of the distance $d_{ci,cj}$ between the two customers because, by convention, vehicle speed is 1.

Table 2.1: Constants

Constants	meaning
N	number of depot and customer nodes.
$MaxVs$	maximum number of vehicles allowed to solve a problem instance.
M	maximum capacity of vehicles.
C	set of customers.
A	set of arcs.
c_i	customer c_i, $0 \leq i \leq N$. Note that c_0 is the depot.
$d_{c_i c_j}$	distance from customer c_i to customer c_j.
$t_{c_i c_j}$	travel time from customer c_i to customer c_j. Note that $(t_{c_i c_j} = d_{c_i c_j})$ such that each step of $t_{c_i c_j}$ = one unit of distance.
q_{c_i}	demand of customer c_i.
s_{c_i}	service time of customer c_i.
r_{c_i}	ready time of customer c_i. Note that $r_{c_0} = S$ = Start time of depot.
d_{c_i}	due date of customer c_i. Note that $d_{c_0} = F$ = Finish time of depot.

Table 2.2: Variables

Variables	meaning
V	set of vehicles.
$NV(S_i)$	number of vehicles in a solution S_i $(= vsize)$. Note that each vehicle has a route R_k.
$TD(S_i)$	total of travelled distances done by vehicles in a solution S_i.
$TT(S_i)$	total of time consumed by vehicles in a solution S_i.
$TL(S_i)$	total of loads done by vehicles in a solution S_i.
$TVWs(S_i)$	total of vehicle waiting times in a solution S_i.
$TCWs(S_i)$	total of customer waiting times in a solution S_i.
$NN(R_k)$	number of nodes in a route R_k $(= csize)$.
c_{ik}	i^{th} customer in a route R_k.
R_k	route $R_k = (c_0, c_1, ..., c_{csize-1})$.
$TD(R_k)$	travelled distance of route R_k.
$TT(R_k)$	total time consumed of route R_k.
$TL(R_k)$	total load of route R_k.
$TVWs(R_k)$	total of vehicle waiting times in route R_k.
$TCWs(R_k)$	total of customer waiting times in route R_k.
a_{c_i}	arrival time at customer c_i.
vw_{c_i}	vehicle waiting time before servicing c_i.
$tvws_{c_i}$	total of vehicle waiting times before servicing c_i.
cw_{c_i}	customer waiting time before c_i being serviced.
$tcws_{c_i}$	total of customer waiting times before c_i being serviced.
tt_{c_i}	total time consumed so far after servicing c_i.
ttd_{c_i}	total of travelled distances when arriving at c_i.
ad_{c_i}	accumulated demands of customers after servicing c_i.

The customers must be serviced by a number of vehicles such that a number of constraints are observed. Each customer has a demand of q_{c_i} and a time window $[r_{c_i}, d_{c_i}]$ and must be serviced within the boundaries of that time window. The service of a customer would take a number of time units equal to s_{c_i}. Now, each vehicle has a route and a maximum capacity equal to M. If a vehicle arrives at a time before the ready time r_{c_i} of a customer, this issue will cause the vehicle to wait a number of time units equal to vw_{c_i}. On the other hand, if a vehicle arrives at a customer after the ready time r_{c_i} and before the due date d_{c_i}, then it means that the customer has waited for a number of time units equal to cw_{c_i}. For a given route $R_k = (c_0, c_1, ..., c_{csize-1})$, c_0 = depot, there are a number of decision variables that are calculated as mentioned below in equations 2.1 to 2.13.

Equation 2.1 gives the arrival time at a customer c_i. The equations 2.2 and 2.3 give a vehicle waiting time and a customer waiting time at a customer c_i. In the equations 2.2 and 2.3, if there is a vehicle waiting time, then the customer waiting time is going to be zero. On the other hand, if there is a customer waiting time, then the vehicle waiting time will be nil.

$$a_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ tt_{c_{i-1}} + t_{c_{i-1}c_i} & \text{if } i > 0 \end{cases} \tag{2.1}$$

$$vw_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ \max(0,\, r_{c_i} - a_{c_i}) & \text{if } i > 0 \end{cases} \tag{2.2}$$

$$cw_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ |\min(0,\, r_{c_i} - a_{c_i})| & \text{if } i > 0 \end{cases} \tag{2.3}$$

Equations 2.4 and 2.5 give the total travelled distance and total time consumed from the beginning of the route till the customer c_i in that route. Also, the equations 2.6, 2.7 and 2.8 calculate the total of demands delivered, total of vehicle waiting times and total of customer waiting times from the beginning of the route till the customer c_i in that route.

$$ttd_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ ttd_{c_{i-1}} + d_{c_{i-1}c_i} & \text{if } i > 0 \end{cases} \tag{2.4}$$

$$tt_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ a_{c_i} + vw_{c_i} + s_{c_i} & \text{if } i > 0 \end{cases} \tag{2.5}$$

$$ad_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ ad_{c_{i-1}} + q_{c_i} & \text{if } i > 0 \end{cases} \tag{2.6}$$

$$tvws_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ tvws_{c_{i-1}} + vw_{c_i} & \text{if } i > 0 \end{cases} \tag{2.7}$$

$$tcws_{c_i} = \begin{cases} 0 & \text{if } i = 0 \\ tcws_{c_{i-1}} + cw_{c_i} & \text{if } i > 0 \end{cases} \tag{2.8}$$

Equations 2.9, 2.10, 2.11, 2.12 and 2.13 calculate respectively the total of travelled distances, total of consumed time, total of demands delivered, total of vehicle waiting times and total of customer waiting times of a route R_k.

$$TD(R_k) = ttd_{c_{csize-1}} + d_{c_{csize-1}c_0} \tag{2.9}$$

$$TT(R_k) = tt_{c_{csize-1}} + t_{c_{csize-1}c_0} \tag{2.10}$$

$$TL(R_k) = ad_{c_{csize-1}} \tag{2.11}$$

$$TVWs(R_k) = tvws_{c_{csize-1}} \tag{2.12}$$

$$TCWs(R_k) = tcws_{c_{csize-1}} \tag{2.13}$$

The objective in solving VRPTW is to find a solution with a minimal number of vehicles as a primary goal and a minimal total of travelled distances as a secondary one. Of course, this objective is subjected to the following constraints mentioned in equations 2.14 to 2.19.

$$R_0 \cup R_1 \cup ... \cup R_{vsize-1} = C \tag{2.14}$$

$$R_k \cap R_l = c_0,\ 0 \leq k, l \leq (vsize - 1),\ k \neq l \tag{2.15}$$

$$c_{ik} \neq c_{jk}, \qquad 1 \leq i, j \leq csize,\ 0 \leq k \leq (vsize - 1),\ i \neq j \tag{2.16}$$

$$a_{c_i} \leq d_{c_i},\ 1 \leq i \leq N \tag{2.17}$$

$$TT(R_k) \leq d_{c_0},\ 0 \leq k \leq (vsize - 1) \tag{2.18}$$

$$TL(R_k) \leq M,\ 0 \leq k \leq (vsize - 1) \tag{2.19}$$

The constraint in 2.14 ensures that all customers must be visited necessarily. The constraint in 2.15 means that all the visited customers are split amongst the tours and the depot must be visited in each one of the tours. In 2.16, the constraint make sure that each customer must be visited once. The constraint in 2.17 takes

care of that the arrival time of a vehicle at a customer c_i must be before the due date d_{c_i}. On the other hand in 2.18, the vehicle must arrive to the depot before its due date is finished. Finally, the vehicle must not be overloaded with items to be delivered according to the constraint in 2.19.

2.2 Problem groups and their instances

In this thesis, the algorithms are tested using a total number of 176 problem instances. The 176 problem instances used are divided into three major problem groups - PG100, PG200 and PG400. The problem group PG100 has 56 benchmark problem instances with 100 customers each and they were first used by Solomon in [1]. The problem groups PG200 and PG400 have 120 benchmark problem instances and they are for Gehring and Homberger and mentioned in [2] and [3]. The problem instances of PG200 have 200 customers each whereas the problem instances of PG400 have 400 customers each. PG200 and PG400 are amongst five problem groups (PG200, PG400, PG600, PG800 and PG1000) in which each problem group has 60 problem instances. The problem instances of these five problem groups are designed in a way to have customers greater than 100 customers like 200, 400, 600, 800 or 1000 and to extend the 56-benchmark problem instances of Solomon [1] in a similar fashion.

Now, each of the three problem groups of PG100, PG200 and PG400 tested in this thesis has six problem sets, which are R1, R2, C1, C2, RC1 and RC2. In each problem set of PG100, the number of problem instances varies between 8 and 12. More precisely, the problem set R1 has 12 problem instances, the problem set R2 has 11 problem instances, the problem set C1 has 9 problem instances and the problem sets of C2, RC1 and RC2 have 8 problem instances each. However in each problem set of PG200 and PG400, there are 10 problem instances. The problem sets R1 and R2 in each problem group have customers who are randomly distributed. Also in each problem group, customers who are clustered in groups are found in the problem sets C1 and C2. On the other hand, customers who have a mixture of the two problem features of randomly distributed customers and clustered customers in groups can be found in the problem sets RC1 and RC2 in each problem group.

As in Tables 2.3, 2.5 and 2.7, each problem instance of the problem groups

PG100, PG200 and PG400 has two statistical values - an average AVG and a standard deviation SD. The AVG and SD values of a problem instance are related to the time windows of the customers of that problem instance. The AVG value of a problem instance refers to the average of the widths of the customers' time windows in that problem instance whereas the SD value refers to the standard deviation. Note that the identification number of each problem instance, in a problem set of a problem group, is mentioned underneath the column PNo.

It can be seen from the statistics in Tables 2.3, 2.5 and 2.7 that the six problem sets of each problem group are divided into two major types indicated by the numbers 1 and 2 that are attached with the names of the six problem sets. Now in the problem sets of type 1 like R1, C1 and RC1, the problem instances have customers who have tight time windows. However in the problem sets of type 2 such as R2, C2 and RC2, the problem instances have customers who have wide time windows. Furthermore, the vehicles in each problem instance of the problem sets of type 1 have small capacities for picking up or delivering items whereas the vehicles in each problem instance of the problem sets of type 2 have large capacities.

For instance in the problem instances R101 and R201 of the problem group PG100, the tightness and the wideness issue of the widths of the time windows is indicated in Table 2.3 by comparing the AVG values 10.00 and 115.96 together, as located in the two table cells intersecting the problem sets R1 and R2 with the problem instance number 01. For that, the problem instance R101 of PG100 has customers who have tight time windows whereas the problem instance R201 of PG100 has customers who have wide time windows.

In a matter of fact, the tightness and the wideness issue of the widths of the time windows can be recognised, in all the three problem groups PG100, PG200 and PG400, between any problem instance in the problem sets R1, C1 and RC1 of type 1 and its corresponding problem instance in the problem sets R2, C2 and RC2 of type 2. For example, the tightness and the wideness issue of the widths of the time windows can be recognized in each problem group between the problem instances C101 and C201 and the problem instances RC101 and RC201 and so on. Furthermore, the time windows of all the customers in some problem instances like the problem instance R101 of each of the problem groups PG100, PG200 and PG400 have the same width and this is indicated by the standard deviation value, SD = 0.

Additionally as in Tables 2.4, 2.6 and 2.8, the statistics related to the average and standard deviation values, AVGs and SDs, indicate that the customers' demand quantities and distance locations from the depot are the same in all the problem instances of each problem set in each of the three problem groups PG100, PG200 and PG400. Also, it is recognised that in all the problem instances of the problem sets R1 and R2 or RC1 and RC2 of each problem group, the customer demand quantities and the customer distance locations from the depot are the same. On the other hand when it comes to the clustered problem sets C1 and C2 of each problem group, the customer demands and the customer distance locations from the depot in the problem instances of C1 are configured differently from those of the problem instances of C2. Note that what is said in the previous sentence is true with the exception of the customer demands in all the problem instances of C1 and C2 of PG100. In other words, the customer demands in all the problem instances of C1 and C2 of PG100 are configured similarly.

Table 2.3: Statistics related to the time windows of the 100 customers in each of the 56 problem instances in the problem group PG100.

	R1		R2		C1		C2		RC1		RC2	
PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
01	10.00	00.00	115.96	35.78	60.76	10.53	160.00	00.00	30.00	00.00	120.00	00.00
02	57.39	82.54	328.81	373.08	325.69	460.40	937.74	1353.88	71.46	72.48	318.96	346.41
03	102.99	93.63	541.66	427.18	588.49	531.36	1714.82	1562.67	112.50	83.43	517.50	399.61
04	148.31	80.67	751.26	371.46	852.94	459.88	2492.58	1353.54	154.60	73.23	717.10	346.67
05	30.00	00.00	240.00	00.00	121.61	20.98	320.00	00.00	54.33	41.81	223.06	162.66
06	72.39	73.85	422.39	317.51	156.15	91.86	486.64	83.99	60.00	00.00	240.00	00.00
07	112.99	83.60	602.99	364.86	180.00	00.00	612.32	302.72	88.21	32.82	349.50	163.84
08	153.31	72.01	783.31	315.37	243.28	41.96	640.00	00.00	112.33	30.80	471.93	71.67
09	58.89	08.93	349.50	163.84	360.00	00.00	-	-	-	-	-	-
10	86.50	39.27	383.27	237.98	-	-	-	-	-	-	-	-
11	93.10	54.73	471.94	71.67	-	-	-	-	-	-	-	-
12	117.64	17.45	-	-	-	-	-	-	-	-	-	-

Table 2.4: Statistics related to the various distances from the depot and the demands of the 100 customers in each of the 56 problem instances in the problem group PG100.

		R1		R2		C1		C2		RC1		RC2	
	PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
Distance from depot	All	24.95	09.53	24.95	09.53	28.85	12.28	29.71	11.58	33.09	12.80	33.09	12.80
Demand	All	14.58	08.87	14.58	08.87	18.10	10.42	18.10	10.42	17.24	09.42	17.24	09.42

Table 2.5: Statistics related to the time windows of the 200 customers in each of the 60 problem instances in the problem group PG200.

	R1		R2		C1		C2		RC1		RC2	
PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
01	10.00	00.00	121.24	40.10	60.36	10.45	160.00	00.00	30.00	00.00	120.00	00.00
02	150.16	243.54	708.66	1023.99	347.26	497.65	984.96	1432.48	165.05	234.67	708.42	1021.77
03	289.98	280.99	1296.26	1181.65	633.14	575.44	1810.64	1654.84	300.72	271.73	1296.26	1179.29
04	431.01	244.23	1883.08	1023.03	919.14	498.12	2634.32	1432.25	436.06	235.67	1884.62	1021.51
05	30.00	00.00	240.00	00.00	119.26	20.16	320.00	00.00	64.70	28.72	279.08	160.59
06	165.16	234.86	799.19	971.03	157.83	86.76	506.32	90.11	60.00	00.00	240.00	00.00
07	299.98	270.97	1357.28	1120.18	180.00	00.00	630.50	288.64	91.42	31.93	369.81	166.21
08	436.01	235.57	1914.05	969.10	239.34	37.00	640.00	00.00	119.22	32.85	485.44	182.57
09	60.12	18.35	373.10	172.27	360.00	00.00	851.64	372.48	120.00	00.00	480.00	00.00
10	124.50	27.22	476.46	57.96	479.68	86.51	880.00	00.00	150.00	00.00	600.00	00.00

Table 2.6: Statistics related to the various distances from the depot and the demands of the 200 customers in each of the 60 problem instances in the problem group PG200.

		R1		R2		C1		C2		RC1		RC2	
	PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
Distance from depot	All	53.49	19.96	53.49	19.96	55.80	21.75	49.19	21.17	52.98	19.77	52.98	19.77
Demand	All	17.56	08.88	17.56	08.88	17.65	07.63	18.85	08.58	17.79	08.44	17.79	08.44

Table 2.7: Statistics related to the time windows of the 400 customers in each of the 60 problem instances in the problem group PG400.

	R1		R2		C1		C2		RC1		RC2	
PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
01	10.00	00.00	118.78	40.59	59.81	10.02	160.00	00.00	30.00	00.00	120.00	00.00
02	188.14	309.25	870.59	1306.23	378.28	552.53	1005.20	1465.81	191.73	280.76	834.24	1238.72
03	365.41	356.37	1622.05	1510.09	696.30	637.79	1848.58	1690.80	354.68	325.58	1547.66	1429.56
04	543.60	309.35	2376.54	1307.52	1015.45	551.89	2693.56	1464.74	516.48	282.07	2260.80	1237.72
05	30.00	00.00	240.00	00.00	119.64	19.59	320.00	00.00	63.63	27.92	273.00	161.67
06	203.14	300.59	962.32	1252.73	159.49	79.18	497.80	93.59	60.00	00.00	240.00	00.00
07	375.41	346.37	1684.92	1446.87	180.00	00.00	624.12	272.84	89.50	29.46	367.10	170.41
08	548.60	300.70	2407.70	1253.32	240.75	38.98	640.00	00.00	119.22	30.38	480.78	176.04
09	61.45	18.60	367.51	173.70	360.00	00.00	806.16	363.98	120.00	00.00	480.00	00.00
10	119.56	31.15	476.86	60.80	479.38	90.19	880.00	00.00	150.00	00.00	600.00	00.00

Table 2.8: Statistics related to the various distances from the depot and the demands of the 400 customers in each of the 60 problem instances in the problem group PG400.

		R1		R2		C1		C2		RC1		RC2	
	PNo	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
Distance from depot	All	73.19	27.25	73.19	27.25	78.97	26.27	66.10	24.9	77.03	25.25	77.03	25.25
Demand	All	17.77	08.32	17.77	08.32	17.98	07.80	18.90	08.21	17.82	07.81	17.82	07.81

2.3 Complexity issues

In this section, the complexity issues related to the VRPTW problem will be discussed briefly. While discussing the complexity issues of VRPTW, the terms 'problems' and 'problem instances' mean two different things. Problems are like VRPTW, Travelling Salesman Problem TSP, Quadratic Assignment Problem QAP, Sequential Ordering Problem SOP...etc and each problem refers to a class of problem instances. For example, VRPTW is a class of problem instances in which each problem instance is about visiting all the customers using a minimal number of vehicles as a primary goal and a minimal total of travelled distances, as a secondary goal, done by the vehicles in a geographical area that has its own features.

Now in order to solve the VRPTW problem, an algorithm of a number of computational steps needs to be developed. The algorithm is a step-by-step procedure that has the responsibility of solving a particular problem. The efficiency of this algorithm could be measured in complexity theory using the time complexity function. In complexity theory, the efficiency of an algorithm depends on the size of a problem instance. The size of a problem instance might be equal to the number of items needed to be packed in a number of bins or, let us say, the number of customers that are in need to be serviced by a number of vehicles. As mentioned in [9], an algorithm is said to have polynomial time complexity, if the time complexity function is a polynomial one or can be bounded by a polynomial. If the time complexity function cannot be bounded by a polynomial, the algorithm is said to have exponential time complexity. An algorithm is said to have a pseudo-polynomial running time, if the algorithm deals with the largest problem instance. If there is a bound on the length of this largest problem instance, the algorithm will be considered as a polynomial algorithm.

Some problems can be solved in polynomial time, while other problems can only be solved in non-deterministic polynomial time. Therefore, if a problem can be solved in polynomial time using a deterministic Turing machine or algorithm, then it is said to belong to a set called P. However, if a problem can only be solved in non-deterministic polynomial time using a non-deterministic Turing machine or algorithm, then it is said to belong to a set called NP. A problem is said to be NP-complete, if any problem in NP can be transformed to it. Now, problems that

are considered as NP-complete are very hard problems. An example on such kinds of problems is the TSP problem. TSP is considered as an NP-complete problem because there are many hard problems that can be transformed to it such as VRPTW, QAP and SOP. Now is VRPTW hard? The VRPTW problem is proven in [9] to be an NP-hard problem in the strong sense and at least as hard as TSP. VRPTW contains other several NP-hard optimization problems in addition to TSP that makes it even harder such as Bin Packing and VRP.

2.4 Exact solution methods

In the early years into the research of VRPTW, a number of optimal algorithms emerged in order to tackle the VRPTW problem. Those optimal algorithms are used to try to solve a number of VRPTW problem instances to optimality. Such algorithms are able to solve a small set of such problem instances by finding optimal solutions to them. However in other kinds of problem instances, these optimal algorithms are not successful in finding optimal solutions for many reasons related to how hard these kinds of problem instances are and how effective such algorithms are in solving such problem instances.

Optimal algorithms are exact methods that have the responsibility to solve VRPTW exactly by finding an exact solution that is optimal. Examples of such algorithms are dynamic programming, Lagrange relaxation-based and column generation methods. Dynamic programming techniques as mentioned in [9] and [8] are applied once and only to solve problem instances that have up to 15 customers. Such techniques use what is called the Branch-and-Bound technique to achieve optimality.

As mentioned in [9] and [8], there are three different types of Lagrange relaxation-based methods. The first of these three methods is the shortest path with side constraints combined with Lagrange relaxation. In this approach, the constraint that ensures that every customer is served exactly once is relaxed or associated with what is so-called the Lagrange multiplier. The master problem consists of finding optimal Lagrange multipliers. The sub-problem for each vehicle is the shortest path problem with time window and capacity constraints. This approach manages to solve problem instances of 100 customers from the Solomon test cases among them some previously unsolved problem instances.

The second method has presented an algorithm for solving the VRPTW problem optimally where VRPTW is formulated as a K-tree problem with degree $2K$ on the depot. Note that K is equal to the number of vehicles *vsize* - See Section 2.1. Here, VRPTW is described as mentioned in [9] as finding a K-tree as minimum as possible with 2K edges on the depot and 2 edges on each customer. In this approach, all the constraints are Lagrangian relaxed except the constraints ensuring that at most one side of the arc is joining the customers c_i and c_j. In other words, either e_{ij}or e_{ji}. When solving a K-tree problem, 2K edges must occur on the depot. Behold that finding a K-tree is subjected to time and capacity constraints. If the depot has not reached the degree of 2K edges, then a series of arc or edge exchanges occurs until the 2K-degree occurs on the depot. By using this algorithm, it is possible to solve several clustered problem instances of Solomon to optimality. On the other hand, this approach is not able according to [9] to solve any of the random problem instances.

The third method is variable splitting, also called as Lagrange decomposition or cost splitting, combined with Lagrange relaxation for the VRPTW problem. In this approach, the researchers have concluded that VRPTW can be split into two sub-problems using one of two ways. The first way decomposes VRPTW into an Elementary Shortest Path Problem with Time Windows ESPPTW and a General Assignment Problem GAP. ESPPTW is subjected to network and time constraints, whereas GAP is subjected to capacity and visiting customer constraints. In this way, the researchers according to [9] is able to solve problem instances with up to 16 customers to optimality. In the second way, VRPTW is decomposed into an Elementary Shortest Path Problem with Time Windows and Capacity Constraints ESPPTWCC and a Semi Assignment Problem SAP. ESPPTWCC is subjected to network, time and capacity constraints, whereas SAP is subjected to visiting customer constraints. However in this way, problem instances with 100 customers from the Solomon test cases can be solved to optimality.

In [9], a column generation technique is used to solve VRPTW. The VRPTW problem is split into two problems. The first and master problem is a relaxed set partitioning problem while the second and sub problem is the shortest path with additional constraints - vehicles' capacity and time windows. In this approach, each column of a constraint matrix corresponds to a feasible route, while the rows

correspond to customers. A large number of columns may be generated but a significant number of the columns will not be having interesting features. Therefore, one of the features of this technique is to add new columns and to remove those that have no benefit for the solution process. Using this technique in [9], it is possible to solve problem instances to optimality as it did not happen before.

Generally speaking, the exact techniques described above have used mostly a special group of 87 problem instances, which is similar to the group of the 56-benchmark problem instances of Solomon considered in this thesis. The main difference between the two groups of problem instances according to [8] is the number of customers. In the group of 87 problem instances, the number of customers varies between 25 and 100. For more information about exact methods, the interested reader can read the following the references [9] [8] and [6]. Many researchers have used exact techniques to solve a small set of such problem instances to optimality, but interest quickly moved to heuristics.

2.5 Route building heuristics

Route building heuristics can be divided into two main branches. The first branch is called sequential heuristics whereas the second branch is called parallel heuristics. The sequential heuristics construct one route at a time until all customers are visited. Once a route is filled with visited customers and has no more capacity to have additional customers, another route is initiated to visit the remaining unvisited customers. On the other hand, the parallel methods can be described by the construction of a number of routes simultaneously.

Clarke and Wright in 1964 as mentioned in [10] were among the first researchers to use a route-building heuristic called Savings. The basic idea of the Savings heuristic is very simple. Suppose that there are a number of customers equal to n and there is a need to build a solution with a number of vehicles equal to n. Then, each vehicle, on its own, would serve one and only one customer with some sort of demand. Then, the total distance travelled of the solution would be $2 \times \sum_{i=1}^{n} d_{c_0 c_i}$. If two customers are served out of the n customer nodes on a single trip via a vehicle, then the saving value of this single trip is going to be the result of this mathematical

formula.

$$\begin{aligned} s_{c_i c_j} &= 2 \times d_{c_0 c_i} + 2 \times d_{c_0 c_j} - (d_{c_0 c_i} + d_{c_i c_j} + d_{c_0 c_j}) \\ &= d_{c_0 c_i} + d_{c_0 c_j} - d_{c_i c_j} \end{aligned} \tag{2.20}$$

The quantity value of $s_{c_i c_j}$ is considered as the saving value occurred because of serving the customers c_i and c_j in a single trip. Now, the bigger the quantity value of $s_{c_i c_j}$ the better to choose the arc or the edge that represents both customers in order to service them on a single trip. However, this saving value will not be regarded, if putting both customers causes constraint violations - such as the vehicle capacity is running out or the vehicle arrives late after the due date of a customer.

Also in [1], Solomon in 1987 was one of the first researchers to use another set of route-building heuristics such as Savings, Time Oriented NN, I1, I2, I3 and Time-Oriented Sweep. Of course, Solomon has used two versions of the Savings heuristic. The first version, Savings or SAV, cares about the saving value in terms of distance while the second version, Savings with waiting time limit or SWT, ignores any saving value that could lead into making a vehicle wait at customer c_j more than a parametric value W. Solomon shows the results of these two savings heuristics on just two problem sets, namely R1 and C1.

The second heuristic that is considered by Solomon in [1] is the Time-Oriented Nearest Neighbour heuristic. This heuristic starts every route by finding the closest unrouted customer in terms of a measure that takes into consideration three factors. The three factors are how close customer c_i from customer c_j in terms of distance, the time difference between the completion of service at customer c_i and the beginning of service at customer c_j and the urgency of delivery to customer c_j. Once the closest customer, in terms of the measure described previously, is routed, the search of the heuristic looks for another unrouted customer. If the search fails to find a customer with a feasible insertion, another route is initiated to try to visit the remaining unvisited customers.

Then, Solomon in [1] uses a class of three route building or insertion heuristics called I1, I2 and I3. Each one of those heuristics has the responsibility to insert customers into routes. Routes are built sequentially or on a route-by-route basis. Once the search of a heuristic is failed, another route is initialised. Now for the purposes of this thesis, the differences between the three heuristics I1, I2 and I3

are not really important. For that at first, a general description of how does each of the three heuristics work will be mentioned below. Later then, the I1 heuristic only is going to be explained in detail. A heuristic I1, I2 or I3 starts a route by seeding it with an unrouted customer using one of the seeding strategies mentioned in [1]. The seeding strategies of the earliest deadline and the farthest are the most and main strategies used in all the three heuristics - Note that I1 uses only the two seeding strategies mentioned previously. After seeding a route with a customer, a heuristic starts to insert the remaining unvisited customers via two functions until the route is filled and does not have the capacity to serve any more customers. One of the two functions is a minimization function while the other is a maximization one.

Now, for every unrouted customer, the general idea of the minimization function is to find an insertion place that is feasible and its cost is the minimum in a route. Once a minimum cost value is calculated for every unrouted customer, a maximization function is used to pick the best customer that has a maximum value of some sort of these unrouted customers in order to insert that selected customer at a later stage of the heuristic into that route. For example, let $(i_0, i_1, i_2, ..., i_m)$ be the current route with $i_0 = i_m =$ depot. For each unrouted customer c_u, the best feasible insertion place or minimum cost place between two customers i_{p-1} and i_p is computed in the current route via the equation in 2.21. Next, the best unrouted and feasible customer c_{u^*} to be inserted in the current route is the one that maximizes the equation in 2.22.

$$c_1(i_{p-1}, c_u, i_p) = \min\lfloor c_1(i_{p-1}, c_u, i_p)\rfloor, \text{ where } p = 1, ...,m \tag{2.21}$$

$$c_2(i_{p-1^*}, c_{u^*}, i_{p^*}) = \max_{c_u}\lceil c_2(i_{p-1}, c_u, i_p)\rceil, \text{ where } p = 1, ...,m \tag{2.22}$$

Now, the two functions in each of the three heuristics I1, I2 and I3 are coded differently. However experimentally, I1 is proven to be the best heuristic amongst all other route building heuristics mentioned in [1]. As an example on the two general functions in 2.21 and 2.22, the I1 heuristic calculates the best feasible insertion place with minimum cost in the current route for every unrouted customer through the equations in 2.23, 2.24 and 2.25 where $a_1 + a_2 = 1$ and $a_1 \geq 0$, $a_2 \geq 0$. The variables $d_{i_{p-1}c_u}$, $d_{c_u i_p}$ and $d_{i_{p-1}i_p}$ are distances between the customers i_{p-1}, c_u and i_p. On the other hand, the variable $b_{i_p c_u}$ expresses the beginning of service at customer i_p after

inserting the customer c_u whereas the variable b_{i_p} represents the beginning of service at customer i_p before inserting the customer c_u.

$$c_1(i_{p-1}, c_u, i_p) = a_1 c_{11}(i_{p-1}, c_u, i_p) + a_2 c_{12}(i_{p-1}, c_u, i_p) \quad (2.23)$$

$$c_{11}(i_{p-1}, c_u, i_p) = d_{i_{p-1}c_u} + d_{c_u i_p} - \mu d_{i_{p-1}i_p} \quad (2.24)$$

$$c_{12}(i_{p-1}, c_u, i_p) = b_{i_p c_u} - b_{i_p} \quad (2.25)$$

Furthermore, the I1 heuristic chooses the best unrouted customer with maximum value for insertion through the following equation 2.26, where the variable $d_{c_0 c_u}$ is the distance between the depot c_0 and the customer c_u and $\lambda \geq 0$.

$$c_2(i_{p-1}, c_u, i_p) = \lambda d_{c_0 c_u} - c_1(i_{p-1}, c_u, i_p) \quad (2.26)$$

A route building heuristic called Time-Oriented Sweep is presented in [1]. This heuristic has two stages. The first stage is called the clustering phase while the second is called the scheduling phase. In the clustering phase, customers are assigned to vehicles. In the scheduling phase, the customers of a vehicle are scheduled using the I1 route building heuristic.

In [11], a PARallel InSertion heuristic PARIS of Potvin and Rousseau is created to build routes in parallel. The PARIS heuristic is inspired by the I1 heuristic of Solomon [1]. The main difference between PARIS and I1 is that PARIS builds the routes in parallel rather than one by one. In PARIS, many different seed customers are selected to create the initial set of routes - i.e. each route serves a single customer. One of the main challenges has risen in PARIS is to know what is the minimum number of required routes to visit all customers of a problem instance. Therefore, an estimate of the minimum number of routes is decided in [11] by running the I1 heuristic once and later to use that number to build the parallel routes needed in PARIS. Of course, the seed customer of each route is chosen using the farthest seeding strategy.

Later, PARIS computes the minimum feasible insertion costs for each unrouted customer c_u in each route - note that I1 computes one and only one minimum feasible insertion cost because of its sequential nature. Then, PARIS selects the unrouted customer that maximizes what is called the generalized regret measure. Hence,

unrouted customers with large regret measures must be considered first. Here, a large regret measure means according to [11] that there is a large gap between the best insertion point of a given unrouted customer in a route R_1 and the best insertion points of the same unrouted customer in other routes. Of course, the steps mentioned in this paragraph are repeated within PARIS until all customers are visited. Also, once a solution with the least number of vehicles is selected out of three solutions, created by three different parameter settings of a_1, a_2 and μ, PARIS is initiated again with a number of vehicles equal to the least number of vehicles minus one. PARIS continues to reduce the number of vehicles until no more feasible solutions can be found with the three different parameter settings.

2.6 Local Searches

Local searches are some sort of approximate algorithms that are used to find near-optimal solutions in the neighbourhood of a solution S. These kinds of algorithms have come to the field of VRPTW in order to improve on the quality of solutions generated by route building heuristics in Section 2.5 and meta-heuristic algorithms in Section 2.7. The main purpose of a local search is to create neighbouring solutions of a solution via a move-generation method. A neighbouring solution is selected and accepted instead of the solution S according to some acceptance-criterion.

In order to design a local search algorithm, a number of components need to be specified carefully: What is the sort of solution S to start improving upon, what is the move-generation method needed to create neighbouring solutions of the current solution S, what is the acceptance-criterion used to accept a neighbouring solution instead of the solution S and what is the stopping criterion of the local search designed. All the previous components are very important ones in designing a local search. The sort of solution, to start local searching from, could be a feasible or an infeasible one. The move-generation method could generate a number of neighbouring solutions via move operators by changing one of the features or a combination of features of the solution S in one step. In this thesis, one step refers to trying a number of move operators between two or more customers. The features of a solution could be the customers or the edges that relate them. The move operators could be like swap, relocate, insert...etc.

According to [8] and [12], there are two main kinds of acceptance-criteria. The first kind is called the first-accept criterion while the second is called the best-accept criterion. In the first-accept criterion, the local search accepts the first neighbouring solution S^* of a number of neighbouring solutions to replace the solution S, if S^* is better than S. On the other hand in using the best-accept criterion, the local search examines all the neighbouring solutions generated via the move operators and accepts the best neighbouring solution S^* to replace the solution S - Note that S^* should be better than S in quality. The stopping criterion is the kind of component that stops the local search, if a number of iterations is done, a certain amount of CPU time is elapsed or no improvement is achieved during the previous iteration of the local search.

Local searches can be applied on a single route or two or more routes at a time. In the case it is applied on a single route, this local search will be referred to as an intra-route improvement local search. On the other hand, if the local search is used for two or more routes, it is called an inter-route improvement local search. Note that there are local searches considered as intra-route and inter-route improvement local searches at the same time. In order to let these local searches work and later to try to improve a solution S, they need to have one or more move operators. The purpose of a move operator is to make a neighbouring solution S^* of the solution S. In the literature, there are many move operators mentioned.

One of the most popular move operators used in TSP and VRPTW is 2-Opt. The 2-Opt move is about exchanging two old edges with two new edges. Therefore, a local search that has 2-Opt would continue in trying to improve the solution S until no more improvement can be gained. In VRPTW, 2-Opt is used for intra-route and inter-route improvements as in Figures 2.1 and 2.2 where the depot is indicated by the square shape. Figure 2.1 represents a 2-Opt move in a single tour, whereas Figure 2.2 represents a 2-Opt move between two tours. In an intra-route improvement of a 2-Opt as in Figure 2.1, the edges $e_{i,i+1}$ and $e_{j,j+1}$ of a single tour are replaced by the edges $e_{i,j}$ and $e_{i+1,j+1}$. Note that a part of the single tour in Figure 2.1 is reversed. Also in an inter-route improvement in Figure 2.2, the same kind of move would do the same thing but between two tours - Note that the 2-Opt move in Figure 2.2 keeps the order in which the customers are visited. Here, the edges $e_{i,i+1}$ and $e_{j,j+1}$ of two tours are replaced by the edges $e_{i,j+1}$ and $e_{j,i+1}$.

Another move operator is the relocate move. In this move, a customer is relocated to be between two customers. This move could happen in a single route as in Figure 2.3 or between two routes as in Figure 2.4. In a single route as in Figure 2.3, the customer at index i is relocated to be between the two customers at the indices j and $j+1$. In this relocate move, the edges $e_{i-1,i}$, $e_{i,i+1}$ and $e_{j,j+1}$ are replaced by the edges $e_{i-1,i+1}$, $e_{j,i}$ and $e_{i,j+1}$ - Note that the orientation of the route is preserved here. In Figure 2.4, the relocation of the customer at index i happens between two tours. For that, the edges are replaced but in two routes and therefore the edges $e_{i-1,i}$, $e_{i,i+1}$ and $e_{j,j+1}$ are replaced by the edges $e_{i-1,i+1}$, $e_{j,i}$ and $e_{i,j+1}$.

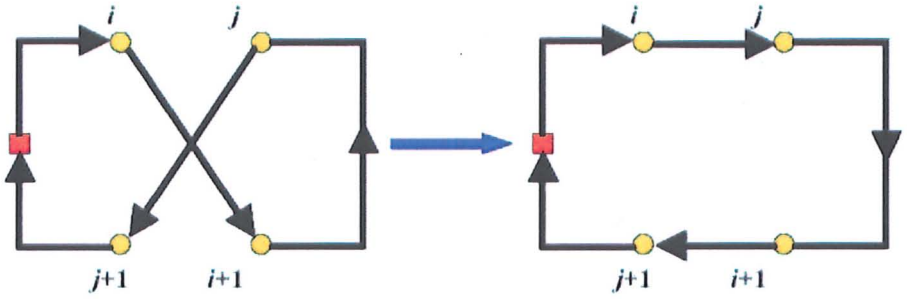


Figure 2.1: Two Opt for intra-route improvements

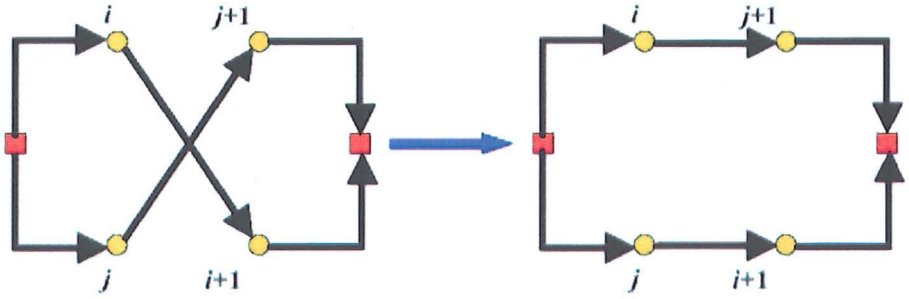


Figure 2.2: Two Opt star for inter-route improvements

The exchange operator is also used a lot in the literature. The purpose of the exchange operator is to exchange two customers on a single tour as in Figure 2.5 or between two tours as in Figure 2.6. For example, the exchange move in Figure 2.5 exchanges two customers on a single tour. Therefore, the edges $e_{i-1,i}$, $e_{i,i+1}$, $e_{j-1,j}$ and $e_{j,j+1}$ are replaced by the edges $e_{i-1,j}$, $e_{j,i+1}$, $e_{j-1,i}$ and $e_{i,j+1}$. However in Figure 2.6 the exchange of two customers between two different tours causes the

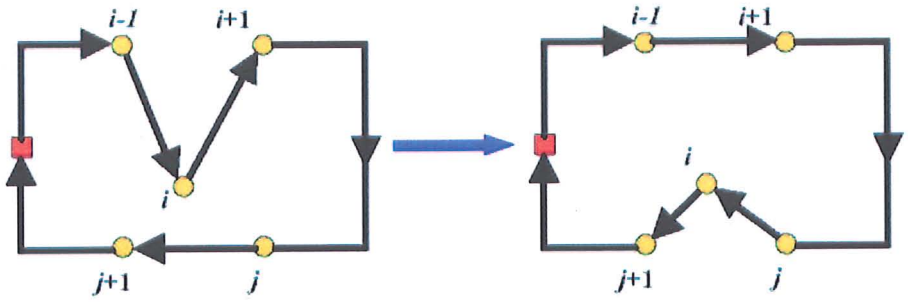


Figure 2.3: Relocate for intra-route improvements

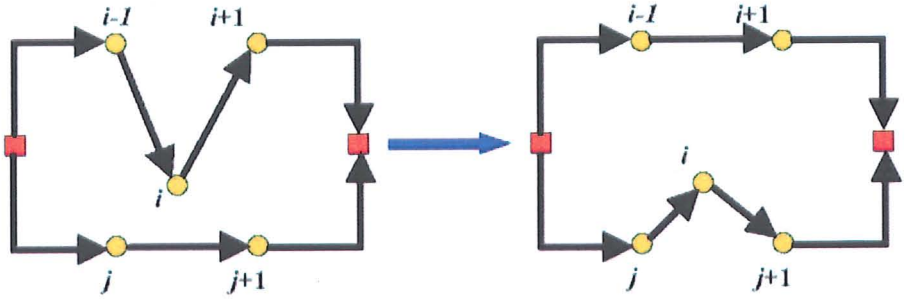


Figure 2.4: Relocate for inter-route improvements

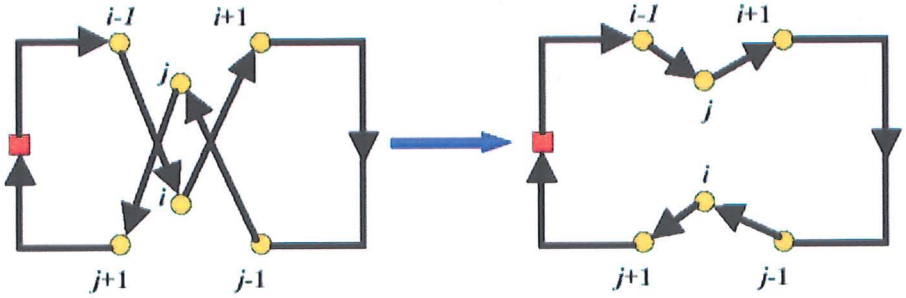


Figure 2.5: Exchange for intra-route improvements

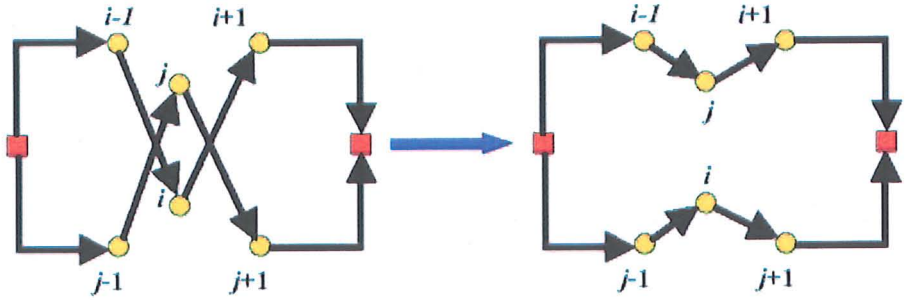


Figure 2.6: Exchange for inter-route improvements

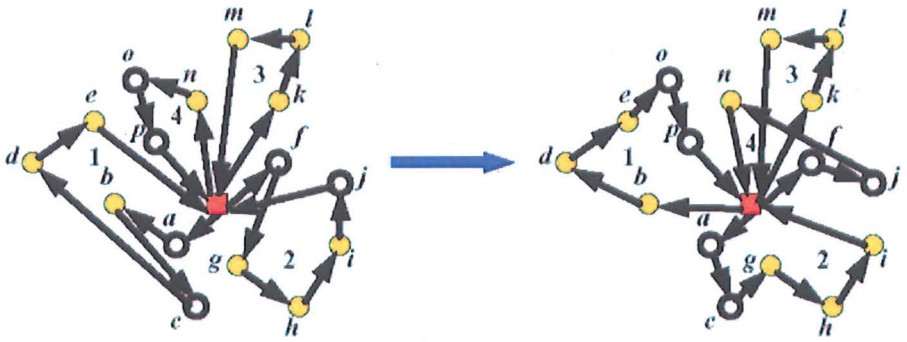


Figure 2.7: Cyclic Transfer

edges $e_{i-1,j}$, $e_{j,i+1}$, $e_{j-1,i}$ and $e_{i,j+1}$ to replace the edges $e_{i-1,i}$, $e_{i,i+1}$, $e_{j-1,j}$ and $e_{j,j+1}$. Another operator mentioned in the literature is the Cyclic Transfer operator [13]. The basic idea of the cyclic transfer is to transfer simultaneously the customers denoted by white circles in Figure 2.7 in cyclical manner between the tours. More precisely, here customers a and c in tour 1, j and f in tour 2 and o and p in tour 4 are simultaneously transferred into tours 2, 4, and 1 respectively and tour 3 remains untouched. In [9] and [8], there is a lot of mention to the Or-Opt move. The Or-Opt move is about moving a segment of length l of consecutive customers, say $l = 3$, from a place on a tour tr_1 to another place either on the same tour or on a different tour, tr_2 for example. Also, there is a lot of mention for the operator $\lambda-$interchange. Its basic idea is to exchange a subset of customers of size $\leq \lambda$ with a subset of customers of size $\leq \lambda$ from another tour.

In the literature, many VRPTW algorithms have used local searches or improvement heuristics that use the move operators mentioned above or some sort of operators similar to them. In [14], the researchers solve the VRPTW problem using a parallel tour construction heuristic. In this approach, the tours are constructed depending on the process of negotiation between customers and tours. In the process of negotiation, each unrouted customer c_u requests and receives from every tour in the schedule a price (i.e. minimum insertion place as described in Equation 2.23) for a potential service or insertion. Then, each unrouted customer who did receive a finite offer sends a proposal to that tour which offered him the lowest price or minimum insertion place between all the tours. Later, each tour accepts among all proposals received the customer c_u, whose insertion is the most efficient according

to Equation 2.26. The researchers in [14] describes this approach as the sequential application of an auctioning process or a market game. Furthermore, if a feasible solution with a number of vehicles equal to V for the problem data in hand is obtained, then the same approach will be initiated to find a solution with $V \Rightarrow V - 1$. Then, the researchers in [14] applies an improvement procedure to the feasible solution with the least number of vehicles found in an effort to improve its quality in terms of distance. This improvement procedure unschedules a number of what are so-called expensive customers in order to re-route them again through the process of negotiation described above.

The Guided Local Search GLS for VRPTW is used in [15]. GLS shares similarities as mentioned with tabu search in Section 2.7.2. The objective function of this approach is augmented with a set of penalty values - i.e. a penalty value for each arc of the graph. The penalty values are to encourage moving the search away from already visited local minima and exploring the other points in the search space. Strictly speaking, the search is penalised if it strays too close to previously visited local minima. In GLS, each solution is expressed as having a collection of features or arcs. In the objective function and when calculating the quality value of a solution, each feature is attached with a cost value and a penalty value expressing the number of times in which such feature is penalised. Therefore, the features of a solution that occur in a local minimum solution previously found will be penalised using the GLS. The reason is to avoid returning back by the search to the same local minimum already visited.

In this case, the GLS encourages exploring the other different parts of the search space in order to access other solutions that could be better than the local minimum solution found so far. In such technique, four move operators of the improvement heuristics in [15] are used to improve the quality of the solutions. Those move operators are the 2-Opt, relocate, exchange and cross moves. The 2-Opt move is applied to work as an intra-route move operator by swapping two old edges with two new edges within a single route. The other moves are applied as inter-route move operators that are used between any two tours. For example, the exchange move exchanges two customer visits between two routes. The relocate move has the responsibility of relocating a customer from a route to another route. On the other hand, the cross move swaps the end portions of two vehicle routes. It is

worth noting that the Guided Local Search has used two types of vehicles to visit customers. Those two types are either real or virtual. The real vehicles should respect the constraints imposed by the VRPTW problem. On the other hand, there is only one virtual vehicle that visits all the customers that are not visited by the real vehicles. Of course, the constraints imposed on the real vehicles are not applied here on the virtual one. However, the quality of the solution that has a virtual vehicle will be less than the quality of the solution that has all the customers visited by real vehicles.

In [16], a local search method called Large Neighbourhood Search LNS is investigated. This technique explores a large neighbourhood of a solution by selecting a number of customer visits to remove from the routing plan, and re-inserting these visits using a constraint-based tree search. The set of customer visits chosen for removal and re-insertion defines a move. Of course, many factors that are related to this search technique are discussed in [16] such as what is the size of the visits to be removed, how much such visits relate to each other and how the visits are chosen. In this large neighbourhood search, the size of the customer visits, to be removed, is dynamic. So, if the size is 3 and does not improve the solution, then this size is increased by one over time to try to enhance the quality of the solution.

In the LNS approach, the customer visits, to be removed, have to be somehow related. Therefore, any two customer visits relate to each other in terms of how close in distance they are from each other and whether the two customers are members of the same vehicle. Furthermore, the visits, to be removed, are chosen using a randomised choice method. After choosing the visits, a branch and bound technique and the Limited Discrepancy Search are used to re-insert the removed visits in the routing plan again in order to reduce the cost of the solution. Using the branching heuristics of the branch and bound technique mentioned in [16], a visit might have a collection of insertion points from cheapest to most expensive. The visit could be inserted in anyone of the insertion points. Then, the sub-problem of re-inserting the remaining removed visits is solved, whenever a visit is inserted. In the Limited Discrepancy Search, when the Discrepancy limit is small, a large number of attempted moves of re-inserting the removed visits will be made with little exploration of possible insertions, from cheapest to most expensive, for each removed visit. For higher values of the Discrepancy limit, the opposite situation holds.

The multi-start local search MSLS is a two-phase non-deterministic approach that is mentioned in [17]. In the first phase, a construction heuristic, similar to the one used in the Reactive Variable Neighbourhood Search RVNS described in Section 2.7.5, is used to create initial solutions. The routes of the initial solutions are seeded from a set of customers who are either the farthest from the depot or the earliest deadline for service. The unrouted customers that are close geographically to at least one of the visited customers of a partial route are only considered for insertion. After creating the initial solutions, the approach of Injection Trees IT that is considered as an extension of the well-known Ejection Chain approach EC explained in Section 2.7.5 is used to reduce the number of routes. This process of reducing the number of routes lasts as long as there is a possibility for more reduction.

The Injection Trees approach IT differs from the Ejection Chain approach EC in a number of aspects. The IT approach does not eject any customer in order to make room for a new customer as in the EC approach. Instead, the IT approach directly inserts customers in the target route even if it leads to violation of time window or capacity constraints. Then, IT removes an unlimited number of customers from the target route if they can be inserted directly to a neighbouring route without having to remove customers there to maintain feasibility. For speed up reasons, IT eliminates shorter routes first and choose only geographically close routes for insertion (i.e. distant routes from those shorter routes are ignored). A reordering procedure that depends on simple intra-route re-insertions is used within IT as in EC in order to increase the number of feasible relocations of customers.

Then in the second phase, a modification of the CROSS-exchanges local search, which swaps two segments of consecutive customers between any two routes as in Sections 2.7.3 and 2.7.5, is applied on solutions that have the minimum number of vehicles in order to reduce distance. In this new version, when exchanging two segments of customers between two routes, the local search of CROSS-exchanges tries two kinds of moves. The first move is to preserve the order of the customers in the exchanged segments while the second move is to invert the order of the customers. Also for each removed segment, CROSS-exchanges considers the most promising insertion position (which is the closest customer geographically) in the target route for placing that segment. Furthermore, the most promising moves are

tried first in a first-accept strategy. These moves consist of first removing segments that include customers closest to the customers on the other route and inserting them selectively. Later, the best solution identified by the MSLS is post-optimised using a Threshold Accepting method TA as in Section 2.7.6.

2.7 Metaheuristics algorithms

Routing problems and even scheduling problems are solved in the literature [18] [19] using many metaheuristic techniques. Metaheuristic techniques are approximate algorithms that are used to solve hard combinatorial optimisation problems where classical or exact methods are not able to do so. Metaheuristics are almost different configuration systems of a number of different components that are derived from classical heuristics, artificial intelligence, biological evolution, neural systems, social systems and statistical systems. In literature, such metaheuristic techniques are called Adaptive Memory Programming AMP [20]. They are called AMPs because they use some sort of adaptive memory towards ultimately reaching the goal that is a very good solution and not necessarily optimal for the problem in hand. Examples of those AMPs [15] [21] [20] [22] [23] [4] [16] in literature are tabu search, scatter search, genetic algorithms and ant colonies.

The VRPTW problem is one of the problems that are solved using many metaheuristic techniques. In this section, those VRPTW metaheuristic techniques will be reviewed and mentioned. Most of the metaheuristic techniques of VRPTW use a special kind of representation of solutions called permutations or other types of some elaborated sort. Up to these recent days, VRPTW is solved widely using many different types of metaheuristics such as multiple ant colonies [4], simulated annealing [24] [25], embedding taboo searches [23] [26], guided local search [15], large neighbourhood search [16], hybrid algorithms [22] [27], reactive variable neighbourhood searches [5] [8] and threshold accepting techniques [28]. The main aim of all such metaheuristics is to find very good solutions for the problem in hand.

2.7.1 Simulated Annealing

Simulated Annealing is an optimisation technique that has some similarities with the physical process of annealing of solids. In the physical process of annealing, a

solid is heated to a high temperature T and gradually cooled to have at the end the kind of molecular geometric structure that is needed. In order to get the kind of molecular geometric structure wanted, the temperature T is lowered gradually. If the solid is cooled very quickly, then this process may end up with a molecular geometric structure that is not desired. In the case of cooling the solid very quickly, the atoms of the solid do not have enough time to align themselves to form a fine shape with a minimum energy state. However, if the solid is cooled very slowly, it gives the atoms enough time to align themselves to reach the minimum energy state of a fine shape.

In combinatorial optimisation, the states of the solid correspond to the feasible solutions. The energy at each state represents a value of the objective function. Of course, the minimum energy represents the optimal solution value. Simulated annealing is a metaheuristic technique that allows the search to move to a neighbouring state even if the move causes the value of objective function to become worse. Simulated annealing SA guides any local search as mentioned below. SA starts with a high temperature T and an initial state S. Then, the best state S_{best} is set to be equal to the initial state S. Later, a neighbourhood operator or a move is applied to the current initial state S to yield the state S^*. If the cost value of S^* is lesser than the cost value of S, S^* becomes the current state.

However, S^* becomes the current state S with a probability $e^{(cost(S)-cost(S^*))/T}$, if the cost value of S^* is greater than the cost value of S. Note that if S^* does not become the current state, then S remains the current state. Also, the current state S may replace the best state S_{best} found so far, if the state S is better in quality than the best state S_{best}. Of course, the application of the neighbourhood operator and the acceptance of newly generated states are repeated until a fixed number of iterations is reached, an amount of CPU time is elapsed or no improvement is achieved for a number of iterations. If a local minimum energy is reached after an iteration of SA, then the temperature value of T is lowered and the application of the neighbourhood operator and the acceptance of newly generated states are repeated as explained above.

In the literature of vehicle routing, simulated annealing approaches as in [24] [25] are used. In [24], two main neighbourhood operators are applied. The first is called the $\lambda-$interchange operator while the second is called the $k-$node interchange

operator. The $\lambda-$interchange operator works as explained in Section 2.6. The $k-$node interchange operator works as mentioned in [9] as follows. Customer c_i and his successor c_j on a route and the two customers closest to customers c_i and c_j, but not on the same route, are removed. Then, the neighbourhood is defined by trying to insert these four vertices in any other possible way. As this neighbourhood is quite large, only the k most promising nodes are checked. Also in [24], the issue of using a short-term memory or tabu list, as in Section 2.7.2, is tested as a basis for enhancing the simulated annealing approach.

Another example of the usage of the simulated annealing approach is the one used in a two-stage hybrid local search [29], called as SA+LNS, for tackling the VRPTW problem. At first in one stage using simulated annealing, the hybrid local search reduces the number of routes. Then in the second stage, it minimises the travel cost using a large neighbourhood search LNS by relocating a large number of customers from their positions in a routing plan or a solution to other positions.

In the simulated annealing stage, the neighbourhood solutions are created using a number of move operators, which are 2-exchange, Or-exchange, relocation, crossover and exchange. In order to reduce the number of routes effectively, the simulated annealing algorithm chooses good solutions according to an evaluation function that considers the three factors mentioned below.

P1- A factor that prefers solutions that minimise the number of routes.

P2- A factor that prefers solutions that have routes with few customers over solutions with more evenly distributed customers between the routes.

P3- A factor that prefers solutions where customers on the smallest route can be relocated to other routes in such a way that causes no constraint violations or if the time window violations are as minimal as possible.

In the LNS stage, minimising the travel cost depends on two main components. The first component is to choose a set of related customers to be relocated from their positions in a solution. Then, the second component is to explore a set of neighbourhood solutions after such related customers are re-inserted again in different ways. For more information about the LNS, check Section 2.6.

In [30], a multi-start Simulated Annealing or a hybrid system that associates a non-monotonic Simulated Annealing technique to a Hill Climbing strategy with

Random Restart is used to tackle the VRPTW problem in order to reduce the travelled distance as a primary objective. In the hybrid system, the SA strategy creates at first an initial solution using a Push-Forward Insertion Heuristic (PFIH) of Solomon [1].

Then, the SA strategy begins to evolve the initial solution and tries to escape the local minima using five neighbourhood operators - swap, insert, scramble, inversion and an operator that removes a number of customers in a route and reinserts them using PFIH. At a later stage, the hill climbing strategy is used to refine the worse solutions computed by the SA stage. After implementing the SA and the hill climbing stages, the strategy of random restart is used to let the hybrid system start from a different point in the solution space. At the end and out of thirty random restarts for the hybrid system, the solution with the shortest travelled distances is chosen to represent the best solution found so far for a VRPTW problem instance.

2.7.2 Tabu Search

Tabu search techniques [26] [2] [3] [20] [31] [25] [23] are studied a lot in the literature. A basic tabu search is usually built around a local search but it has additional components such as a diversification technique, an intensification technique, a short-term memory or tabu list...etc. In a basic tabu search, an initial solution is created and assigned to be the current solution S. Also, the best solution S_{best} is set to be equal to the current solution S. The tabu search is run for a number of iterations, until an amount of CPU time is elapsed or until no improvement is occurred for a number of iterations.

In iteration k, one of two main methods of the objective function, intensification or diversification, is used to create the neighbourhood $N(S)$ of the current solution S. The intensification method has the responsibility to penalize the solutions that are far in quality from the current solution S. On the other hand, the diversification method punishes the solutions that are close in quality to the current solution S. After applying either the intensification or the diversification method of the objective function, the neighbourhood of the current solution S is explored and the best neighbour solution S_{bneigh} in the current neighbourhood is selected as the new current solution S. Furthermore, the best solution S_{best} found so far is updated, if the new current solution S turns out to be better in quality.

In order to escape from a local optimum, the current solution S is set to be the best neighbour solution S_{bneigh} in the neighbourhood even if this solution is worse than the current solution already got. In order to prevent re-visiting recently visited solutions (i.e. cycling), a short-term memory or a tabu list is used. This short-term memory has a tabu on every move done by a move operator in solutions during previous iterations. In vehicle routing, this tabu could be put on an edge done between two nodes in a vehicle or on a node allocated to a vehicle.

The duration that an edge or a node in a vehicle remains tabu is called tabu-tenure and usually lasts for a number of iterations. Of course, the short-term memory varies in its size and its structure from a problem instance to another. Now in iteration $k+1$, the neighbourhood of the current solution is created with excluding those edges or nodes allocated to specific vehicles that are considered as tabu from the neighbourhood solutions. If an improvement is about to happen on the best solution S_{best} at some iteration but the move that would cause the improvement is tabu, then the tabu list that has a tabu on that move will be overruled or overridden by a criterion called the aspiration criterion.

In [23], a network flow-based tabu search heuristic is used to try to solve vehicle routing problems with capacity constraints CVRPs. This network flow-based tabu search heuristic is a new local search that depends on two major components or local searches in creating moves to have new quality solutions. The first component is called the network flow model, whereas the second is called the direct customer swap procedure. The network flow model is used to evaluate simultaneously several customer ejection and insertion moves. If the number of moves to be tested is 1, then the model will evaluate the best ejection/insertion move.

If the number of moves is greater than 1, then the model evaluates the best composite move, which is a combination of multiple ejection and insertion moves. In the easier case when the number of moves is equal to 1, the model is capable of knowing the encouragement/discouragement value of removing customer c_j from a route R_i. In other words, if the total load to be delivered of route R_i is very much so below the maximum capacity value of a vehicle, then the model discourages removing customer c_j. On the other hand, if the total load to be delivered of route R_i is very high, then the model encourages removing customer c_j. Furthermore, the model knows the encouragement/discouragement value of inserting the same customer c_j

into another route R_k. The direct swap procedure is a local improvement approach that swaps customers between two different routes. In the direct swap procedure, the best swap is selected among a number of swap moves.

In this approach, either the network flow model or the direct swap procedure could be used to make the best move to have a new current solution S if possible. Then, the current solution is exposed to further improvement by a 3-Opt local search or another tabu local search procedure that uses the two moves eject and swap. In order to escape local optima in the network flow based model and the direct swap local procedure, a tabu search memory is used. Tabu search restrictions are imposed on three different neighbourhood moves: dropping a customer from its current route, inserting a customer into a different route and swapping two customers between routes. This approach uses also a diversification technique that has a long-term frequency memory, which has similar effects somehow to the short-term memory or tabu list but in the long-term. The purpose of the long-term memory is to record the frequency of a customer being assigned to a specific route. Also, this approach uses an intensification technique that depends on a restart/recovery strategy. This restart/recovery strategy might select a solution of a list of elite solutions and assigned it to be the new current solution if necessary.

The researchers in [26] have presented a probabilistic technique to diversify, intensify and parallelize a local search adapted for solving the vehicle routing problem. They use this technique with two tabu searches that are developed for the VRP and VRPTW problems. This technique diversifies the search by exploring solutions that are very different from each other. Then, it intensifies the search in order to identify better local optima in a promising region of a set of feasible solutions. The local search of this technique works in parallel and generates different solutions as mentioned in [26] because it has some random component. This technique passes through two major phases. The first phase is called the initialisation phase while the second is called the intensification and diversification phase.

During the initialisation phase, different initial solutions are generated with the local search. Then, each tour belongs to a solution is labelled with the value of that solution. Tours that have one customer each are removed. The remaining tours of each solution are added to a set called T that represents all the tours generated from all the solutions. The set T is sorted in an increasing order according to the values of

the labels of the tours. In the diversification and intensification process, two major stages happen. In the first stage, the set T is cloned to have a new set called T'. Then, a new solution S is initialised to nil and later built out of T'. The process of building a new solution depends on the following three steps: (a) A tour is selected probabilistically and added to be part of the new created solution S. (b) All the tours that visit the same customer nodes of the selected tour are removed from T'. (c) Steps a and b are repeated until no tours are left in T'. After the process of building a solution, the new solution S could be built partially. In other words, some customers might be not visited in the solution S - i.e. infeasible solution. For that in the second stage, a feasible solution S' is built to include those chosen tours in S and all the remaining customers by including them in one tour or possibly more. Thereafter, the feasible solution S' is improved using the local search. Then, the tours of the improved solution are labelled as in the initialisation phase, the set T is updated such that the set T is not exceeding a certain limit and therefore tours with worst values are removed and the intensification and diversification search continues until a stopping criterion is satisfied.

Tabu search approaches are used with other meta-heuristic approaches like evolutionary algorithms in Section 2.7.3 as in the guided cooperative search in [32] [33] [34] to tackle the VRPTW problem. The guided cooperative search approach uses a solution warehouse or a population of solutions to combine the efforts of the search threads of several meta-heuristics. If a number of search threads use the same meta-heuristic, such search threads are differentiated by using different initial solutions and parameter settings.

The meta-heuristics used are the unified tabu search, the Taburoute search, an evolutionary algorithm with the order crossover (OX) and an evolutionary algorithm with edge recombination (ER). Independent search threads of the meta-heuristics mentioned earlier send their improved solutions to the post-optimisation algorithms like 2-Opt, 3-Opt, Or-Opt and ejection chain. These solutions are post-optimised and then they are sent as quality solutions to the solution warehouse. Of course, the solution warehouse uses a pattern-identification mechanism that support cooperative search with capabilities to create new information and guide global search in each of the search threads of the meta-heuristics mentioned above.

For that, the solution warehouse sends back to individual search threads of the

meta-heuristics mentioned above not just new solutions to diversify the search but also information about promising and unpromising patterns in the solution space. Note that the solutions, should be sent, are selected by the solution warehouse randomly according to probabilities biased towards the best solution. A frequent (or an infrequent) pattern that is related to a set of arcs consists of arcs appearing with high (or low) frequency in a set of solutions. By fixing or prohibiting specific solution attributes or arcs in an individual search thread, the search in that search thread focuses on desired regions.

2.7.3 Evolutionary Algorithms

An evolutionary algorithm [18] [35] is an adaptive search heuristic that maintains a population of individuals or candidates. The individuals of a population can be seen as entities of artificial chromosomes of fixed length. Each chromosome has a solution and a fitness value describing the goodness of the solution. In evolutionary algorithms, the representation of chromosomes is a vital and an important component in the methodology of problem solving. Therefore, the representation of chromosomes may vary from an evolutionary algorithm to another in a way depending on the sort of the problem should be solved. As a consequence, there are many different kinds of representation schemes for the chromosomes in the literature and examples of those schemes are the binary-code representation, the permutation-based representation or others of some elaborated sort.

Evolutionary algorithms have the general purpose of trying to solve problems in a way same as the other meta-heuristic techniques. Therefore, one of its purposes is to derive the search from a population of poor quality solutions into a population of good quality solutions. For that, a number of components, such as selection, recombination, mutation, replacement, are used in addition to the representation component of chromosomes. All the components mentioned are vital parts of the evolution process of an evolutionary algorithm. An evolutionary algorithm is run usually for a number of generations or iterations, until an amount of CPU time has elapsed or until some convergence criteria are met. Now in evolutionary algorithms EAs, there are two major kinds of such algorithms: The first is called the steady-state EA while the second is called the generational EA. In a steady-state EA, one or two offspring chromosomes are produced and replace one or two individuals that

are worst in a population. However in a generational EA, an offspring population might replace the entire population of individuals every generation.

In an EA, two or more parent chromosomes are selected using a selection operator such as a proportionate or a tournament selection operator. Then, every two or more selected parent chromosomes are recombined using crossover operators, such as the one-point and two-point crossovers for the binary-code representation or special crossovers for the permutation-based representation, to create offspring chromosomes. Later, the offspring chromosomes are exposed to one or more mutation operators, such as the standard-bit mutation operator for the binary-code representation or special mutation operators for the permutation-based representation, to improve further on the quality of their solutions. Afterwards, the offspring chromosomes of good quality solutions replace the chromosomes of bad quality solutions in a population. In the literature, there are many different types of evolutionary algorithms [18] [35] that are studied and applied such as genetic algorithms [36] [37] [38] [39] [40] [41] [11] [21], classifier systems [42], genetic programming [43], evolutionary strategies [2] [3] [44] and evolutionary programming [18] [45].

The hybrid GA mentioned in [22] uses an integer string representation of length n with group information of genes attached at the end portion of each chromosome and incorporates with other techniques such as group information updating and local search improvement. For example, the chromosome in Table 2.9 has initial grouping information, namely [5 4 3]. At the end of the chromosome, the grouping information of [5 4 3] tells how many routes are in the routing plan and how many customers are going to be visited in each route - Note that grouping information is created using a push forward insertion heuristic that inserts customers sequentially into routes. In the information of the feasible grouping [5 4 3], there are three routes as explained between the brackets and each route has a number of customers to be visited. The first number says that there are five customers should be visited and they are 2, 5, 8, 10 and 12. The second number between the brackets says that four customers should be visited in the second route. On the other hand, the third number says that three customers should be visited in the third route. Each chromosome in this hybrid GA could have different types of groupings.

Now, the appropriate selection of groupings for chromosomes is the major focus

```
2 5 8 10 12 9 3 7 6 1 4 11 [5 4 3] Initial grouping information

2 5 8 10 12 9 3 7 6 1 4 11 [4 4 4] Other possibilities
2 5 8 10 12 9 3 7 6 1 4 11 [3 6 3]
2 5 8 10 12 9 3 7 6 1 4 11 [5 5 2]
```

Table 2.9: Chromosome Groupings

in this hybrid GA. Therefore, the hybrid GA makes use of a local search method, I-Interchange, to search for new groupings for each chromosome. For instance, if the initial grouping is [5 4 3] as in Table 2.9, then new combinations of groupings, such as [4 4 4], [3 6 3], [4 5 3] and so on, will be searched for by that local search. The search will stop when there is a new better grouping in terms of total of travelled distances - Note that the primary goal in this hybrid GA is reducing distance. Once the grouping update is finished, the chromosome with new grouping information will undergo further local search improvement. Behold that not all chromosomes can undergo this local search improvement. The whole improvement procedure, including grouping update and local search, is repeated several times and then the population will undergo the same selection and reproduction processes in a simple GA. The hybrid GA uses the tournament selection operator [36], the PMX crossover operator [18] and a mutation operator to swap nodes in the same chromosome.

A new hybrid evolutionary algorithm is presented in [27] to obtain feasible solutions. The hybrid evolutionary algorithm HGA+EA is based on a hybridisation of a hybrid genetic algorithm HGA and an evolutionary algorithm EA consisting of several local search and route construction heuristics. In the first phase of the hybrid evolutionary algorithm, the HGA is used to obtain a feasible solution. Then in the second phase, the solution is improved using an evolutionary algorithm. The HGA algorithm was built on top of Galib, MIT genetic algorithm [27], and uses well-known heuristics to derive the search process. The HGA directly applies the genetic operators to solutions - i.e. a solution is a set of feasible routes. The initial population for HGA is created with a random heuristic that selects and inserts customers into routes in a totally random manner. The selection operator, the proportionate operator with the roulette-wheel sampling scheme, consists of choosing

two individuals (i.e. parent solutions) within the population for mating purposes.

Two crossover operators, IB-X(k) and IRS-X(k), are used in the HGA for recombination purposes. The first crossover IB-X(k), known as the Insertion- based crossover, creates an offspring solution from two parent solutions P_1 and P_2. The crossover process is iterated k times to create k offspring routes. Each offspring route is created by combining a route R_1 of the parent solution P_1 with a subset of customers created from a number of routes of the parent solution P_2, which are the nearest neighbours to the route R_1 of the parent solution P_1. Note that one subset of customers a time is formed at every iteration k in order to help in the creation of an offspring route. An offspring route R_1^* is created partially by removing a number of customers from a route R_1 of the parent solution P_1. Note that a customer is chosen, for removal, based on waiting time, large distance separating him from his neighbours or randomly. Then, the partially altered offspring route R_1^* is combined with the subset of customers using a modified version of the Solomon insertion heuristic $I1$.

On the other hand, the second crossover IRS-X(k), which stands for Insert Related Solomon, also creates an offspring solution from two parent solutions P_1 and P_2. But, the crossover removes a number of related customers from the parent solution P_1. Here, customer relatedness refers to customers that can be somewhat interchanged on the basis of similar selected features such as position, near similar time intervals and/or visit time. This crossover creates offspring routes iteratively in the same way, described in the previous paragraph, but the removal strategy of customers from the parent solution P_1 is changed as described in this paragraph. Also, the insertion procedure used is a variant of the large neighbourhood search or the LNS method [16], which uses an insertion cost function of a Solomon heuristic.

Furthermore in the HGA, the two mutation operators RSS-M(k) and LNS-M are used. The first mutation operator RSS-M(k), which stands for Reinsertion Shortest Solomon, has the responsibility to eliminate a number of k routes having only a few customers and to re-insert their customers into alternate routes using an insertion technique similar to Solomon insertion heuristics. The second mutation operator LNS-M or the Large Neighbourhood Search operator has the responsibility to remove a random number of related customers from a solution and re-insert them using the LNS method into alternate routes.

After the evolution process of the HGA, a feasible solution is handed to an evolutionary algorithm. The evolutionary algorithm EA considers each feasible route as an individual. As the handing of the initial solution to EA is done, the routes of the initial solution are stored into a route pool, which forms the initial population of the EA. Thereafter, the tours are put in a random order and all possible pairs of two routes, two-route combinations, are picked according to this order. Once a pair or a combination of chosen parent routes is selected, four crossover operators of local searches and route construction heuristics are randomly applied on the two parent routes in order to generate offspring routes.

Those four crossover operators are CROSS-exchanges crossover CE-X, Insertion crossover I-X, Rebuilding crossover R-X and Random Rebuilding crossover RR-X. The CE-X crossover is based on swapping segments of varied size of consecutive customers between the parent routes. The I-X crossover selects customers in random order from a parent route R_1 and tries to insert them into the other parent route R_2. The R-X crossover marks the customers in the two parent routes as unrouted and focuses on the usage of a cheapest insertion heuristic like I1 of Solomon to insert the unrouted customers again into the best possible places in the parent routes. The RR-X crossover marks the customers in the two parent routes as unrouted and selects the customers in random one at a time and insert them into the best possible places in a parent route - Note that at each customer insertion, a parent route of the two parent routes is selected randomly.

Finally, the offspring routes generated, using these crossover operators, are mutated by randomly applying the two mutation operators P-M and R-M. The P-M operator stands for the Permute Mutation operator, whereas the R-M operator stands for the Reorder Mutation operator. Each one of the two mutation operators is applied to a single route at a time. The P-M mutation operator is based on Or-Opt exchanges through attempting to insert all possible segments of three consecutive customers between all possible pairs of consecutive customers in the route. Then, segments of two consecutive customers and one customer respectively are examined also. The other mutation operator R-M focuses on reordering customers whom can be serviced in various orders. More precisely, the customers for whom time windows are tight are first inserted in a route. Then, the remaining customers are inserted using a variant of a Solomon insertion heuristic.

A route-directed hybrid genetic algorithm RHGA is addressed in [46] for the vehicle routing problems with time windows. The RHGA is a steady-state GA that uses two populations of individuals pursuing two different objectives, read below. The first population evolves individuals with V tours to minimize the total of travelled distances, whereas the second population evolves individuals with $V-1$ tours to minimize the temporal constraint violation. The two populations interact with one another whenever a new feasible solution emerges in a way reducing the number of tours that should be imposed as a limit on future solutions. The RHGA algorithm combines the simultaneous evolution of two populations and the partial temporal constraint relaxation to improve the quality of solutions. The algorithm uses, in each of the two populations, genetic operators that maximize as many as possible the number of customers served in a solution and then relaxes the issue of temporal constraint violation in order to insert the remaining unvisited customers. The genetic operators are simply applied to a population of solutions or solution individuals rather than a population of encoded solutions or chromosomes.

The solutions of the initial populations in RHGA are first generated using a sequential insertion heuristic in which customers are inserted in random order at randomly chosen insertion places within routes. In this algorithm, the proportionate selection operator that uses the roulette-wheel sampling scheme is used. Also, two recombination operators, IB-X(k) and IRN-X(k), are used on solution individuals. The first of these two recombination operators is similar to the IB-X(k) crossover, known as the Insertion-based crossover talked about above in the HGA phase of the HGA+EA approach [27], and it recombines one route R_1 at a time from the parent solution P_1 with a subset of customers created from a number of routes of the parent solution P_2, which are the nearest neighbours to the route R_1 of the parent solution P_1.

However, the second crossover IRN-X(k), which stands for Insert within Route Neighbourhood, works as the first crossover but the k routes of the parent solution P_1 are considered all at once rather than one at a time. Then, a partial offspring solution is created after removing a number of customers, using some sort of removal strategy, from the k routes of the parent solution P_1. Later, all subsets of customers, created from the nearest neighbour routes of the parent solution P_2, are re-inserted using a variant of an insertion heuristic into the partial offspring solution. Note that

each subset of all the subsets of customers is created from a number of routes of the parent solution P_2, which are the nearest neighbours to some route R_i of the k routes of the parent solution P_1.

Furthermore, the five mutation operators LNSB-M(d), EE-M, RS-M, RSR-M(I) and RC-M(I) are used. The LNSB-M(d) mutation operator is a Large Neighbourhood Search Based operator that removes repeatedly related customers and reinsert them using a constraint-based tree search. In this operator, customer relatedness defines a relationship linking two customers based upon specific properties - e.g. how close the two customers to each other and/or whether they are members of the same route for example. The Edge Exchange EE-M and the Repair Solution RS-M mutation operators are inter-route improvement operators. The EE-M is inspired from the λ-interchange mechanism and performs re-insertions of customer sets over two neighbouring routes.

In this operator, each customer of a route is explored for re-insertion in one of his neighbouring routes - Here, the number of the neighbouring routes is equal to 2. The neighbouring routes of a customer are selected such that the distance separating their centroid from a customer location is minimal. The RS-M operator focuses on exchanges that involve one illegal customer visit. So, each illegal visit in a route is exchanged with another customer legally visited or a sequence of two legally visited customers. The RSR-M(I) operator, which stands for Reinsert Shortest Route, eliminates the shortest route of a solution, reducing by one the total number of routes. The RC-M(I) mutation operator, which stands for Reorder Customers, is an intensification procedure that is used to reduce the total of travelled distances of feasible solutions by reordering customers within a route.

In [7] [47], a steady-state hybrid genetic algorithm is used with a kind of crossover called the natural crossover. The most notable feature of the natural crossover is that it uses the 2D image of a solution itself for chromosomal cutting. The initial population of the hybrid genetic algorithm is created using a stochastic version of the insertion procedure I1 of Solomon. The selection operator used is the binary tournament scheme in order to select two chromosomes - Note that the 2D image of routes represents a chromosome. Once the 2D images of two solutions A and B are chosen, free curves and figures (i.e. circles, squares, lines...etc) are drawn on the 2D space, where customers are located. The free curves and figures always partition

the 2D or the chromosomal space into two equivalent classes marked as white and grey areas.

Every customer belongs to one of the two classes. If a customer belongs to the white area, it will have a circle of a black colour. If a customer belongs to the grey area, it will have a circle of a white colour. Now, for every arc of the parent A, if both the start-point and end-point of the arc are marked black, then this arc survives in the offspring solution. Also, for every arc of the parent B, if both points are marked white, the arc survives in the offspring. Then a number of disconnected segments are going to appear in the offspring solution in a way that makes it look invalid. Later, a repair algorithm is used to add the missing arcs and to make the offspring solution valid. Once an offspring solution is created, a mutation operator is used to improve the quality of the solution. Also, three local search heuristics, Or-Opt, crossover and relocation, are used in sequence in order to improve further the quality of the offspring solution. After improving the offspring solution, the offspring solution may replace any one of the two parent solutions in the population, if it is better. Otherwise, it replaces the worst solution in that population.

A parallel hybrid evolutionary metaheuristic is presented in [2] for the VRPTW problem. This algorithm uses a hybrid metaheuristic that consists of two procedural phases. The first phase minimizes the number of vehicles by means of a (1, λ)-evolution strategy, whereas the second phase minimizes the total of travelled distances using a tabu search algorithm. In the first phase, a start solution S is created using a modified savings algorithm, which takes into considerations the time window constraints. The start solution S is assigned to be the best solution S_{best}.

Then, the (1, λ)-evolution strategy tries to create a number of feasible neighbouring solutions of the solution S using the move set - Or-Opt, $2 - Opt^*$ and 1-interchange. Also, for each feasible neighbouring solution generated, a modified Or-Opt is used to reduce the number of vehicles. At the end, the (1, λ)-evolution strategy must generate feasible neighbouring solutions of size λ and choose the best solution of them. The best solution of the set of feasible neighbouring solutions is chosen and assigned to be the new start S. In addition, if the new start solution S is better than the best solution S_{best} found so far, then S replaces S_{best}. This strategy continues in its work until an amount of CPU time has elapsed.

The best solution with the least number of vehicles found in the first phase is

passed over to the tabu search - second phase. In the second phase, no attempts are made to reduce the number of vehicles further. The aim in this phase is to reduce the total of travelled distance done by vehicles. The second phase generates feasible neighbouring solutions of a start solution S using the move set described above. A feasible neighbouring solution is selected, if none of the connections between customers or customers and the depot of its routes are set tabu or if the aspiration criterion is met. The aspiration criterion means here that the tabu status of one or more connections of the routes in a neighbouring solution is ignored, if the neighbouring solution represents a new best solution.

Later, the best evaluated neighbouring solution of a start solution S is selected and assigned to be the new start solution S. Also, the tabu list is updated with the connections of the routes of the best-evaluated neighbouring solution or the new start solution S. Now, if the new start solution is better than the best solution S_{best} found so far, then S replaces S_{best}. This tabu search works until an amount of CPU time has elapsed. Later the researchers in [2] have presented another parallel two-phase metaheuristic in [3] for the VRPTW. This metaheuristic combines a (μ, λ)-evolution strategy and a subsequently executed tabu search.

In [48], a meta-heuristic approach, called Active Guided Evolution Strategies AGES, is introduced especially for tackling large VRPTW problem instances with 200, 400, 600, 800 and 1000 customers. The AGES approach has two phases. In the first phase, an initial solution S^0 is generated using a hybrid insertion heuristic of some sort. Then, a filling procedure is used to minimize the number of routes in the solution S^0. If the number of routes in the solution S^0 cannot be reduced any more, S^0 is assigned to be the best global solution. Later, the AGES approach assigns the best global solution to be as the current solution S_i to start improving on in the second phase. In the second phase, there are two stages - a guided local search GLS and a $(1+1)$-evolution strategy.

In the first stage, the guided local search GLS works by penalizing long arcs, which appear in the current solution S_i - local minimum. In the current solution S_i, GLS defines a penalized arc with a maximum value that reflects the cost of that arc and how many times the arc is penalized. Then, GLS is restricted to use a set of geographically closest routes to the defined arc in S_i and is applied to create a neighborhood of solutions from the move operators (relocate, 1-interchange and 2-

Opt*). The best neighborhood solution S_{bneig} is selected and compared with the best global solution and it updates it if it is better in quality. Then, the best neighborhood solution S_{bneig} might also replace the current solution S_i in case that solution is better. Of course, GLS continues in working as described in this paragraph and if no improvements are found, the second stage starts working.

The second stage uses a $(1+1)$-evolution strategy that depends on a local search procedure that is similar to the large neighborhood search LNS [16]. Of course, the large neighborhood search LNS uses a set of routes in the parent solution S_i that are closest geographically to a penalized arc. Then, LNS removes a number of related customers from that set of routes in order to re-insert them again. An offspring solution is created and compared with the best global solution and with the parent solution S_i and if it is better in quality than any one of them, it may replace it. This stage is an iterative stage and once it cannot improve the current solution S_i, it switches back to the first stage and so on.

2.7.4 Ant colony systems

Ant Systems and Ant Colony Systems are investigated and studied in [49] [50] [51] [52] [4] [53] [54] [55] [56] [6] and [57] as efficient techniques that outperform other meta-heuristic techniques when it comes to try to solve some particular problems, such as TSP, VRPTW, QAP and SOP mentioned in Section 2.3, in a very short amount of CPU time. The ant systems in [49] [51] [54] are the origin of all research efforts with ant algorithms and were first applied to the travelling salesman problem TSP. In a TSP problem, a number of m ants in the ant system utilize the edges of the graph of a TSP problem in order to build tours or solutions - i.e. a tour or a solution for each ant.

Each edge of the TSP graph has a visibility function or a cost measure $\eta_{c_i c_j}$ that equals the inverse of the distance value between any two cities c_i and c_j and a desirability measure $\tau_{c_i c_j}$ or a pheromone trail, which is initialised using the inverse of a solution value created by the nearest neighbour heuristic and is updated at run time by artificial ants. When the ant system is applied to symmetric instances of the TSP problem, the pheromone value on each side of the two sides e_{ij} and e_{ji} of an edge is always the same because the distance of each side is the same. However, if the ant system is applied to asymmetric instances of the TSP problem, it is possible

that the pheromone value on one side is different from the pheromone value on the other side because both sides might have different distance values.

The ant system has an initialisation step and a cycle step. In the initialisation step, the pheromone trails of all the edges of a graph are initialised using a value that matches the inverse of a solution value created by the nearest neighbour heuristic. Now in the cycle step, a number of m ants is created to build a solution each. Each ant generates a complete solution by choosing the cities using a probabilistic state transition rule called the random-proportional rule or the exploration mode as in the equations 2.27 and 2.28, where $N_{c_i}^k$ is the set of cities that remain to be visited by the ant k positioned on city c_i and β weighs the relative importance of the static heuristic value $\eta_{c_i c_j}$. According to these equations 2.27 and 2.28, ants prefer to move to cities, which are connected by short edges with a high amount of pheromone. In this exploration mode of the equations 2.27 and 2.28, an ant k located, on city c_i, chooses with a great probability or chance the edge that is connected to another city c_j and has a great value of a combination of two values - how short the edge is and how attractive the edge is from the pheromone point of view.

$$p_{c_i c_j} = \begin{cases} prob_{c_i c_j} & \text{, if } c_j \in N_{c_i}^k \\ 0 & \text{, otherwise} \end{cases} \tag{2.27}$$

$$prob_{c_i c_j} = \frac{\tau_{c_i c_j} \cdot [\eta_{c_i c_j}]^\beta}{\sum_{c_l \in N_{c_i}^k} \tau_{c_i c_l} \cdot [\eta_{c_i c_l}]^\beta} \tag{2.28}$$

Once all the ants have completed building their solutions, a global pheromone update rule is applied to update the pheromone trails of all the edges of the graph according to the equations in 2.29 and 2.30. Note that in the equations 2.29 and 2.30, the evaporation parameter ρ represents a value between 0 and 1 and the L_k term represents the length of the tour done by an ant k. In the global updating of an ant system, a fraction of the pheromone evaporates on all the edges of the graph and then each ant deposits an amount of pheromone on edges, which belong to its solution, in proportion to how short its solution is. In other words, edges that belong to many short solutions are the edges, which receive the greater amount of pheromone. Pheromone updating is to allocate a greater amount of pheromone to shorter tours or better quality solutions.

$$\tau_{c_i c_j} = (1 - \rho).\tau_{c_i c_j} + \sum_{k=1}^{m} \Delta\tau_{c_i c_j}^k \tag{2.29}$$

$$\Delta\tau^{k}_{c_i c_j} = \begin{cases} 1/L_k & \text{, if } e_{c_i c_j} \in \text{ tour done by ant } k \\ 0 & \text{, otherwise} \end{cases} \tag{2.30}$$

Pheromone placed on the edges plays the role of a distributed long-term memory. This memory is not stored locally within the individual ants, but it is distributed on the edges of the graph. This memory allows an indirect form of communication called stigmergy. The cycle of the ant system lasts until a fixed number of solutions is generated, a fixed amount of CPU time has elapsed and/or no improvement is achieved during a given number of iterations. The ant system described above is useful to discover very good and competitive solutions for small TSP problem instances up to thirty cities in a very short amount of time. However for larger problem instances, the time required to find such competitive solutions is not possible.

Therefore, the researchers in [51] [50] [52] have included four major types of modifications described below to the Ant System described above to have a new system called Ant Colony System ACS. Firstly, the use of the probabilistic state transition rule in equations 2.27 and 2.28, which uses the exploration mode in almost 100% probability, has changed to a new probabilistic state transition rule that uses the exploration mode in 10% probability and an exploitation mode in 90% probability. In the exploitation mode, an ant k located, on city c_i, chooses the edge that is connected to another city c_j and maximizes a value of a combination of two values - a short edge and an attractive edge. Secondly, the usage of a local pheromone-updating rule is added as a vital component to the work of each ant.

This local pheromone-updating rule comes into play to diminish the pheromone trail of an edge once an ant chooses an edge between two cites c_i and c_j through using either the exploration mode or the exploitation mode. Thirdly, the global pheromone-updating rule is applied only to edges, which belong to the best ant tour or solution. Fourthly, a local search is added to be part of the ant's work. Of course, adding all the four components to the original ant system AS has led to a system that has better performance and can bring very competitive solutions not just on small problem instances but also on larger ones.

Later in [4], a multiple ant colony system is used for the VRPTW problem. In this approach, one colony is used to try to reduce the number of vehicles while a second colony is used to minimize the total of the travelled distances done by the

vehicles. As well, a multiple ant colony system is given in [53] to deal with the issue of time dependency in VRPTW and how to let an ant put a target node in its solution in a way using the pheromone trail and depending also on the departing time at a particular moment from a departure node. Furthermore, a simple ant colony system in [6] is examined with different initialisation techniques and visibility functions of the ants. In [57], a Savings-based Ant System is introduced. The ants in [57] use the well-known savings algorithm, which is adapted to solve vehicle routing problems, as a vital component of the constructive phase of each ant. Here, each ant builds its solution based on choosing a value that is composed of two values. The first of the two values refers to how much large is the saving value of an edge, a combination or a pair of two customers in order to serve the two customers of that edge on the same tour whereas the second value refers to how much attractive that edge is.

Also, an improved Particle Swarm Optimisation (PSO) algorithm, like ACO mentioned above, is presented for the VRPTW problem in [58]. The PSO algorithm is a collaborative population-based search that models the social behaviour of bird flocking and fish schooling. In [58], a new particle coding for the VRPTW problem is used to help convert the discrete combinatorial problem into a continuous one so that the PSO algorithm can be directly applied. Each particle of the PSO algorithm represents a solution and the best position P_i (or the best fitness value) of that particle is recorded. In the PSO algorithm, the best particle among all the particles in a population is referred to with the symbol g.

The PSO algorithm runs for a number of iterations. In any iteration, each particle has a current position in the solution space and a velocity rate, which are used to help in changing its position for the next iteration. For that at first, each particle computes a new velocity rate in a way that depends on its current velocity rate and its current position from its best position P_i and the best position P_g of the whole group of particles. Then afterwards, the new computed velocity rate and the current position of each particle are used to calculate a new position and this results in letting each particle fly to its new position.

According to [58] and as in ACO, the PSO algorithm runs faster and gets quality results and solutions in a way better than that of the GA algorithm. It uses local and global searches and attempts to balance between the global exploration and the local exploitation using a weight called the inertia weight. A larger inertia weight

facilitates the global exploration while a smaller inertia weight facilitates the local exploration.

2.7.5 Reactive Variable Neighbourhood Searches

In [8] [5], a new deterministic approach called Reactive Variable Neighbourhood Search RVNS that is based on a modification of a variable neighbourhood approach is inspected for the VRPTW problem. The proposed approach is a four-phase approach. In the first phase, initial solutions are created using route construction heuristics. The route construction heuristics are cheapest insertion based heuristics, which have a lot of similarities with the studies of Solomon on insertion heuristics [1].

In order to generate the initial solutions, combinations of parametric values and seeding schemes or scenarios are used. The routes of an initial solution are built one at a time in a sequential fashion. During the building process of a route, if k customers are inserted, an intra-route improvement procedure called Or-Opt is used to reorder the customers in the partial route. This Or-Opt procedure depends in its work on the Or-Opt move, which relocates a chain of consecutive customers from a position on a route to another position on the same route. This Or-Opt move replaces three edges in the original route by three new edges without modifying the orientation of the route.

Then in the second phase, once an initial solution S is created, a route elimination procedure is applied to reduce the number of vehicles of the solution S until no more routes can be eliminated. In this route elimination procedure, a route R_e of the initial solution S is selected for elimination. At first, a simple insertion heuristic of the route elimination procedure is applied to re-insert customers from the route R_e into the other routes. If the simple insertion heuristic fails in eliminating the route R_e, then another insertion heuristic called IR-insert or intelligent reordering is applied.

The IR-insert re-inserts each customer c_i of the route R_e into a location of a route R_n that least increases an insertion cost value - Note that the feasibility checks of time windows are not accounted for here. Therefore, the violations of time window constraints might occur in the new route R_n that has now a customer c_i. In the case there are violations in terms of time window constraints, IR-insert locates the first violation of the time window constraints of a customer c_v. Then, two alternatives ways of re-ordering are used to try to reorder the customers of the route R_n and to

make the route R_n feasible. The first way is to try to serve the customers located before the customer c_v on the route R_n by re-inserting them after the customer c_v. The second way is to re-order the customers located before the customer c_v in the route R_n in order to minimize the arrival time at each customer.

If IR-insert has failed also, then an ejection chain technique is used. The ejection chain technique is based on the simple idea mentioned below. In order to re-insert a customer c_i of the route R_e selected for elimination into another route R_n, then a customer c_d from the route R_n has to be ejected and re-inserted into a route R'_n, where $R'_n \neq R_e$. Once c_i and c_d are inserted successfully into the routes R_n and R'_n respectively, an ejection chain is completed. If the insertion of the customer c_d is not successful in the route R'_n, then the removal of the customer c_d from the route R_n and the insertion of the customer c_i into the route R_n are cancelled. Also, the customers c_i and c_{d+1} are tried and so on. Of course, this ejection chain procedure is repeated for each customer c_i of the route R_e. Furthermore, in the ejection chain procedure, a breadth-first search strategy is used. In other words, all possible chains that require one ejection and two successful insertions as described above are examined. Then, all possible chains requiring two ejections and three insertions are examined and so on until stopping criteria are met.

Once the solutions with the smallest number of vehicles are created in the first and second phases, the third and fourth phases are used and repeated for a number of cycles in order to reduce only distance, without any regard to the reduction of the number of vehicles, using the route improvement procedures mentioned below. In each cycle of the third and fourth phases together, the cost function of the route improvement procedures is changed, within the forth phase, when it is necessary in order to escape local minima.

In the third phase, route improvement procedures are used to improve on the travelled distances of the solutions that have the smallest number of routes. Therefore, the best solution found so far can be updated, if an improved solution in terms of distance is found. Four types of route improvement procedures (ICROSS, IRP, IOPT and O-Opt) are used until no improvement can be found. The first two types ICROSS and IRP are used for inter-route improvements, whereas the other two IOPT and O-Opt are used for intra-route improvements. ICROSS is an extension of the CROSS-exchanges local search. The basic idea of the CROSS-exchanges local

search is first to swap two segments of consecutive customers seg_1 and seg_2 between two routes R_1 and R_2. So, the route R_1 will end up having the segment seg_2 and the route R_2 will embed the segment seg_1.

In the CROSS-exchanges local search and after exchanging the segments between the selected routes, the order of customers visited within the segments is preserved. However in ICROSS, the original and the reversed order of the customers within the segments are considered first and then the best in terms of reducing cost is always selected. The IRP 'insert related parallel' removes a set of related customers, in terms of how close they are in distance to each other, from the routes R_1 and R_2. Then, the removed customers are re-inserted into the partial routes R'_1 and R'_2 using a parallel cheapest insertion heuristic. Later, the customers in the new routes R'_1 and R'_2 are re-ordered using the IOPT operator mentioned below.

The IOPT is a generalization of the Or-Opt heuristic. IOPT has two major modifications of Or-Opt. Firstly, the IOPT relocates a selected segment of any length within a single route R_1. Secondly, it considers inverting as well as preserving the order of the customers within the selected segment. Therefore with IOPT, two kinds of routes might be created. One route R'_1 preserves the order in which the customers are visited within the selected segment, while the other route R^*_1 inverses the order of the customers. The best route of the two routes R'_1 and R^*_1 is selected to replace the original route R_1.

The O-Opt selects a number of customers k_{R_1}, say 4, from a route R_1 that visits, say 10 nodes including the depot. The O-Opt selects the k_{R_1} customers as far or dispersed as possible from each other to create the initial partial route R'_1, which has a similar shape to the original complete route R_1 but with the k_{R_1} customers only. In other words, the k_{R_1} customers are selected as the initial visits in the partial route R'_1. The order of the initial visits in the partial route R'_1 is same as that order of the same visits in the original route R_1. Then, the k_{R_1} customers are put in all orders to create $k_{R_1}!$ partial routes. Later, each feasible route of the created partial routes is rebuilt using a cheapest insertion heuristic that inserts the remaining customers that are sorted in an increasing order of the time window widths. Of course, the Or-Opt operator is used also to re-order the customers of a partial route after each time a number of customers are inserted.

In the fourth phase, the best solution found so far is exposed to further improve-

ment in terms of distance after using a post-optimisation phase, which modifies the cost function of the route improvement procedures of the third phase for a few iterations before switching back again to the original one. Thus, if none of the route improvement procedures is able to improve the best solution, then the cost function is changed in order to escape local minima and find new better solutions later.

2.7.6 Threshold Accepting algorithms

Threshold Accepting [28], a variant of simulated annealing, is applied to the VRPTW problem. The Threshold Accepting algorithm TA is a post-processor or post optimisation technique that is used to improve upon the distance results of solutions obtained from other algorithms such as a hybrid genetic algorithm [46] and a multi-start local search [17]. The TA post-processor uses the IOPT intra-route improvement heuristic and an inter-route improvement heuristic called GENICROSS.

The TA meta-heuristic is a modification of the well-known simulated annealing meta-heuristic SA. The TA meta-heuristic simplifies the SA procedure by leaving out the probabilistic element in accepting worse solutions. Instead, the TA procedure uses a deterministic threshold t. The TA procedure accepts a worse solution instead of the current solution, if the difference between the two values of the worse and the current solutions is smaller than or equal to $1 + t$.

The TA algorithm runs for a number of iterations and starts at a threshold value t equal to $t_{max} = 1$. At each iteration i, the routes of the current solution S_i are put in a random order. Also, all pairs of routes are exposed to improvement using the local search operators GENICROSS and IOPT. Accepting worse solutions in the local search operators is controlled by the current value of the threshold t. For that, if the relative difference in the lengths of the new and the current sets of selected arcs is less than or equal to 1+t, the modified pair of routes is accepted. The threshold t is reduced by $\Delta t = 0.025$ units in each iteration i until zero is reached. Now, if the stopping criteria or maximum number of iterations are not met, the threshold t is reset to equal t_{max} and the lowering of the threshold starts for the remaining iterations. Note that when the threshold t is reached zero, the search is repeated for four iterations before resetting the threshold t to equal t_{max}. Also, if the best solution S_{best} found so far is not updated for a number of iterations, the threshold t is reset to become equal to t_{max}.

During the local search, the current solution S_i might be updated. Therefore after using the local search operators IOPT and GENICROSS, if there is a solution S_i better than the best solution S_{best} found so far, then S_{best} becomes equal to S_i. Of course, the IOPT local search operator works as explained in Section 2.7.5. However, the GENICROSS local search operator is a large neighbourhood operator and is an extension of the local search operator of CROSS-exchanges. The difference between the GENICROSS and the CROSS-exchanges operators is that GENICROSS evaluates all possible exchanges of segments of consecutive customers between two routes. Because of the large neighbourhood structures of GENICORSS and IOPT, at most one move to a neighbouring solution is performed from each of the two operators GENICORSS and IOPT for each pair of routes.

2.8 Conclusions drawn from Literature

Previous research in the literature has focused extensively on solving the VRPTW problem using exact methods, heuristics and meta-heuristics. Exact methods, such as dynamic programming, Lagrange relaxation-based and column generation, are able to solve a small set but not all of the problem instances of Solomon [1] to optimality. Of course, this issue has encouraged many researchers to move to heuristics such as route construction and improvement heuristics in order to try to solve or to find very good quality solutions, possibly near optimal, for such problems instances.

Presently, meta-heuristics and hybrid approaches, such as simulated annealing, tabu search, evolutionary algorithms, ant colony systems, deterministic approaches and many others, are found to be much better in finding very good quality solutions than the route construction and improvement heuristics when applied alone on such problem instances. As shown in Table 2.10 where NV and TD refer to the number of vehicles and the total of travelled distances respectively, most of these approaches are tested on the classical set of 56 benchmark problem instances of Solomon [1], which have 100 customers each. However, there is less work on problem instances that have customers greater than 100 - like 200, 400, 800, 1000 or even more.

For that, one of the research questions to be exposed for further investigation in this thesis is whether the algorithms and the approaches used in this thesis are able to scale up when they are used to tackle problem instances greater than 100 like the

extended benchmark problem instances in [2] and [3] of Gehring and Homberger. Testing an algorithm or an approach on problem instances with 100 customers each is not a guarantee that the algorithm and the approach is doing well on larger problem instances. Consequently, using such problem instances alone may lead into having an algorithm or an approach that is not able to handle larger problem instances. In other words, it is easy and quite reasonable to forget whether that algorithm is able to scale up to tackle larger problem instances.

Furthermore, the metaheuristic approaches like tabu search, evolutionary algorithms, simulated annealing, ant colony systems...etc often claim that the success they have in finding very good quality solutions or improving the results is because of having a particular component, which makes the difference. However also, such metaheuristic approaches with the key component use the local searches a lot and they depend on them heavily. Therefore the question would be: if the key component of a metaheuristic approach and a local search are working together, then which one is doing the work.

For example in ant colony systems, the pheromone trails deposited on edges are used as key components in improving results and having very competitive solutions for problems like TSP and VRPTW. However such ant colony systems use the local search a lot as another basic component in addition to the pheromone trails. Often, local searches used in the literature are heuristic-based algorithms that are composed of components like move operators, acceptance criteria...etc and they cannot search completely the whole neighbourhood of a solution. Now if the local search is doing something with the pheromone trails, then what is the sort of local search should be used, what are the right structural properties of a VRPTW problem instance that should be part of the search engine of that local search and is it possible to update the search engine of a local search to even get better results and performance of the ant colony system used.

When using and updating the pheromone trails of the edges are switched off and the local search is left to work alone as part of a deterministic approach, then are the solutions optimized going to be still reasonably good. If letting the local search work alone to get reasonably good results is possible, then what are the sort of the structural properties of a VRPTW problem instance that could be used to improve its efficiency in finding even better quality solutions. If a local search has to use

the structural properties of a VRPTW problem instance, then are the structural properties to be used in a local search somehow related to any of the following features of problem instances such as how far in distance the customers are from the depot, how tight the time windows of the customers are or how large in amounts the demands of the customers are.

During the research and the investigation to the algorithms and approaches used in this thesis, the author is not just interested in reporting the average performance of an algorithm or an approach over a problem set but also in showing how are our algorithms and approaches doing on each of the problem instances that are members of that problem set. Note that the thesis will return back again to all the points mentioned above in later chapters. In conclusion of the previous research and what is mentioned above, meta-heuristics are so successful in getting competitive and quality solutions. Meta-heuristics are made up of simple components that make them a feasible idea for further research and investigations.

Table 2.10: Averages of the best solutions computed by different VRPTW algorithms on the problem group PG100 - check Table I.10 for more information.

		R1		C1		RC1		R2		C2		RC2	
Algorithm	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
RS [59]	1740	13.92	1211.22	10.56	846.88	13.75	1399.76	4.91	917.54	3.88	598.70	5.63	1055.61
AKRed [60]	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RTa [26]	2600 - 9800	12.25	1208.50	10.00	828.38	11.88	1377.39	2.91	961.72	3.00	589.86	3.38	1119.59
RVNSa [5]	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
RVNSb [5]	4950	11.92	1222.12	10.00	828.38	11.50	1389.58	2.73	975.12	3.00	589.86	3.25	1128.38
ES [2]	1200	12.41	1203.00	10.00	830.00	12.25	1357.00	2.91	955.00	3.00	592.00	3.25	1154.00
ES4 [2]	1200	12.41	1198.00	10.00	829.00	11.88	1356.00	2.82	947.00	3.00	590.00	3.25	1144.00
ES4C [2]	1200	12.41	1201.00	10.00	829.00	12.00	1356.00	2.91	945.00	3.00	590.00	3.25	1140.00
LC03 [33]	3600	12.08	1209.19	10.00	828.38	11.50	1389.22	2.73	963.62	3.00	589.86	3.25	1143.70
LCK05 [32]	3600	11.92	1214.20	10.00	828.38	11.50	1385.30	2.73	954.32	3.00	589.86	3.25	1129.43
BBB [46]	1800	11.92	1221.10	10.00	828.48	11.50	1389.89	2.73	975.43	3.00	589.93	3.25	1159.37
MACS-VRPTW [4]	1800	12.00	1217.73	10.00	828.38	11.63	1382.42	2.73	967.75	3.00	589.86	3.25	1129.19
TD-MACS [53]	1800	12.33	1199.91	10.00	828.38	11.88	1359.84	3.09	946.21	3.00	589.50	3.38	1124.63
HGA+TA [28]	2094	11.92	1216.58	10.00	828.38	11.50	1387.66	2.73	966.05	3.00	589.86	3.25	1143.12
LS+TA [28]	156	12.00	1214.69	10.00	828.38	11.50	1389.20	2.73	960.44	3.00	589.86	3.25	1124.14
LS+HGA+TA [28]	2250	11.92	1215.62	10.00	828.38	11.50	1385.17	2.73	956.35	3.00	589.86	3.25	1123.84
ALV [61]	3600	13.25	1183.38	10.00	828.38	12.88	1341.67	5.55	899.90	3.00	589.86	6.50	1015.90
LL [62]	140 - 6500	12.08	1215.14	10.00	828.38	11.75	1385.47	2.91	953.55	3.00	589.86	3.25	1142.48
BRDU1 [63]	120	12.00	1248.61	10.00	828.38	11.50	1421.20	2.73	981.60	3.00	589.86	3.25	1171.73
BRDU1 [63]	546	12.00	1220.14	10.00	828.38	11.50	1397.44	2.73	977.57	3.00	589.86	3.25	1140.06
MSLS1 [17]	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.86	3.25	1152.37
MSLS+TA1 [17]	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37
HG05b [64]	1050	12.08	1211.67	10.00	828.45	11.50	1395.93	2.82	950.72	3.00	589.96	3.25	1135.09
RTb [26]	2600 - 9800	12.67	1204.20	10.00	830.25	12.25	1376.26	3.36	981.21	3.00	590.80	4.00	1157.98
TB [31]	11264 20232	12.17	1209.35	10.00	828.38	11.50	1389.22	2.82	980.27	3.00	589.86	3.38	1117.44
HGA+EA [27]	903	12.33	1216.08	10.00	828.75	12.13	1366.16	3.00	974.68	3.00	591.31	3.38	1165.38
SA+LNS [29]	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HM4 [3]	3240	12.08	1208.14	10.00	828.50	11.50	1389.65	2.73	965.46	3.00	589.88	3.25	1126.22
HM4C [3]	3240	12.00	1217.57	10.00	828.63	11.50	1395.13	2.73	961.29	3.00	590.33	3.25	1139.37
HGES1 [65]	522-990	11.92	1228.06	10.00	828.38	11.63	1392.57	2.73	969.95	3.00	589.86	3.25	1144.43
HGES2 [65]	570-1407	12.00	1226.38	10.00	828.38	11.50	1406.58	2.73	1033.58	3.00	589.86	3.25	1175.98
RGP02 [66]	10998	12.08	1210.21	10.00	828.38	11.63	1382.78	3.00	941.08	3.00	589.86	3.38	1105.22
MSSA [30]	660	13.33	1186.94	10.00	828.38	13.25	1362.44	5.36	878.79	3.00	589.86	6.13	1004.59
MSLS2 [17]	132	12.00	1222.55	10.00	828.38	11.50	1400.91	2.73	968.77	3.00	589.86	3.25	1139.51
MSLS+TA2 [17]	162	12.00	1212.89	10.00	828.38	11.50	1389.20	2.73	960.44	3.00	589.86	3.25	1124.14
VGA1 [7]	13 - 87	13.25	1179.95	10.00	828.38	13.00	1343.65	5.36	878.41	3.00	589.86	6.25	1004.21
VGA2 [7]	13-109	13.25	1180.20	10.00	828.38	12.88	1340.53	5.27	878.46	3.00	589.86	6.38	1004.57
LSH [67]	1200	12.17	1249.57	10.00	830.06	11.88	1412.87	2.82	1016.58	3.00	591.03	3.25	1204.87
TH [25]	100 - 3749	12.33	1234.83	10.00	830.89	12.00	1409.38	3.00	988.18	3.00	640.75	3.38	1190.25
PB [41]	600 - 2460	12.58	1296.80	10.00	838.01	12.13	1446.20	3.00	1117.70	3.00	589.93	3.38	1360.57
TLOL [68]	N/A	12.92	1205.05	10.00	841.96	12.63	1392.36	5.09	929.67	3.00	611.29	5.88	1080.07
DBF [69]	N/A	14.17	1214.86	10.00	829.77	14.25	1385.12	5.27	930.18	3.25	604.84	6.25	1099.96
ILS [70]	15000	12.00	1213.68	10.00	828.38	11.50	1394.43	2.73	971.41	3.00	589.86	3.25	1146.12
AMLS [70]	15000	12.00	1216.03	10.00	828.38	11.50	1401.19	2.73	960.29	3.00	589.88	3.25	1129.44
CR1 [71]	N/A	12.42	1289.95	10.00	885.86	12.38	1455.82	2.91	1135.14	3.00	658.88	3.38	1361.14
CR2 [24]	144	12.50	1308.82	10.00	909.80	12.38	1473.90	2.91	1166.42	3.00	666.30	3.38	1393.70
CR3 [72]	3624	12.17	1204.19	10.00	828.38	11.88	1397.44	2.73	986.32	3.00	591.42	3.25	1229.54
CLM [73]	N/A	12.08	1210.14	10.00	828.38	11.50	1389.78	2.73	969.57	3.00	589.86	3.25	1134.52
Best [Table I.1]	N/A	11.92	1209.89	10.00	828.38	11.50	1384.16	2.73	951.66	3.00	589.86	3.25	1119.17

Chapter 3

Presentations of techniques and results

This chapter describes critically how some very-well known ACO techniques (like MACS-VRPTW [4] Savings-based AS [57], ACS+NN [6], ACS+I1 [6]...etc) and their local searches and results are presented in the VRPTW literature and how they should be introduced. In Sections 3.1, 3.2 and 3.3, a critical review for such techniques and results are explained in detail in addition to the experiences and the problems, faced at the time of corresponding with the research team of the very-well known MACS-VRPTW system. Later in Sections 3.4 and 3.5, this chapter explains how such techniques should be described and how their results should be reported. At the end of Section 3.5, a set of recommendations in reporting such results are suggested.

3.1 How algorithms are described in terms of the level of detail?

When it comes to describing algorithms, it is known that many researchers do a very bad job of it. Some researchers would claim that there is not an enough space available for writing a conference paper or the number of pages that is required to write a paper is limited to 8 pages for example. Other researchers might claim that they want to keep for themselves some important aspects of their applications secret for copyright reasons. Others might say that they did not notice that the

description was insufficient or that they did miss some of essential details because they thought they are very highly familiar with their techniques. Such conference papers and journal articles should convey sufficient details about such techniques to allow other researchers to recreate their results with decent accuracy.

An example of what is described in the previous paragraphs is the description of the MACS-VRPTW system in [4]. When looking at its cycle phase, there are a number of issues that need to be clarified. For example, it can be stopped using a stopping or a halting criterion, which is not described and could be any combination of the following - the number of iterations allowed, the number of solutions created, the amount of CPU time elapsed and/or the number of edges evaluated. Also, the cycle phase of MACS-VRPTW does not show exactly what and how to keep activating the search in both colonies ACS-VEI and ACS-TIME and as a consequence there is confusion about when the role of the initialisation phase and the role of the cycle phase in each colony should come into play. Furthermore, the description of the cycle phase is not clear about how it receives newly improved solutions from the colonies involved and in what sense and order it updates the best global solution.

In the initialisation phase of each colony whether ACS-VEI or ACS-TIME, there is no clear indication how to initialise the pheromone trails. The cycle phase of each colony has a step that is unclear about how it sends and notifies MACS-VRPTW and the other colony about any new improved solution found so far in the colony involved. Also, the stopping criterion of a colony is not told what is it and therefore it could be any combination of those described in the previous paragraph. In addition, the constructive phase or the routing builder of an ant of a colony is described in an unclear way in the sense that it does not show how and when it uses either the exploitation part or the exploration part of the probabilistic state transition rule. Also, the probabilistic state transition rule does not show in what order it uses a set of feasible nodes still to be visited and when it comes to the insertion procedure and the local search of CROSS-exchanges, they are described in a marginal way as if their contribution is minimal.

3.1.1 Are the pheromone trails doing the work?

In the literature, some ACO techniques (like MACS-VRPTW [4], Savings-based AS [57], ACS+NN [6], ACS+I1 [6]...etc) show the importance of the pheromone

component over other components like the local search for example. This gives the impression that the pheromone component is the one that is doing the useful work for getting good performance and results. In the abstract of MACS-VRPTW [4] for example, the paper says that cooperation between colonies is performed by exchanging information through pheromone updating and does not mention in any way the importance of the local search used or any other ingredients as it is done in the abstract of this PhD thesis.

Also as an indication to what it is mentioned above, the article body of MACS-VRPTW talks a lot about the pheromone trails and their importance and it describes all the different components that relate to those pheromone trails in a way that ignores and misses out other ingredients, such as the local search, as if they are not doing anything that is significantly important. For example, the constructive phase describes in a lengthy and an imprecise way how the ants build their solutions using the probabilistic-state transition rule with its exploitation and exploration parts (e.g. such parts should use heuristics that include those pheromone trails) and how they diminish locally the pheromone trails using a pheromone local update rule. Furthermore, the local search of CROSS-exchanges is mentioned in that constructive phase in a very marginal way that declares its name only.

Another example, the pheromone global updating that increases the pheromone trails of the visited arcs of the best global solutions is talked about with emphasis and this is indicated from the fact that the two pheromone global update rules of the colony VEI are mentioned at the end of the cycle phase. Also, the same is true for the pheromone global update rule of the colony TIME.

3.1.2 How local searches are described?

Local searches in some ACO algorithms, as MACS-VRPTW in [4], Savings-based AS [57], ACS+NN [6], ACS+I1 [6]...etc, are not given the importance factor they should be given. Therefore in such ACO algorithms like MACS-VRPTW, there is the impression that move generation methods or the move operators are the only ones that make the local search work and without mentioning other kinds of essential ingredients as those mentioned below.

For example, the way in which the local search works and the order in which the customers or the tours are used should be regarded as important as knowing about

the move operators themselves that create the neighbouring solutions of a currently used solution. Once there are move operators to be used in a local search, then their kinds, what they do and whether they are intra- and inter-route improvement operators are also the kinds of details that should be mentioned in a clear way without any ambiguity.

Another important issue is that whether there is any kind of limitation for the search space of the move operators by imposing for example a candidate list of n nearest nodes for each visited node in a tour and whether there are any particular acceptance criteria that should be followed in accepting a neighbouring solution to be as the new solution to start improving upon. For instance, are these acceptance criteria somehow related to the strategy of accepting the best solution out of a number of neighbouring solutions or to the strategy of accepting the first solution created so far in the neighborhood. Of course at last and not least, the stopping criteria, used to halt the search, should be indicated by saying, for example, what is the number of iterations should be used, what is the amount of CPU time should be elapsed, what is the number of edges should be evaluated and/or whether the local search should keep searching until no improvement can be achieved.

3.2 How the results are reported?

In the VRPTW literature, researchers report usually the averaged results in terms of the number of vehicles and the total of travelled distances on the six problem sets of Solomon [1] without showing any more information about how their methods work in terms of performance on the problem instances of such problem sets. Also, other details, such as the specifications of the used PC machines, the development environments and the number of made runs and the averaged CPU time it takes an algorithm to reach to those kinds of results, are mentioned.

For example in the system MACS-VRPTW in [4], the researchers mention about the details of the PC machine used (a Sun UltraSparc 1 167 MHZ) and that the experiments are done by running MACS-VRPTW on each problem instance for three runs. The researchers of MACS-VRPTW mention in a first table the performance and the results of the system on the six problem sets of Solomon [1] during five different amounts of CPU time, which are 100, 300, 600, 1200 and 1800 seconds.

In the same table, the performances and the results of other different VRPTW algorithms, run at other different amounts of CPU time (such as 450, 900, 1300, 1382, 2700, 2296, 2900, 3600, 6887 and 13774), are described also. In MACS-VRPTW, the researchers say that the computational times of different methods cannot be directly compared for different reasons like having computers with different features such as different PC processor types, PC speeds, RAM capacities...etc and that some methods were designed to solve harder problem instances and to be faster due to being implemented in development environments that are considered as fast in comparison to others.

Here in the first table, the results reflect the performance of the system MACS-VRPTW on the six problem sets during the five different amounts of CPU time described above but there are a number of issues that are not clear with it. For instance, MACS-VRPTW in [4] does not tell anything about how much precise, robust, reliable and consistent the algorithm is in getting those kinds of results. What's more, it is not clear if the researchers of MACS-VRPTW have picked the best result of the three runs of each problem instance in this first table or if the three runs' results are averaged over three. Also, the table does not show how MACS-VRPTW does behave on each problem instance of a problem set. In addition in that table, it is not clear why some researchers run their algorithms for several lengths of CPU time that are different from other lengths of CPU time done by other researchers. Then in a second table, the researchers of MACS-VRPTW report, on the six problem sets of Solomon, the averaged results of the best solutions found in all the experiments done.

In this second table, there is ambiguity about whether the results of MACS-VPRTW are produced after doing very long runs of CPU time that are different from those described in the first table or after doing many experiments that are not known about. The reason behind saying this is that the averaged results of the MACS-VPRW system on the six problem sets in both tables (first and second) are different from each other. Another reason is that some of the methods listed in this second table are not listed in the same way with their results in the first table. For example, the methods of TB, CR and PB of the second table are not mentioned in the first table. Furthermore, the second table does not indicate that the best of three results is picked up after computing the three results for each problem instance

in the first table for example. Finally in a third table, the researchers of MACS-VRPTW show the best results obtained by MACS-VRPTW only on certain specific problem instances and there are no more other tables of results available.

3.3 What sorts of experiences and problems faced?

The ambiguity of the description and the results of the MACS-VRPTW system in [4] has been one of the motives behind the decision to ask the researchers in [4] for the original software in order to study and investigate it further. For example, MACS-VRPTW does not mention clearly in [4] the fact the ants might choose sometimes a little bit longer arcs. The ants are expressed as agents that choose always shorter paths or arcs with higher amounts of pheromone - even if the amounts of time (could be spent) in such arcs are going to be a little bit bigger than in longer ones. Of course, the argument of choosing always shorter arcs with higher amounts of pheromone is imperfect and not always right. Sometimes, the physically longer arc might be the one that is best to be chosen.

For instance, the quickest way from one side of Edinburgh to the other is sometimes to use the city bypass, even though it is much longer than going through the city centre. Of course after asking the researchers of MACS-VRPTW about having the original software, the answer was no and this maybe due to copyright issues. Also, other motives have been to see whether MACS-VRPTW could scale up in solving larger problem instances with 200, 400, 600, 800 and 1000 customers and to check the effects of using different kinds of route construction heuristics and improvement heuristics.

How much the system MACS-VRPTW is precise, robust, reliable and consistent in getting the published performance and results is another motive behind the decision to study and to investigate that system. Furthermore, the article of MACS-VRPTW shows us how the system does behave on the problem sets without showing us how it behaves on each of the 56 problem instances of Solomon [1]. In [4], there is not any indication which problem instances MACS-VRPTW does behave very well or terribly bad on in terms of performance. The final motive is to check whether the usage and the update of the pheromone trails are the main reasons behind the success of MACS-VRPTW in getting that good performance and results. Or maybe,

there are other kinds of ingredients that are doing their bits also. Maybe, such ingredients rather than the usage and the update of the pheromone trails are the ones behind the success of MACS-VRPTW.

Afterwards, the motives described above were the trigger point to develop an approach of a multiple ant colony system called DACS in [55] [56] from the abstract ideas described in the paper of MACS-VRPTW in order to study and to investigate what are mentioned above from motives. Unfortunately, the performance and the results of the developed system were very poor and there were several doubts around a number of issues as mentioned below.

For instance, the way the pheromone trails are initialised and the management of the evaporation process of the pheromone trails are one of those doubts that could answer why the system DACS at that time was having such poor performance and results. Unfortunately, neither the initialisation of the pheromone trails nor the management of the evaporation process are the reasons behind that poor behaviour of DACS. Another issue is to think about whether the pheromone trail re-initialisations might be the ones that should be included in the implementation of a DACS system. The pheromone trail re-initialisations have improved the performance and results of the DACS system but such improvement is not good enough.

On the other hand, the confusion that has resulted from the description of the cycle phases of MACS-VRPTW and its colonies and how they manage their roles has led into reconfiguring such cycle phases on a number of occasions and has led into thinking MACS-VRPTW might be a multi-threaded one. Therefore, after reconfiguring the cycle phases of the DACS system and its colonies, that does not seem to make any difference and the same is true when the colonies are converted to threaded colonies that have the ability of yielding the process of searching to each other.

Also at some stage, another solution to improving the performance and the results of the DACS system was to think about changing the kind of the individual move operator that is used and therefore a number of individual move operators are tried and that does not seem to have helped the performance and the results of that DACS system to improve. This kind of thought is the result of how poorly local searches are described in the literature and it shows that local searches are not explained very well as they should be in proceeding papers and journal articles and

the most important ingredient talked about always is the kind of move operators used. It is believed that the other remaining ingredients of the local searches, as talked about in Section 3.1.2, can be explained in detail and without any cost in terms of space because they do not take that much space to be explained but researchers, instead, seem to find it easier to give the credit to other ingredients, like the pheromone trails for example as in MACS-VRPTW.

Likewise, there are many other issues that are not explained in detail in MACS-VRPTW such as the usage of duplicated depots in the probabilistic-state transition rule and the pheromone updating locally and globally especially in relation to whether that pheromone updating is done for the two edge sides of an arc or to one edge side only. Additionally, the global pheromone diminishing of the unvisited edges of the best global solutions is not explained in MACS-VRPTW as it is the case in other ACO techniques. Also, the way the τ_o term is initialised in each colony and whether such term is different from a colony to another are among the other issues that puzzle researchers when reading about MACS-VRPTW in [4].

In addition, issues like the way the ants search for solutions (whether in parallel or in sequential) and the way the initial solutions are created in the initialisation steps of the DACS system and its colonies are studied and investigated but there are no indications that these kinds of issues have any significant change on the performance. Furthermore, issues like the usage of different kinds of candidate lists in the routing builder of an ant has not led into making the performance of the DACS system any better.

As a result of what is described above, a number of questions have been asked by the author to the researchers of MACS-VRPTW. Sometime later, abstract answers were received by the author to mainly two things mentioned below. Firstly, the CPU time is used as the stopping criterion of the whole system of MACS-VRPTW. Secondly, the move operators are about either swapping two customers or relocating a customer after another. The local search of the move operators works in the order of the built tours of a solution starting from the first tour built and ending at the last tour built. When building new solutions using the move operators mentioned above, the local search uses a candidate list of 20 nearest nodes for each visited customer and tries with each visited node and each nearest node to build three solutions. The best of three solutions is chosen. The local search works until no improvement can

be achieved.

Of course, the performance of the developed DACS system with the new abstract information received has improved but such modified system does not even produce the comparable performance that is wanted and expected from MACS-VRPTW [4].

3.4 How the algorithms should be described better?

Algorithms should be described better by mentioning much more details about the system involved and not to hide information about the components that could be essential. Therefore in the most recent developed DACS systems, components that are regarded as key ingredients like the push-forward and push-backward strategy PFPBS, the hybrid local search HLS and the 2-Opt move are not hidden or missed out. Consequently, the usage of the pheromone components and the triple move local search without such key ingredients is definitely going to lead to bad performance.

Additionally, the roles of the components should be indicated clearly by knowing when such roles come into play. For example in the DACS system in Figure 4.2, the cycle phase of the coordinator and how it works and when it stops are explained clearly and there is no confusion about the waiting role of the coordinator DACS for newly improved best global solutions as in MACS-VRPTW. Therefore, it is clear when the initialisation and the cycle phases of the colonies VMIN and DMIN are called. As a result of that, there is no ambiguity when the role of each colony (VMIN or DMIN) comes into play and when the cycle of each colony is activated. Also, the cycle phase of the coordinator DACS shows how to keep activating both colonies.

For the both colonies of VMIN and DMIN of the DACS system as in Figures 4.3 and 4.4, it is clear how the pheromone trail of each edge is initialised and how the cycle of each colony updates the best global solution and when a colony notifies the coordinator and the other colony about any solution update if there is any thing like that. The cycle of each colony stops searching when it is supposed to and yields the process of searching for the other colony when the time comes for that moment. Other details such as doing arc or one side pheromone updating (globally and locally) and doing global pheromone dimensioning are described also. For the routing builder of the ants of the DACS system as in Figure 4.5, the components

of the routing builder and how they work together are explained in a much clearer way.

For example, the usage of candidate lists within the probabilistic state-transition rule and when the exploitation and the exploration parts should be applied are mentioned clearly. The insertion procedure in Figure 4.6 is explained in detail and uses candidate lists and there is a clear declaration for how the local search in Figures 4.7 and 4.8 does work with its move operators and candidate lists and when it stops.

3.5 How the results should be reported?

Reporting the results of a system should lead into telling a great deal about how the system is performing not just on the problem sets involved after a few runs of CPU time but also on each problem instance and in VRPTW research this issue is not taken into consideration in the sense that VRPTW researchers, as described in Section 3.2, do not give enough information about that. Researchers might want to have algorithms that are able to bring a very good performance on a whole problem set without seeing what the algorithms are doing in terms of performance on each of the problem instances of that set but such thing has a number of problems attached to it and therefore the following issues should be regarded seriously when trying to tackle sets of problem instances of a combinatorial optimization problem like VRPTW.

Firstly, the problem instances of a problem set, like C1 for example, are different from each other and each problem instance has its own features that are different from the general features that unifies the problem instances in that problem set. For instance in the problem set C1 of Solomon [1], there are ten clusters with varying numbers of customers in them and this is regarded as the general feature that unifies the problem instances of C1. However, when it comes to the problem instances in C1, each problem instance is distinguished, according to the features it has, from other problem instances in the same problem set. In C101, just one customer is ready at any time whereas C105 has one or two customers per cluster that are ready at any time and if C109 is used, there are up to six customers that are ready at each time.

Secondly, averaging the results over a problem set after a few runs does not show how an algorithm is behaving on each problem instance. Because of the features that might distinguish each problem instance, an algorithm might be doing well on certain problem instances but it might be as well doing terrible on others. For example in Section 4.7.3, once the hybrid local search HLS is added as in the system DACS+HLS, the averaged NV and TD result of 12.51 and 1224.75 on the problem set R1, as in Table 4.18, has improved in terms of the number of vehicles and has deteriorated in terms of the total of traveled distances in comparison to that of the system DACS 03 - NV = 12.60 and TD = 1221.49. But in Table 4.18, the averaged NV and TD result of DACS+HLS on R1 does not tell us that the NV result 17.30 of DACS+HLS on the problem instance R102 has deteriorated on average as in Table 4.19, for example, as compared with the NV result 17.20 of DACS 03 in Table 4.15.

Likewise on the problem set R2 in Table 4.18, the addition of HLS into DACS 03 has led into improving the averaged NV result from 3.04 to 2.92 and deteriorating the averaged TD result from 958.53 to 970.87 but when it comes to a problem instance, like R210, a different story is seen. On R210 in Tables 4.19 and 4.15, the NV results obtained, on average, are the same in both systems DACS+HLS and DACS 03 whereas the TD result 975.99 of DACS+HLS on R210 indicates that there is an improvement on average as compared with the TD result 984.59 of DACS 03. As a consequence, it is a good idea to show how the system is behaving also on each problem instance.

Thirdly, running a system for three runs might be good for an end user who would like to choose the best solution out of three runs but to other end users three (maybe) is rather few and it does not tell that much about the system itself and how it behaves if a large number of runs, like thirty for example, is used. For example in the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt in Sections 4.7.1, 4.7.3 and 4.7.4, if such systems are run for thirty runs, quality data results are gained in a way that is comparable and competitive with those obtained by the state-of-the-art approaches. Also, running a system for many runs, like thirty, would tell a lot about the robustness and the consistency of the system in getting good quality performance.

Fourthly, reporting the averages and standard deviations on problem instances

might show the robustness and the consistency of a system in some occasions but such statistical measures might be informal on other occasions. Therefore on occasions where the optimal solution of a problem instance is known, such statistical measures might not tell the end user that the system in 99.9% is able to bring the optimal solution. For example for the problem instance C105, the system DACS 03 in Section 4.7.1 is able after 1800 seconds to bring in 29 runs out of thirty runs the NV and TD values of 10 and 828.94 and this good performance is not reflected in the average and standard deviation values of NV and TD. The average NV and TD values on C105 are 10 and 830.05 and the standard deviation values are 0 for NV and 6.10 for TD. One way to overcome this issue might be to report about the frequency of getting the optimal result of NV and TD on a problem instance.

Fifthly, some people are happy to spend huge amounts of computer time to try to get the best possible result, for example. But in practice many people just want something quick, and they would prefer to only do a few runs. Now in order to help in knowing how many runs an end user might need to run an algorithm, the first thing that comes up to the mind is whether or not the runs, to be done, are close in performance to one another. As a consequence, if there is a way in calculating how much precise an algorithm in getting its performance, then the end user may be afterwards able to decide on the number of runs might be needed. After investigation, there is a way to measure the precision of an algorithm in getting its performance.

In statistics, the precision measure is called the coefficient of variance (or coefficient of variation) CV,which is defined as - (standard deviation/average) * 100. This measure is a commonly-used measure of the consistency of a set of data points or results and here CV is expressed as a percentage. If the CV is small, it is because the variation is small compared to the average, so that most results are close to that average. So, the closer the CV value is to zero, the more the algorithm is considered as precise in getting its performance. The greater the CV value, the less precise the algorithm is. Then, once the precision of an algorithm is known on a particular problem data or a group of problem instances, the number of runs to be used is going to be determined easily from the end user's point view.

Sixthly, it is a very good idea to report about the performance of a system in the best, the average and the worst case scenarios and to show the end user different

scenarios of the system's performance in comparison to scenarios of other versions of the same system. For example, the best, average and worst case performance scenarios of the system DACS+HLS+2-Opt in Section 4.7.4 are better than those scenarios in Section 4.7.1 of the system DACS 03.

Seventhly, running a system for long CPU times is not enough to tell about how good a system is and therefore it is important for some end users to have a system that is able to bring high quality results in very short amounts of CPU times as well. For example, an end user might prefer and require a system like DACS+HLS in Section 4.7.3 or DACS+HLS+2-Opt over a system like DACS 03 only because the results and the performances of the former systems are simply better in quality and more trust worthy.

Eighthly, comparing directly on the basis of CPU time between the different VRPTW algorithms cannot be done because of the differences that such algorithms have when it comes to the hardware and software used. Algorithms are usually run using different PC speeds, RAM capacities, development environments...etc and therefore it would be unfair to do that kind of comparison directly. However, the comparisons can be done indirectly only and using the percentage of deviation in Equation 3.1 between any two algorithms A and B in order to have a better understanding of how our systems are doing in terms of performance and whether the performance of a particular system of ours is really horrible or not from the perspective of an end user.

$$\%\text{ of deviation to } Algor_A = \left(\frac{ResultOfAlgor_B - ResultOfAlgor_A}{ResultOfAlgor_A}\right) \times 100 \quad (3.1)$$

So, if the percentage of deviation to the algorithm A from the algorithm B is in the minus, it means that B is getting an improved result in terms of performance. If the percentage of deviation is a positive value, then the result of the algorithm B and therefore its performance are deteriorating compared to those of the algorithm A. Now, when the percentage of deviation is equal to zero, the results of both algorithms A and B match each other in terms of performance.

For example, when the local search of triple moves (swapping and relocating) is included in Section 4.6.1 as in the system DACS 02, the performance has improved. Also despite the inclusion of the local search of triple moves, there is a realization from indirect comparisons with MACS-VRPTW that there is a serious problem

with DACS 02 and this is according to percentages of deviations in Table 4.6 that indicate that MACS-VRPTW is much better in terms of performance. Therefore, the push-forward and push-backward strategy PFPBS, which has improved things dramatically as in the system DACS 03 in Section 4.7.1, is included to the put the performance in a better position.

Ninthly, there is not a standard way or a generally agreed method of comparisons between the different VRPTW methods because such kinds of comparisons if there are any depend really on what an end user wants. End users might concentrate on differing combinations of speed, chances of producing near optimal results, sensitivity to unexpected changes in a problem instance such as more or fewer customers, vehicle breakdowns, changes in the servicing hours of some of the customers, changes in driver hours and so on or other kinds of such criteria. End users will all have their own criteria, and results on a couple of comparison methods would not be helpful except to academics competing with each other and using the same features of hardware and software in their experiments such as same PC types, PC speeds, RAM capacities, development environments...etc.

For example, a supermarket might want an algorithm that is doing a very fast computational effort in order to get any results using a PC LAN of 4 CPU processors (like in the algorithms of Ghering and Hambourger) or within one hour on a 3.0 GHz Pentium 4 with 1GB memory in a way that beats another commercial algorithm in the sense of using the same or fewer vehicles in 75% of the time on average and with a smaller distance if the number of vehicles is the same. On the other hand, another supermarket might want any solution that costs no more than a few pounds and can be found within one CPU day and could be modified to add more customers in a region without any more cost than a number of additional pounds X per customer and so on.

Tenthly, is the number of edge evaluations the right thing to measure with in order to make the comparisons fair between the different VRPTW algorithms? Now, some of these VRPTW algorithms are developed as non-deterministic algorithms and the number of edge evaluations on average is not the same. Maybe, it is needed to check how many edge evaluations on average it is expected to have after a limited amount of CPU time. Then, the worst number of edge evaluations of a number of runs, after a particular limited amount of CPU time, might be chosen to be as a

limit to stop the search in an algorithm. But does that mean that the CPU time as a measure is not needed to be mentioned near the other features of hardware and software used in the experiments?

Well, the answer is not because some end users might want at the end of the day algorithms that are fast enough and bring them high quality results in very short amounts of CPU time. Therefore when it comes to the number of edge evaluations as a measure, the different VRPTW algorithms are still having their differences in terms of how they are skillfully implemented and whether the algorithms were designed efficiently to tackle large problem instances and on that basis the performances cannot still be fairly compared. As an example, let us imagine that there are two researchers who are using the same kind of hardware and software features in their experiments but both of them are different from each other in terms of development skills. In this case, a researcher might bring his good results after 1,000,000 edge evaluations and in CPU time equal to 100 seconds and another researcher might bring on average the same results after the same number of edge evaluations but in CPU time equal to 1800 seconds for example and in this case some end users might still prefer the algorithm of the first researcher simply because it is faster and able to bring high quality results in very short amounts of CPU time.

Finally, it is very important when reporting about results to check the significance of key ingredient components by using and then not using them. The significance of such key ingredient components can be checked by using statistical methods like the Student's t-test and the signed rank Wilcoxon test and charts such as histograms and lines. For example, switching off the usage of the local search of triple moves, the strategy of push-forward and push-backward PFPBS and the hybrid local search HLS is going to lead to performance that is worse than the performance resulting from switching off the usage and the update of the pheromone trails.

Now on the basis of what is described above, the following guidelines are highly recommended when reporting about results.

$R1-$ The features of the hardware and software used in the experimentation of an algorithm should be mentioned clearly and those features are the processor's type and speed, the RAM capacity, the operating system used, the development environment...etc.

R2– The number of runs used in the experimentation of an algorithm should be determined without any ambiguity. For example, the system DACS+HLS+2-Opt in Section 4.7.4 is very precise (the coefficient of variance CV is less than 1.60%) on each problem set and therefore can bring very good quality results on each problem instance in three runs only. For that, three runs should be enough for an end user to gain the good quality results needed. However in order to get competitive results with those of the state-of-the-art approaches, the end user might need to do lots of runs like thirty for instance.

R3– An algorithm should be tested in a way that shows how much it is precise, in terms of the coefficient of variance CV ([standard deviation/average] * 100), during the running on each problem instance and therefore each problem set in order to see the robustness, the reliability and the consistency of that algorithm. Therefore, reporting statistical measures like average NV and TD results of the many runs done and standard deviations can help in determining such precision, robustness, reliability and consistency but in some occasions where the optimal solution of a problem instance is known, then it is better to report in this case the frequency of the NV and TD result of the best solution found so far for each problem instance because averages and standard deviations might not tell about the true performance of a system.

R4– An algorithm should be tested for different durations of CPU time (long and short) during each run. Some real end users care about having a high quality performance in a very short amount of time as mush as they care about the quality of that performance in the long term. Moreover, an end user who wants to apply an algorithm might want to know for how long to run it. With a non-deterministic approach like ACO that make random choices sometimes, the longer these techniques are run in terms of CPU time the better the results are going to be. If the results regularly get better with more time, then increase the time allowance even further. Of course, it may happen that the results very occasionally get better when more time is allowed. In such cases, there has to be some trade-off between the huge amounts of time on the one hand and on the other hand being content with a good rather than an optimal result. For that, such techniques can be measured (in terms of being content) with how

much precise they are in getting those kinds of good quality results.

R5– An algorithm should be shown how it behaves in terms of performance in the best case, average and worst case scenarios. In other words, it would also be useful to say a bit on each problem instance and each problem set more about the worst and best cases in addition to the average case.

R6– In order to compare fairly the performances of a number of different VRPTW algorithms for each problem instance, the features of the hardware and software used in the experimentation process should be the same. For comparison reasons, the percentages of deviations, calculated using Equation 3.1, between such algorithms on each problem instance and set should be reported and shown in tables. Otherwise, it might be a good idea to show how many edge evaluations on average can be done during a number of runs that last up to a certain amount of CPU time but this issue needs to be studied and investigated further in the future by researchers by trying a number of different VRPTW algorithms having many differences in terms of the features of the hardware and software used and checking whether that show any fair comparison. Now in addition to the random choices or the non-deterministic nature of some algorithms, it has to be noted that it is expected that some of the issues that differentiate the VRPTW algorithms are still going to remain key factors in determining how skillfully such algorithms are implemented. Those key factors are such as how skillful a developer is, how fast an algorithm is in finding a high quality solution in a very short amount of CPU time and how an algorithm is efficiently designed to tackle large problem instances.

R7– An algorithm should be fairly showing the effects of using and not using key ingredients in order to know which key ingredient is doing the real work and this could be discovered by using statistical measures such as the Student's t-test, the signed rank Wilcoxon test...etc. So, is it the usage and the update of the pheromone trails or is it simply the local search used or maybe both of them cannot do anything in a non-deterministic algorithm without the usage of the push-forward and push-backward strategy PFPBS. For example, if there is a DACS system that switches off the usage and the update of the pheromone trails and another DACS system that misses out the local search, then the

performance of the second system without the local search is going to be far worse than the performance of the first system without the pheromone trails' usage and update. Of course, this is an indication that the local search has a great deal in the DACS system's performance.

Chapter 4

Multiple Ant Colonies

The main interest in this chapter is the investigation of variants of multiple ant colony approaches in order to see which ingredients are vital. Are they the pheromone trails, the local searches or other kinds of ingredients? Therefore, the aim or the message to transfer in this chapter is to show what is mentioned below.

> Switching off the work of the local searches will make the performance of a system far worse than switching off the usage of other well known ingredients (like the pheromone trails for instance).

Multiple ant colony approaches are extensions of ant systems [49] [54] and ant colony systems [50] [51] [74] [52] [75] mentioned in Section 2.7.4 and they have come mainly to tackle multiple objective problems like VRPTW. A researcher might ask what if a meta-heuristic technique is so successful because of using local searches and some particular components that distinguishes between attractive and unattractive edges, then which ingredient is going to be behind that success.

In the case of multiple ant colony approaches, the original starting point of finding which ingredients are vital is an attempt to re-create a double ant colony system "DACS" from the abstract ideas of the system MACS-VRPTW of Gambardella as mentioned in [4]. MACS-VRPTW has two colonies and one of the two colonies reduces the number of vehicles while the other minimises the total of travelled distances. Although MACS-VRPTW has been surpassed in some ways by Braysy's reactive variable neighbourhood search [8] [5], it does produce very good results. However, it turns out that it is not possible to obtain those similar good results simply by following the published description due to the effects of some missing

details. Therefore, none of the multiple ant colony systems that are studied and investigated in this thesis is in any way the MACS-VRPTW system itself but however such systems have a lot of similarities with each other.

Consequently, other variants of DACS systems are tried in order to get the performance as good as MACS-VRPTW but such variants are not good enough. For that after talking directly to the researchers of MACS-VRPTW, it has become apparent that the published description [4] lacks certain details about the local search that turn about to be very important. Later even after the improvement in performance because of including those details in the local search, the DACS system has not shown that it is behaving as it should be in MACS-VRPTW. Of course, this issue has triggered more questions about what sort of components are particularly important and therefore more reasonable variations are tried.

However, such variations are not able also to provide the comparable performance that is expected as in MACS-VRPTW. Then after a thoughtful research and a caring investigation from the author's side, the push-forward and push-backward strategy PFPBS is added and this addition has made the results comparable especially to the results of MACS-VPRTW. Thereafter, more additions, such as the hybrid local search HLS and the 2-Opt move variant, has made the system show results that are competitive to those obtained by MACS-VRPTW and other VRPTW algorithms in the literature. Later, specific versions of multiple ant colony systems that use more deterministic ants have shown that they are better in performance than other approaches that use the pheromone ants and this discovery is indeed an interesting one.

Chapter 4 introduces multiple ant colony systems in Section 4.1. Then, the experimental methodology that is followed when testing such systems is explained in Section 4.2. In Section 4.3, the motivations behind using such systems are mentioned. Next in Section 4.4, this chapter describes in detail the components of the best DACS system explored so far. Soon, this chapter starts in Section 4.5 talking about a DACS system that uses the XCHNG local search and some of the experiments done with it. Later in Section 4.6, new reconfigurations of DACS systems and some experimental work on them will be talked about in detail. Those new reconfigurations show the effects of including components like the "triple moves" local search, the parallel ants, the candidate lists...etc. Also, the chapter describes in

Section 4.7 specific multiple ant colony systems that are tried and the experimental work done on them. These specific systems try components like the push-forward and push-backward strategy PFPBS and colonies with unique objective functions that are different from the ones that reduce the traveled distances or the number of vehicles. This section discovers also the effects of using components such as the hybrid local search HLS, the 2-Opt move and the saving ants. Moreover, it tries to discover which component (of the pheromone trails and the local searches) causes the synergetic effects of the ants and it examines the possibility of changing the pheromone ants to more deterministic ones that use simple heuristics. Finally, a summary of Chapter 4 is introduced in Section 4.8.

4.1 An introduction to multiple ant colony systems

Multiple ant colony systems depend basically on Ant Colony Systems ACSs and Ant Colony Optimisation ACO as mentioned in [4] [50] [51] [52] and [75]. ACO is a metaheuristic technique that is inspired by the foraging behaviour of real ants in colonies. However, multiple ant colonies are introduced in [4] to tackle multiple objective problems. For example in a vehicle routing problem, a multiple ant colony system would have two colonies or ACSs seeking two different objectives.

One colony minimises the number of vehicles while the other colony minimises the total of travelled distances. Now in each colony, the ants use heuristics that include the pheromone trails deposited on the edges of the graph to build their solutions. Also, the ants diminish the pheromone trails of the edges visited in order to allow for new waves of ants to visit edges, which possibly have not been visited yet. In other words, each ant builds a solution of edges that contain all the customers should be visited. Once the solution of an ant is built, then the pheromone trails of the visited edges will be evaporated - evaporation process. Of course, the evaporation process allows the visited edges to be less attractive in order to give more opportunities for other edges to be more attractive and therefore to be used by the other ants that are still waiting to build their solutions. What's more, each built solution is then exposed heavily to local search improvements that enhance its quality.

In a colony, an attractive solution could be discovered. Once that attractive

solution is discovered, the quality of the solution is reported back to the multiple ant colony system or the coordinator that manages the work of both colonies described above. The coordinator has the ability to update the best global solution found so far, if the quality of the new solution is better. Afterwards, the colony increases the pheromone trails of the visited edges of the best global solution Ψ^{gb} with more pheromone - reinforcement process. Moreover, the pheromone trails of the unvisited edges in Ψ^{gb} will be diminished. As a consequence, the visited edges of the best global solution would have more opportunities to be considered in the solutions of the future waves of ants.

4.2 Experimental methodology

In this chapter, around 64 different multiple ant colony approaches are tested on the problem groups PG100, PG200 and PG400 in Section 2.2 and these approaches are related to each other according to the map in Figure 4.1. The approaches are implemented using the Java programming language and run on a PC machine with the following hardware features - Pentium IV with 2.66 GHz speed and 512 MB RAM. Of course, the performances of the approaches used in this chapter, cannot be directly compared with the performances of other VRPTW approaches in the literature, as mentioned in Tables 4.3, 4.12 and 4.13, on the basis of CPU time due to hardware and software differences. However the performances of the various VRPTW algorithms can be compared indirectly in order to enhance the understanding of how such systems do work.

Each approach is run on each problem instance of the six problem sets R1, C1, RC1, R2, C2 and RC2 of each problem group for a number of runs between 3 (one batch) and 30 (ten batches of three runs) inclusive. Each run is stopped after a limited amount of CPU time in seconds equal to 100, 300, 400, 600, 1200, 1800, 2400 or 4800. Of course, the approaches are tested in batches of three runs on each problem instance and this is according to what is mentioned in the article of MACS-VRPTW in [4]. The reason behind using each problem instance for three runs in such non-deterministic approaches is the fact that the runs are close in performance to one another. For any approach, this issue can be discovered after computing the precision factor in terms of the coefficient of variance CV, [standard

deviation/average]*100, as explained in Section 3.5.

For instance, the runs of the approach DACS+HLS+2-Opt are close in performance to one another and can bring after 1800 seconds very good results in three runs only. Thus, such approach is very precise on each problem set of Solomon [1] and CV on each problem set is less than or equal 1.60%. Hence, it should be enough for the commercial end user to run such a system a few times like three rather than so many times and then accept the best solution discovered. However in order to get competitive results with the state-of-the-art approaches, many runs (like thirty for instance) might be needed. Doing many runs might help also in forming some idea of how variable the results obtained are in such approaches.

Also in the case of doing many runs like thirty, it is not possible to report the final averaged results of all the batches used in each of the allocated amounts of CPU times. For that in this chapter, scenarios of the best, the average and the worst-case performances of some approaches on each problem instance and each problem set are created in order to have a better understanding of such systems in comparison to each other.

After the allocated amount of CPU time is elapsed in a number of runs of an approach, the results of all solutions, computed for any problem instance, are averaged over the number of runs done and later the averaged results of all problem instances in a problem set are averaged again over the number of problem instances in that problem set. Then where NV and TD refer to the number of vehicles and the total of traveled distances, the final averaged results of a problem set are reported, in addition to the standard deviation values of NV and TD and their percentages of deviations to other algorithms, in a table in front of the CPU time allocated. Also on each problem instance, the averaged NV and TD results of the many runs done and their standard deviation values and percentages of deviations to other algorithms are reported.

Also for each approach, the best and worst results computed, in all the experiments done for each problem instance, are usually extracted. This kind of extraction, from all the experiments, is done according to the way in which the best results of MACS-VRPTW in [4] are obtained. In this thesis also, the percentages of deviations of such best and worst results computed to those of other algorithms are considered. Moreover, the frequencies of those best and worst computed results are taken into

account. For example, if three or thirty runs for an approach are done, the solution with the best (or worst) result out of three or thirty solutions computed for a problem instance is extracted and reported and the frequency of that best (or worst) result is documented in addition to the percentage of deviation of that result to that of a particular algorithm. In published research on VRPTW, it is common to report the averaged result on a set of problem instances rather than emphasizing on finding the best and worst computed results for each problem instance and reporting them.

There are good reasons for this. The problem sets are each of a certain type. But the true interest lies in seeing what kind of approach does well on which problem instance. After all, any approach may sometimes discover a wonderful solution to a single problem instance by luck, but may also perform badly on many other problem instances of the same problem set. Therefore in this thesis for each approach tested, the best and worst computed results of all the problem instances in a problem set are reported in tables near to the problem instance numbers 1 to 12.

4.3 Motivations

There are four main motivations behind using multiple ant colony systems (MACSs) to tackle the VRPTW problem. The first motivation is to check in such MACS systems which ingredient of the two (the local searches and the pheromone trails) is doing the real work. Is it because of looking at the attractive edges from the pheromone point of view or is it simply because of using the local searches? Therefore, multiple ant colony systems are an opportunity to study and to investigate further.

The second motivation is the interest to check the effects of using different types of routing constructive and improvement heuristics on the decision making of the artificial ants. The third of these motivations is the way in which a multiple ant colony system would handle a multiple objective problem by letting a number of colonies to run together to pursue different objectives. The final motivation is the ability of the multiple ant colony systems to find very competitive solutions in a very short amount of CPU time and to check whether such systems have the ability to scale up when trying to solve problem instances larger than 100 customers - like 200 and 400 customers.

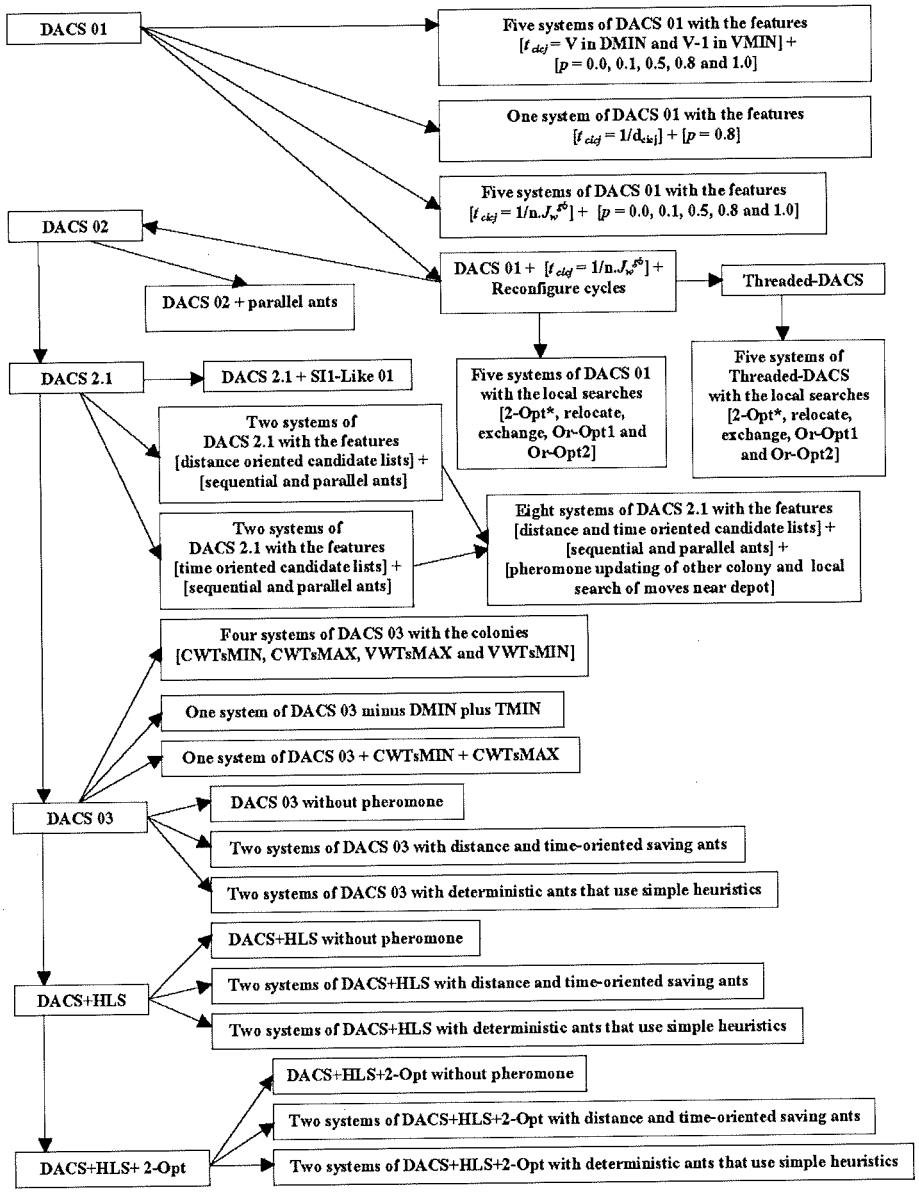


Figure 4.1: Map of multiple ant colony approaches.

4.3.1 Deficiencies of using static heuristics only in MACS systems

One of the key ingredients in an ant is using a static or a visibility heuristic $\eta_{c_i c_j}$, which may represent the inverse of the distance between two cities as in TSP such that $\eta_{c_i c_j} = 1/d_{c_i c_j}$ or the inverse of a complex value as in Equation 4.1 between two customer nodes or a customer node and the depot as in VRPTW.

$$\begin{aligned} T_{c_i c_j} &= \max(a_{c_i} + s_{c_i} + t_{c_i c_j},\, r_{c_j}) - (a_{c_i} + s_{c_i}) \\ V_{c_i c_j} &= d_{c_j} - (a_{c_i} + s_{c_i}) \\ C_{c_i c_j} &= \max(1.0,\, T_{c_i c_j} V_{c_i c_j} - U_{c_j}) \\ \eta_{c_i c_j} &= 1/C_{c_i c_j} \end{aligned} \tag{4.1}$$

The static heuristics suffer from many deficiencies and drawbacks that do not make such heuristics the appropriate choice for finding very good quality solutions to complex optimisation problems such as VRPTW. A static heuristic does not have the insight into why a customer, a vehicle or an edge that connects a vehicle and a customer together is chosen at a particular moment. For that, this choice does not mean that it is the right choice.

Therefore, choosing a customer, a vehicle or an edge at a particular time might lead also into building a very bad quality solution in the long term. Furthermore, the static heuristics do not differentiate between attractive and unattractive edges when they choose customers, vehicles or edges. As a consequence, these static heuristics drawback the efficiency and the performance of the routing builders in MACS systems in a way that makes them not have the ability to build alone the routes properly.

4.3.2 Competitive artificial ants

One notable feature about the artificial ants in ant systems ASs, ant colony systems ACSs and multiple ant colony systems MACSs is the ability to build solutions in a competitive way and in a very short amount of CPU time. The ants are made up of a number of ingredients (like the pheromone trails, the local searches...etc) that make them very competitive in finding very good quality solutions to complex problems in [50] [76] [52] and [4] like TSP, QAP, SOP, VRPTW...etc.

If an ant wants to build a solution using a static function like the ones described in Section 4.3.1 without the pheromone trails and the local searches, then the ant will build a low quality solution and it will look in its behaviour as greedy. In order to avoid building a low quality solution, an ant builds its solution using a heuristic $(\tau_{c_ic_j}.[\eta_{c_ic_j}]^{\beta})$ that has two sub-heuristics and later improves it using a local search of some sort.

The first sub-heuristic represents a static heuristic $\eta_{c_ic_j}$, while the second sub-heuristic represents a pheromone trail heuristic $\tau_{c_ic_j}$ that is variable. The purpose of the static heuristic is to show how short an edge connecting two customer nodes or a customer and the depot is in its static value. On the other hand, the purpose of the pheromone trail heuristic is to show how much attractive the edge is from the pheromone point of view. Now therefore in ant colony optimisation, each edge of a graph has two heuristic values and the pheromone trail heuristic is one of the key ingredients described above in making the ants as competitive as they are.

4.3.3 Attractive edges

When building the solution of an artificial ant, attractive edges are always looked at during the search. Therefore during the search, two simple components are used in addition to the static and pheromone trail heuristics of $\eta_{c_ic_j}$ and $\tau_{c_ic_j}$. The first simple component is called the exploitation mode while the second component is called the exploration mode.

In the exploitation mode, an ant chooses the edge that has the maximum heuristic value $(\tau_{c_ic_j}.[\eta_{c_ic_j}]^{\beta})$. However in the exploration mode, the artificial ant chooses with a great probability an edge that is short and has a great amount of deposited pheromone but not necessarily the edge with the maximum heuristic value as in the exploitation mode. Once an edge is selected using either the exploitation or the exploration mode, the pheromone trail of that selected edge is diminished.

Now once an ant has built its solution, it improves its quality using a local search and then another ant is set free to build its solution but here the new ant might use a number of edges different from those edges visited by the preceding ant. The reason is that the amounts of pheromone trails on the edges visited by the preceding ant are going to be lesser when a new ant is activated to build its solution. Accordingly, the new ant might build its solution using other attractive edges that are different

from the edges built by the preceding ant.

Of course, this process of building solutions continues until all the ants available at a colony build their solutions. Once all the ants available in a colony build their solutions, the solution with the best quality value may replace the best global solution found so far and the pheromone trails of the visited edges of the best global solution are reinforced with more pheromone in order to let such visited edges have more opportunities to be attractive for the next wave of ants.

4.3.4 How to treat the deficiencies

In order to treat the deficiencies talked about in Section 4.3.1 and therefore to improve the performance of the static heuristics in meta heuristic systems, a number of updates need to be considered. Firstly, the static heuristics of $\eta_{c_i c_j}$ needs to be combined with the pheromone trail heuristics of $\tau_{c_i c_j}$ in order to help producing new solutions that are attractive.

Secondly, the routing builder of a solution needs to be designed in a way to have transition components like the exploitation and the exploration modes that choose edges based on heuristics that combine the two heuristics of $\eta_{c_i c_j}$ and $\tau_{c_i c_j}$. Thirdly, the routing builder of a solution needs to be developed with two more components - pheromone evaporation and reinforcement. Finally and most importantly, the local search should be considered somewhere in the structure of the routing builder itself.

4.4 Double Ant Colony System

Because of the motivations talked about in Section 4.3, studying and further investigating a multiple ant colony system, called MACS-VRPTW [4], has become inevitable and necessary. As said before, this system uses two ant colonies that minimize the number of vehicles and the total of travelled distances separately. The multiple ant colony system used here is called as DACS, a double ant colony system, which has a lot of similarities with MACS-VRPTW and is described in detail in the following subsections.

4.4.1 DACS as a coordinator

In this system, DACS is a coordinator of two colonies that are pursuing two different objectives and has two main phases as in Figure 4.2. The first phase in the coordinator DACS is an initialisation phase in which an initial solution that is feasible is created and assigned to be as the best global solution. The initial solution is created using a greedy heuristic called the nearest neighbourhood heuristic NN. The NN heuristic creates the initial solution of a problem instance using an unlimited number of vehicles. Therefore, the upper limit of the number of vehicles might be created is equal to the number of the customers in the graph. At the end of the NN procedure, an initial solution with a number of vehicles equal to V, which is less than or equal to the upper limit of the number of vehicles, should be created.

Subsequently, each node c_i, whether a customer or a depot, is created with two candidate lists "time-oriented and distance-oriented" of n candidate nodes (that equal the number of nodes in a problem instance) for using them within the colonies, should be created in the cycle phase of the coordinator. The colonies apply such candidate lists within the routing builder of each ant. The time-oriented candidate lists are used especially in the part, which looks and accounts for feasible edges before any usage of the probabilistic state transition rule described in Section 4.4.5. Also, they are used within the local searches in Sections 4.4.8 and 4.4.10. Conversely, the distance-oriented candidate lists are used only within the insertion procedure in Section 4.4.7.

Each time-oriented candidate list is sorted in an ascending order according to what is called the time-oriented formula in Equation 4.2. Thus, such candidate lists are called time-oriented candidate lists because the candidate nodes are sorted not just depending on the travelling time $d_{c_ic_j}$ between any two nodes of c_i and c_j but also using the ready times r_{c_i} and r_{c_j}. Now, if there are any infeasible candidate nodes for any reason, then they are put at the end of the candidate list of the node c_i. The candidate nodes in each distance-oriented candidate list are ordered in an ascending order as well but according to the distance values only from a particular node c_i.

$$time_{c_ic_j} = \max(r_{c_i} + d_{c_ic_j}, r_{c_j}) \tag{4.2}$$

The second phase of the coordinator DACS is the cycle phase, which is run for a

```
//Initialise phase

I1- Create an initial solution using the nearest neighbourhood heuristic
    that uses an unlimited number of vehicles;
I2- Assign the initial solution as the best global solution;
I3- For each node ci whether a customer or a depot, create two kinds of
    candidate lists "time-oriented and distance-oriented" for using them
    within an ant's routing builder - described in Section 4.4.4. Hence
    in each kind, the candidate nodes should be ordered in an ascending
    way according to either the time-oriented formula in Equation 4.2 or
    the distance values between the node ci and the candidate nodes
    themselves;

//Cycle phase

C1-start time = getTime();
C2-new time = getTime();
C3-elapsed time = (new time - start time) / 1000;

while (elapsed time < CPU time limit) do

      C4-V = the number of the vehicles used in the best global
             solution;
      C5-Create two colonies VMIN and DMIN;
      C6-Initialise VMIN with V-1 and DMIN with V;
      C7-new time = getTime();
      C8-elapsed time = (new time - start time) / 1000;

      while (elapsed time < CPU Time limit) do

        C9- Activate the cycle of VMIN;
        If (VMIN has discovered a new best global solution with a number
            of vehicles that is less than V) do
            • break;
        C10-Activate the cycle of DMIN;
        If (DMIN has discovered a new best global solution with a number
            of vehicles that is less than V) do
            • break;
        C11-new time = getTime();
        C12-elapsed time = (new time - start time) / 1000;

      od while

      C13-  Kill the two colonies VMIN and DMIN currently active;
      C14-  new time = getTime();
      C15-  elapsed time = (new time - start time) / 1000;

od while
```

Figure 4.2: The coordinator DACS.

number of iterations until a limited amount of CPU time is elapsed. During the cycle phase, two colonies are active at all times and each colony is pursuing a different objective from the objective of the other colony.

The first colony is called VMIN, which tries using its ants to find a solution with a number of vehicles less than or equal to $V - 1$, which is less by one than the number V of the vehicles of the best global solution. The main objective of VMIN is to reduce the number of vehicles and therefore VMIN does not care about reducing the total of the travelled distances. However, the second colony is called DMIN, which uses its ants to minimize the total of travelled distances by trying to find a solution with a number of vehicles that is less than or equal to V. For that, the primary objective in DMIN is to reduce the total of travelled distances.

As a result, the DMIN colony might also get a solution with a reduced total of travelled distances and a reduced number of vehicles equal to V-1 at the same time. Now, if DMIN has found a solution improved in terms of the total of travelled distances, then the coordinator updates the best global solution and notifies the VMIN colony about that. On the other hand, if any of the two colonies is able to find a solution with a reduced number of vehicles less than or equal to $V - 1$, the coordinator updates the best global solution, kills the two colonies currently active and creates two new colonies with new limits for the number of vehicles to be used by the next wave of ants of each colony.

For example, once the best global solution is created with the NN heuristic, let us say, with a number of vehicles equal to 14.00 and a total of travelled distances equal to 1364.93, then the coordinator DACS creates two colonies of VMIN and DMIN with two different objectives. VMIN tries using its ants to find a solution with a number of vehicles less than or equal to 13. On the other hand, DMIN lets its ants use 14 vehicles as a limit in order to try to find a solution with a reduced total of travelled distances. Now, if DMIN finds a solution with a reduced total of travelled distances equal to 1255.33 for instance and a number of vehicles equal to 14, then the best global solution is updated and VMIN is notified about that.

However in the case there is a solution with a reduced number of vehicles like 12 from any of the two colonies, then the coordinator updates the best global solution with the new best solution found so far. Furthermore, the coordinator kills the two colonies already active and creates two new ones with two new limits for the number

of vehicles - i.e. the new VMIN is set with 11 vehicles as a limit and the new DMIN is set with 12 vehicles.

4.4.2 A colony of vehicle minimization - VMIN

The VMIN colony has the responsibility to minimize the number of vehicles as explained in Section 4.4.1 and therefore it is assigned with V-1 vehicles as a limit. In VMIN as in Figure 4.3, there are two major phases. The first phase is an initialisation step whereas the second phase is a cycle step.

The initialisation step is prompted to work through DACS as in the step that starts with the words 'initialise VMIN' in Figure 4.2. In the initialisation step, a solution is created with a number of vehicles equal to V-1 and assigned to be as the solution Ψ^{VMIN} with maximum visited customers, which is used only within VMIN. Also, a pheromone memory structure is created and the pheromone trail of each edge $e_{c_i c_j}$ representing two customers or a customer and a depot is initialised with the term τ_o that equals $1/(n.J^h_\Psi)$ where n refers to the number of visited nodes (depot or vehicle nodes + customer nodes) and J^h_Ψ represents the total of travelled distances of the solution Ψ^{VMIN}, created by the nearest neighbourhood heuristic NN.

After the pheromone trail initialisation, the pheromone memory structure might be used by the cycle phase of the current VMIN so many times. Also in DACS, pheromone trail re-initialisations are applied only when a new best global solution with a reduced number of vehicles is captured and there is a need to kill the current VMIN and create a new VMIN. Once the initialisation phase is finished, DACS as in Figure 4.2 activates the cycle phase of VMIN in order to find solutions and it is run for a number of iterations. In each of the iterations, ten ants are created and each ant builds its solution with V-1 vehicles using its routing builder in Section 4.4.4. The ants work sequentially and therefore the solutions are built in a way depending on a solution-by-solution basis. If none of the ten ants is able to build a feasible solution, the hybrid local search HLS in Section 4.4.10 is applied on the solution of the ant captured with the least unvisited customers. The main aim here is to let HLS enter as much as it could the unvisited customers in the solution of that captured ant.

Once all the ten ants have built their solutions, the solution S_{best} with maximum

```
//Initialise phase

I1- Make the maximum number of vehicles allowed for building
    solutions equal to V-1;
I2- Build a solution ψ^VMIN using the nearest neighbourhood heuristic NN
    with the maximum number of vehicles allowed;
I3- Get the number of nodes n (depot nodes + customer nodes) and the
    total of travelled distances J_ψ^h from the solution ψ^VMIN;
I4- Calculate τ_0 = 1/(n.J_ψ^h) to initialise the pheromone trail of each
    edge (i.e. two customers or a customer and a depot) in the
    pheromone memory structure with the τ_0 value;

//Cycle phase

while (the number of iterations is less than the maximum number of
       iterations allowed) do

      C1- Let ten ants build a solution each using the routing
          builder in Section 4.4.4;

      C2- if there are no feasible solutions built out of the ten
          ants in step C1, then

              a. Apply the hybrid local search HLS in Section 4.4.10
                 with the solution of an ant that has the least
                 unvisited customers;

      C3- Pick the solution S_best of an ant that has the maximum
          visited customers;

      C4- If the number of visited customers of the solution S_best is
          greater than the number of visited customers of the
          solution ψ^VMIN, then

              b. Let ψ^VMIN = S_best;
              c. Initialise with zero numbers a data structure that
                 memorises how many times each customer node has not been
                 visited in solutions;
              d. If the solution ψ^VMIN is feasible, make it as the new best
                 global solution ψ^gb or let ψ^gb = ψ^VMIN;

      C5- Use the two global update rules as in Section 4.4.9 to
          enforce the pheromone trails of the visited edges in each
          of the two solutions ψ^VMIN and ψ^gb with more pheromone
          trails;

      C6- Diminish the pheromone trails of the unvisited edges in the
          two solutions ψ^VMIN and ψ^gb as in Section 4.4.9;

od while
```

Figure 4.3: The colony VMIN.

visited customers is brought and compared with the solution Ψ^{VMIN} that has the maximum visited customers so far. If the number of visited customers of S_{best} is greater than the number of visited customers of Ψ^{VMIN}, then S_{best} is assigned to be the new best solution Ψ^{VMIN} so far with maximum visited customers. Then, a data structure, which memorises how many times each customer is not assigned to solutions built by ants, is re-initialised with zero values. Of course, this data structure is created before the beginning of any computation in the cycle phase of VMIN.

Later, if the solution Ψ^{VMIN} is feasible, then Ψ^{VMIN} is sent to the coordinator DACS and the best global solution Ψ^{gb} is updated with Ψ^{VMIN}. Furthermore, the coordinator issues the kill flag to kill the colonies VMIN and DMIN currently active in order, sometime later, to create two new colonies. Afterwards in VMIN whether the best global solution Ψ^{gb} is updated or not, two pheromone global update rules in Section 4.4.9 are used to reinforce those pheromone trails of the edges visited in the two solutions Ψ^{VMIN} and Ψ^{gb} found so far. On the other hand, the unvisited edges of the solutions Ψ^{VMIN} and Ψ^{gb} are diminished as explained in Section 4.4.9 in order to make such edges unattractive to ants that are going to search the solution space in the future cycles of the current VMIN.

4.4.3 A colony of distance minimization - DMIN

The DMIN colony as in Figure 4.4 minimizes the total of travelled distances and it has also two phases that are called the initialisation and cycle phases as in the VMIN colony but with few differences. In the initialisation phase, DMIN is assigned with a limited number of vehicles equal to V through a step that starts with the words 'initialise DMIN' as in Figure 4.2 in the coordinator body of DACS.

Next, a pheromone memory structure is created within DMIN and the pheromone trail of each edge $e_{c_i c_j}$ is initialised with the term τ_o as in VMIN exactly but here the τ_o value in DMIN is going to be different from that of VMIN because both colonies use different numbers of vehicles as limits. Also, pheromone trail re-initialisations are applied only whenever a new DMIN is created instead of a current one because of finding a new best global solution with a reduced number of vehicles. Therefore, the pheromone memory structure might be used by the cycle phase of DMIN currently active so many times before any pheromone trail re-initialisation could happen.

```
//Initialise phase

I1- Make the maximum number of vehicles allowed for building
    solutions equal to V;
I2- Build a solution using the nearest neighbourhood heuristic NN
    with the maximum number of vehicles allowed;
I3- Get the number of visited nodes n (depot nodes + customer nodes)
    and the total of travelled distances J_π^h from the solution created
    in the step I2;
I4- Calculate τ_0 = 1/(n.J_π^h) to initialise the pheromone trail of each
    edge (i.e. two customers or a customer and a depot) in the
    pheromone memory structure with the τ_0 value;

//Cycle phase

while (the number of iterations is less than the maximum number of
       iterations allowed) do

         C1-Let ten ants build a solution each using the routing
            builder in Section 4.4.4;

         C2-if there are no feasible solutions built out of the ten
            ants in step C1, then

               a. Apply the hybrid local search HLS in Section 4.4.10
                  with the solution of an ant that has the least
                  unvisited customers;

         C3-Pick the solution S_best of an ant that has the least total
            of travelled distances;

         C4-If the solution S_best is feasible and its total of
            travelled distances is less than the total of travelled
            distances of the solution ψ^gb, then let ψ^gb = S_best;

         C5-Use a global update rule as in Section 4.4.9 to enforce
            the pheromone trails of the visited edges in the solution
            ψ^gb with more pheromone trails;

         C6-Diminish the pheromone trails of the unvisited edges in
            the solutions ψ^gc as in Section 4.4.9;

od while
```

Figure 4.4: The colony DMIN.

Soon, the cycle phase of the DMIN colony is activated by DACS as in Figure 4.2 in order to reduce the total of travelled distances but this is true only after the activation of the cycle phase of the VMIN colony. The cycle phase of DMIN is run for a number of iterations that create ten ants each. In each of the iterations, the ants' solutions are built sequentially or depending on the solution-by-solution basis. Each ant builds its solution with V vehicles at most using its routing builder in Section 4.4.4 and the feasible solution of an ant with the least total of travelled distances is always preferred over the other feasible solutions. If there are no feasible solutions created, the hybrid local search HLS in Section 4.4.10 is called in order to try to make the solution of the ant with the least unvisited customers feasible.

Hence, the best feasible solution S_{best} in terms of the total of travelled distances is selected and compared with the best global solution Ψ^{gb} found so far. If S_{best} is better in quality than Ψ^{gb}, then S_{best} becomes the new best global solution Ψ^{gb}. If a new best global solution Ψ^{gb} has a reduced number of vehicles, then a kill flag will be issued by DACS in order to kill, sometime later, the DMIN and VMIN colonies currently active. Thereafter whether the best global solution Ψ^{gb} is modified or not, DMIN uses a global update rule as in Section 4.4.9 that reinforces the pheromone trails of the visited edges in Ψ^{gb} with more pheromone trail amounts. Afterwards in order to help the ants in future DMIN cycles concentrate on the desired points in the solution space, the pheromone trails of the unvisited edges in the solution Ψ^{gb} are diminished as talked about in Section 4.4.9.

4.4.4 Ants' routing builder

In the cycle phases of the VMIN and the DMIN colonies, each ant builds its routes using a routing builder as in Figure 4.5. The routing builder is equipped now with a number of components that would make each ant build its routes in a way different from the routes built by the other ants in a colony. Those components, as described in Sections 4.4.5 to 4.4.8, are a probabilistic-state transition component with its exploitation and exploration parts, a pheromone local updating component, an insertion procedure and a local search of quadruple moves.

The routes of the vehicles, in an ant's solution, are built sequentially or depending on a depot-by-depot basis and thus the nodes of each route, as well, are inserted in a sequential way. Of course, this sequential nature can be recognized in Figure 4.5

```
a)  Visit a duplicated depot;

do
    b)  Ci = Get the last node visited whether it is a customer or a depot node;
    c)  time-oriented candidate list = Get the time-oriented candidate list of the node Ci;
    d)  CL_Size = Get the size of the time-oriented candidate list;

    for (j = 0; j < CL_Size ; j++) do  //This part calculates the value of each feasible edge that can be visited.

        e)  Cj = Get a node of the time-oriented candidate list at index j;
        if (Cj is a depot) then
                if (there are duplicated depots left)  then
                    f)  if the two nodes Ci and Cj are depots nodes, then continue;
                    g)  Make Cj equal to one of the duplicated depots;
                else if  (there are no duplicated depots left) then
                    h)  continue;
                od if-else
        else if (Cj is not a depot) then
                if (Cj is a visited customer) do
                    i)  continue;
                od if
        od if-else

        j)  if the node Cj is violating one of the three hard constraints H1 to H3 mentioned in Section 4.4.4, then continue;
        k)  ηcicj = Calculate the visibility value ηcicj between the two nodes Ci and Cj from the  formula in Equation 4.1;
        l)  τcicj = Get the pheromone trail value between the two nodes Ci and Cj;
        m)  Calculate the value  of τcicj * Math.pow(ηcicj, β);
    od for

    //Probabilistic state transition rule as explained in Section 4.4.5
    n)  u = U(0, 1);
    if (u <= pE) then  //Exploitation part.
        o)  Let the ant visit the edge with the maximum calculated value of τcicj * Math.pow(ηcicj, β);
    else if (u > pE) then  //Exploration part.
        //The greater the value of τcicj * Math.pow(ηcicj, β) is, the greater the chance of the edge with that value to be chosen.
        p)  Let the ant choose using the roulette wheel sampling scheme the edge that has a great value of τcicj * Math.pow(ηcicj, β) but
            not necessarily the maximum value;
    od if-else

    q)  Local update the pheromone trail of the chosen edge.  //Local pheromone updating rule as clarified in Section 4.4.6

    //This part checks the number of customers should be visited and the number of vehicles allowed.
    if (Cj in the chosen edge is a customer) then
        r)  Add Cj to the set of visited nodes;
    else if (Cj in the chosen edge is a duplicated depot) then
        if (the number of visited customers is not equal to the number of customers in a problem instance) then
                if (the number of vehicles used is less than the number of vehicles allowed to use) then
                    s)  Add the duplicated depot to the set of visited nodes;
                else if (the number of vehicles used is equal to the number of vehicles allowed to use) then
                    t)  break;  //Get out of the do while loop.
                od if-else
        else if (the number of visited customers is equal to the number of customers in a problem instance) then
                u)  break;  //Get out of the do while loop.
        od if-else
    od if-else
while (there are still feasible nodes to be visited);

v)  Insert the remaining unvisited customers, if the solution of an ant is infeasible;  //Insertion procedure as stated in Section 4.4.7
w)  Use the local search of quadruple moves to improve the solution of an ant, only if the solution of the ant is feasible and the ant is coming
    from the DMIN colony;  //Local search of quadruple moves as talked about in Section 4.4.8
```

Figure 4.5: Ants' routing builder.

from the do while loop. Briefly as in Figure 4.5, an ant that is located at a node c_i (whether a customer or a depot) accounts for the feasible edges that it can visit in a time-oriented candidate list and it calculates for each feasible edge $e_{c_ic_j}$ a value of $\tau_{c_ic_j}.[\eta_{c_ic_j}]^{\beta}$, which is a combination of the variable pheromone trail heuristic $\tau_{c_ic_j}$ and the static heuristic $\eta_{c_ic_j}$ in Equation 4.1. Each feasible edge has to obey the following hard constraints.

$H1-$ A route must contain enough capacity to serve a customer c_j.

$H2-$ A route must arrive at a customer c_j before his due date is reached.

$H3-$ A route must have an enough time to return back to the depot c_0 before its due date is reached.

Next, the ant uses the probabilistic-state transition component with its exploitation and exploration parts in Section 4.4.5 in order to choose edges to be a part of its solution. Once a feasible edge $e_{c_ic_j}$ connecting the two nodes c_i and c_j is chosen, a visit is arranged between c_i and c_j and the pheromone local updating component in Section 4.4.6 is used to diminish the pheromone trail of that chosen edge. In order to make sure that the routes of an ant are different from the routes built by other ants in a colony, the probabilistic-state transition component uses the heuristic $\tau_{c_ic_j}.[\eta_{c_ic_j}]^{\beta}$ mentioned earlier above.

Then after building sequentially the routes of an ant with visited nodes, the solution might be feasible or infeasible. If the solution is feasible, the insertion procedure in Section 4.4.7 is not used. On the other hand if the solution is infeasible, the ant uses the insertion procedure for inserting the remaining unvisited customers into its infeasible solution. Shortly afterwards, the ant that has a feasible solution and has come mainly from the DMIN colony applies a local search of quadruple moves in Section 4.4.8 in order to improve the quality of its feasible solution.

4.4.5 Probabilistic transition with exploitation and exploration

In the probabilistic state transition component, an ant starts choosing a feasible edge e_{c_0ci} that connects a duplicated depot and a customer c_i that is not visited yet in order to be a part of its solution. Once the ant has chosen the feasible edge e_{c_0ci}

and it is located at the customer c_i, then the ant will look for the next feasible edge $e_{c_ic_j}$ connecting an already visited customer c_i with the next node c_j to be visited. Of course, the ant will continue choosing edges to be a part of its solution until all the customers are visited.

Now, each feasible edge $e_{c_ic_j}$ is chosen using one of two transition modes - either the exploitation or the exploration mode. In 90% probability, an ant uses the exploitation mode to exploit the situation and to choose the best feasible edge, which is considered at the same time as a short edge, according to its static value computed by Equation 4.1, and an attractive edge from the pheromone point view. However in 10% probability, the exploration mode is used to explore in a set of feasible edges for a feasible edge, which is not necessarily the best edge.

In other words, the exploitation mode is used by an ant to choose a feasible edge $e_{c_ic_j}$ that maximises the heuristic value in the formula $\tau_{c_ic_j}.[\eta_{c_ic_j}]^\beta$ if a probability value, $u \in U(0,1)$, is less than or equal to the probability, $p_E = 0.9$. In this case, the feasible edge $e_{c_ic_j}$ with the maximum heuristic value is regarded as the best edge.

On the other hand in the exploration mode, if a probability value, $u \in U(0,1)$, is greater than the probability value ($p_E = 0.9$), the ant uses the proportionate selection operator with its roulette wheel sampling scheme to create a probability distribution for all the feasible edges that could be visited within the sight of an ant. Then, the ant chooses with a great probability, from a set of feasible edges, a feasible edge $e_{c_ic_j}$, which is not necessarily the best edge and has a great amount of a heuristic value calculated from the formula $\tau_{c_ic_j}.[\eta_{c_ic_j}]^\beta$. So, feasible edges with great heuristic values of $\tau_{c_ic_j}.[\eta_{c_ic_j}]^\beta$ have greater chances to be selected.

$$e_{c_ic_j} = \begin{cases} \max(\tau_{c_ic_j}.[\eta_{c_ic_j}]^\beta) & \text{, if } (u \in U(0,1)) \leq (p_E = 0.9) \\ \text{Eqs. 2.27 and 2.28} & \text{, if } (u \in U(0,1)) > (p_E = 0.9) \end{cases} \tag{4.3}$$

4.4.6 Local updating of pheromone

Once a feasible edge or arc $e_{c_ic_j}$ is selected by an ant through using either the exploitation or the exploration transition mode as in Section 4.4.5, a local updating rule in Equation 4.4 is used to diminish the pheromone trails of the two sides of the selected edge $e_{c_ic_j}$ - i.e. the pheromone trails of the edge sides $es_{c_ic_j}$ and $es_{c_jc_i}$ of the edge described earlier are evaporated simultaneously. Because the problem instances of VRPTW have symmetrical distance values on the two edge sides of each

arc or edge in the graph as in symmetric TSP [51], the pheromone trail amounts of the two sides of a selected edge are kept always identical or equal after using the local pheromone updating rule. Also as a result to what is mentioned earlier, the global pheromone updating component in Section 4.4.9 does the same thing, the time it is applied on the pheromone trails.

The purpose of the local updating rule is to make the edges, selected by an ant, unattractive and to let the other ants look for other edges that are possibly different from the edges selected by the ant that has already finished building its solution. Consequently in Equation 4.4, the value between the brackets $(1-\rho)$ helps in evaporating some amount of the pheromone trails $\tau_{c_ic_j}$ and $\tau_{c_jc_i}$ deposited on the edge $e_{c_ic_j}$ once it is selected. The term ρ is the evaporation parametric value, which could be any value between 0 and 1. On the other hand, the term τ_o is calculated differently in each of the two colonies VMIN and DMIN, as described in Section 4.4.1, in a way that depends on mainly on the number of vehicles allowed to use in each colony.

$$\begin{aligned} \tau_{c_ic_j} &= (1-\rho).\tau_{c_ic_j} + \rho.\tau_o \\ \tau_{c_jc_i} &= (1-\rho).\tau_{c_jc_i} + \rho.\tau_o \end{aligned} \tag{4.4}$$

4.4.7 Insertion procedure

The insertion procedure comes to the stage only when an ant has finished building its solution that is infeasible. The infeasible solution of an ant is built after the sequential use of the exploitation and exploration transition modes in Section 4.4.5 and thus a number of customers are regarded as unvisited in that solution. In this case, the ant that has an infeasible solution uses the insertion procedure in Figure 4.6 in favor of trying to insert the unvisited customers into the feasible parts of the infeasible solution already built.

In the insertion procedure, the unvisited customers are sorted first in a descending order according to their demand quantities - from the largest to the smallest demand. Then starting from the unvisited customer with the largest demand on the sorted list, each unvisited customer is picked for insertion near one of the visited customer and depot nodes in the infeasible solution of an ant. Once an unvisited customer is chosen, its distance-oriented candidate list is brought. As explained in

```
//Insertion procedure of ants...
If (an ant has built an infeasible solution and is coming from the
VMIN or DMIN colony) then

    (a) Sort the unvisited customers in a descending order according
        to their demand quantities;

    For (each unvisited customer in the sorted list of the previous
         step) do

       (b) distance-oriented candidate list = get the distance-
           oriented candidate list of an unvisited customer;
       (c) index = 0;

       while (there is no insertion for an unvisited customer) do

            (d) Pick a visited node of the distance-oriented candidate
                list in step (b) at some index;

            if (the visited node is a customer) then
                  (e) Try to arrange a visit near the visited customer
                      chosen either before him or after him;
                  if (the insertion near the visited customer is
                       successful) then
                       (f) break;
                  od if
            else if (the visited node is a depot) then
                  (g) breakFlag = false;
                  for (each tour in the infeasible solution) do
                     (h) Try to arrange a visit near the depot of a
                         tour either at the start or at the end of
                         that tour;
                     if (the insertion in a tour is successful) then
                         (i) breakFlag = true;
                         (j) break;
                     od if
                  od for

                  if (breakFlag == true) then
                        (k) break;
                  od if
            od if-else

            (l) index = index + 1;
            if (the index has reached the end of the distance-
                oriented candidate list in step (b)) then
                  (m)   break;
            od if

       od while

    od for
od if
```

Figure 4.6: The insertion procedure.

Section 4.4.1, the candidate nodes in such list are ordered in an ascending order (from the nearest to the farthest node) according to how close in distance they are from a particular node, which is here the unvisited customer chosen already.

Subsequently in order to try to insert the unvisited customer chosen already, the nearest visited node on the distance-oriented candidate list is chosen. If the visited node is a customer, then the unvisited customer is inserted into one of two insertion points - either before the visited customer or after him. However, if the visited node is a depot, then the insertion points will be also two but one at the start of the tour and the other at the end of the tour. If both insertion points near the visited customer or the depot are feasible, then the insertion point that minimises distance better than the other is always accepted. If in any case the both insertion points are not feasible, then the second nearest visited node on the distance-oriented candidate list is tried and so on. Afterwards whether the unvisited customer is inserted or not, the next unvisited customer is picked up and the same procedure described above is repeated.

4.4.8 Local search of quadruple moves

Now, the ants that are coming from the DMIN colony and have feasible solutions use the local search of quadruple moves in Figures 4.7 and 4.8, which goes on in its work until no more improvement is found. This local search depends on four major move operators that are mentioned below in M1 to M4 and uses a time-oriented candidate list of maximum 20 closest nodes (i.e. maxNumberOfCandidates in Figure 4.8) for each customer. The time-oriented candidate list of each customer is ordered in an ascending order according to the formula in Equation 4.2 and should be created before any computation could be done in any of the two colonies of the DACS system.

Thus given a feasible solution, the local search works through each vehicle's route in turn in the order of the built tours of a solution from left to right, and for each visited customer on the route it tries three or four of the possible 'moves' described below. It starts working from the first visited customer on the left side of each tour and hence the visited customers are, one by one, considered.

M1- Swap the visited customer c_i with the nearest customer c_j on the time-oriented

candidate list of c_i.

M2- Relocate the visited customer c_i after the nearest customer c_j on the time-oriented candidate list of c_i.

M3- Relocate the nearest customer c_j, on the time-oriented candidate list of the visited customer c_i, after c_i himself.

M4- Use a variant of 2-Opt to swap two edges $e_{c_i c_{i+1}}$ and $e_{c_j c_{j+1}}$ in the two routes selected for improvement purposes, with two new edges $e_{c_i c_{j+1}}$ and $e_{c_j c_{i+1}}$ that would lead into having two new routes.

The first three moves are designed to be as intra-route and inter-route improvement operators at the same time whereas the last and fourth move is only regarded as an inter-route improvement heuristic. Now, each of the previous four moves might or might not create a new feasible solution. Therefore, there is a possibility to create a number of solutions between 0 and 4 feasible neighbouring solutions. If only one feasible solution is created, then that solution is compared with the current solution of an ant. In the case of creating two or more feasible solutions, the best of the new feasible solutions should be considered as the one that should be compared with the current solution of an ant already built.

If all moves between a visited customer and his nearest customer are not feasible, then the visited customer is tried with the second nearest customer on his time-oriented candidate list and so on. If a 'move' improves the current solution of an ant, it is accepted as the new current solution. For that, the new solution replaces the old one and breaks out of the while loop in Figure 4.8 and the QuadrupleMovesLS local search continues with the next visited customer on the route.

4.4.9 Global reinforcement and diminishing of pheromone

As it is described in the pheromone local updating rule in Section 4.4.6 and because of the symmetrical distance values between any two nodes in the VRPTW graph as in symmetric TSP [51], the pheromone trails $\tau_{c_i c_j}$ and $\tau_{c_j c_i}$ of the two sides in each edge or arc, whether visited or unvisited in the best global solutions, are globally updated and kept identical or equal in the DACS system.

```
If (an ant has built a feasible solution and is coming from the DMIN colony)
then LocalSearch(The solution of that ant);

LocalSearch(The solution of an ant)
begin proc

    (a)  improvementFlag = true;

    while (improvementFlag == true)  do

        (b)  improvementFlag = improve(The solution of an ant);

     od while

end proc
```

Figure 4.7: The local search of quadruple moves.

The global updating of the pheromone trails happens at the end of each cycle phase of the colonies VMIN and DMIN. Whether the best global solution Ψ^{gb} is updated or not, a colony like VMIN or DMIN uses Equation 4.5. The term J_{Ψ}^{gb} equals in Equation 4.5 the totals of traveled distances done by the tours in Ψ^{gb} and the term ρ is an evaporation parameter that could be any value between 0 and 1 - inclusive.

The pheromone global updating rule in Equation 4.5 is used to reinforce, using the reinforcement factor ρ/J_{Ψ}^{gb}, the pheromone trails of all the edges visited in Ψ^{gb} with more amounts of pheromone trails. Also, the pheromone trails of all the edges, whether visited or unvisited in Ψ^{gb}, are diminished using the $1-\rho$ sub-formula. Now in the case where an edge is not visited, the reinforcement factor described earlier becomes equal to zero, which makes the unvisited edge much more unattractive from the pheromone point of view.

In VMIN, there is an extra pheromone global updating rule in Equation 4.6 that is used in addition to the one described in Equation 4.5 and works in the same way as described above. But, the main difference between the two equations in 4.5 and 4.6 is that the best-found solution Ψ^{VMIN} so far with maximum visited customers is used instead of the best global solution Ψ^{gb} and therefore the term J_{Ψ}^{VMIN} refers to the totals of traveled distances done by the tours in Ψ^{VMIN}. For that in using Equation 4.6 with Ψ^{VMIN}, VMIN has also the responsibility of reinforcing and

```
boolean improve(The solution of an ant)
begin proc

(a)  A collection of tours = The solution of an ant;
(b)  number of tours = Get the size of the collection of tours already brought;
(c)  improveFlag = false;

for (i = 0; i < number of tours; i++) do
    (d)  tour1 = Get the tour at index i in the collection of tours;
    (e)  tourSize1 = Get the size of tour1;

  for (j = 1; j < tourSize1; j++) do
    (f)  visitedID = Get the ID No. of the visited customer at index j in tour1;
    (g)  visited customer = customers.elementAt(visitedID);
    (h)  tourIndex1 = From the ant, bring the tour index of the tour where visitedID exists;
    (i)  Let updateFlag = false and nearest counter = 0;

    while (nearestCounter < maxNumberOfCandidates) do
          (j)  nearestID = Get the ID No. of the nearest customer located
                           according to the nearest counter in the time-oriented candidate list
                           of the visited customer;
          (k)  nearest customer = customers.elementAt(nearestID);
          (l)  tourIndex2 =  From the ant, bring the tour index of the tour where nearestID
                             exists;
          (m) tour2 = Get the tour located at tourIndex2 in the collection of tours;
          (n)  tourSize2 = Get the size of tour2;

          if (tourIndex1 == tourIndex2) then
                    (o)  Make the three moves explained above in M1 to M3 on the same tour
                         (i.e. either on tour1 or tour2);
                     If (the distance of the new tour of a move is less than the distance of
                        the tour located either at tourIndex1 or at tourIndex2) then
                          •  updateFlag = true;
          else if (tourIndex1 != tourIndex2) then
                    (p)  Make the four moves explained above in M1 to M4 using the two different
                         tours (i.e. tour1 and tour2);
                     If (the sum of the distances of the two new tours created by a move is less
                        than the sum of the distances of the tours already located at tourIndex1and
                        at tourIndex2) then
                          •  updateFlag = true;
           od if-else

           if (updateFlag == true) then
                    (q)  Update the information of a tour or two tours in the collection of
                         Tours;
                    (r)  tour1 = Get the tour at index i in the collection of tours;
                    (s)  tourSize1 = Get the size of tour1;
                    (t)  Let j = 0 and improveFlag = true and then break;
           od if

           (u)  nearest counter += 1;

    od while

  od for

od for

(v)  return improveFlag;
end proc
```

Figure 4.8: The improvement procedure of the "quadruple moves" local search.

diminishing the pheromone trails of all the edges whether visited or unvisited in that solution.

As a result to what is described above, the pheromone memory structure of VMIN is double updated using two pheromone global update rules whereas in DMIN it is single updated. The purpose of the pheromone global updating rules is not just to update the pheromone trail amounts of the edges in a graph but also to enable the ants in the future cycle phases of the colonies involved to seek for and use edges that are attractive and therefore to have more quality and competitive solutions.

$$\begin{aligned}
\tau_{c_i c_j} &= (1-\rho)\tau_{c_i c_j} \\
\tau_{c_j c_i} &= (1-\rho)\tau_{c_j c_i} && \forall e_{c_i c_j} \ni \Psi^{gb} \\
\tau_{c_i c_j} &= (1-\rho)\tau_{c_i c_j} + \rho/J_{\Psi}^{gb} \\
\tau_{c_j c_i} &= (1-\rho)\tau_{c_j c_i} + \rho/J_{\Psi}^{gb} && \forall e_{c_i c_j} \in \Psi^{gb}
\end{aligned} \tag{4.5}$$

$$\begin{aligned}
\tau_{c_i c_j} &= (1-\rho)\tau_{c_i c_j} \\
\tau_{c_j c_i} &= (1-\rho)\tau_{c_j c_i} && \forall e_{c_i c_j} \ni \Psi^{VMIN} \\
\tau_{c_i c_j} &= (1-\rho)\tau_{c_i c_j} + \rho/J_{\Psi}^{VMIN} \\
\tau_{c_j c_i} &= (1-\rho)\tau_{c_j c_i} + \rho/J_{\Psi}^{VMIN} && \forall e_{c_i c_j} \in \Psi^{VMIN}
\end{aligned} \tag{4.6}$$

4.4.10 Hybrid Local Search HLS

During the search of a colony, there is a possibility for the cycle of a VMIN or DMIN colony not to find any feasible solutions using the ants used in that cycle. For that as in the step a in Figures 4.3 and 4.4, a component called Hybrid Local Search HLS is used to capture the infeasible solution of an ant with the least unvisited customers and to insert such remaining unvisited customers in the feasible parts of that infeasible solution.

The Hybrid Local Search HLS is a combination of mainly two components, which are the insertion procedure and the local search of quadruple moves in Sections 4.4.7 and 4.4.8. The combination is managed by putting a call for the insertion procedure between the steps q and r of the local search of quadruple moves in Figure 4.8. For that in HLS, if there is an improvement for the feasible parts of the current infeasible solution of a captured ant after applying a move of the four moves, HLS tries later to insert, using the insertion procedure, the remaining unvisited customers as much as it

can between the visited customers of the current infeasible solution newly improved. In order to let HLS deal with the remaining unvisited customers effectively without logical problems, the local search of quadruple moves, used in HLS, is modified also with another thing, which is to ignore any unvisited remaining customers regarded as nearest customers in the time-oriented candidate list of a visited customer.

The hybrid local search HLS continues in its work until no improvement can be achieved. Afterwards, the current solution of a captured ant either becomes a feasible solution or stays as an infeasible solution. For example after applying HLS in the VMIN colony, the quality of the current solution, whether feasible or infeasible, of the captured ant is compared with the quality of the solution Ψ^{VMIN} with the maximum visited customers. Therefore, if the number of visited customers of the current solution is exceeding the maximum number of visited customers of the solution Ψ^{VMIN}, the current solution replaces the solution Ψ^{VMIN}.

On the other hand in the DMIN colony and after applying HLS, the quality of the current feasible solution of a captured ant is compared with the quality of the best global solution Ψ^{gb} found so far. If the current solution is having a total of travelled distances that is less than the total of travelled distances of the best global solution Ψ^{gb}, the best global solution Ψ^{gb} is replaced with the current solution.

4.4.11 Push-forward and push-backward strategy PFPBS

In the DACS system, when an ant uses the probabilistic state transition rule with its exploitation and exploration parts in Section 4.4.5 to visit nodes and to build its solution, the ant does store any information about the visited nodes of its solution, as in the points I1 to I6 enumerated below, anywhere in its body. Furthermore during the usage of the insertion procedure in Section 4.4.7 or the local search of quadruple moves in Section 4.4.8 with the solution of an ant, there is a use for a push-forward and a push-backward strategy PFPBS that would push either forward or backward customers and would use at the same time the information of other nodes, not pushed either backward or forward. Of course, this is in order to update the information of the pushed customer nodes, either forward or backward, in a number of neighbouring solutions - which might be created for trying to replace the solution of an ant with the best one of them.

I1- The total of travelled distances up to the node n_{ik} in the route R_k.

I2- The total of consumed time in travelling, vehicle waiting and servicing up to the node n_{ik}, including its service time, in the route R_k.

I3- The total of consumed time in travelling, vehicle waiting and servicing up to the node n_{ik}, excluding its service time, in the route R_k.

I4- The total of items picked up or delivered up to the node n_{ik} in the route R_k.

I5- The total of waiting time done by a vehicle, to the start times of customers to begin, up to the node n_{ik} in the route R_k.

I6- The total of waiting time done by a collection of customers, to a vehicle to arrive, up to the node n_{ik} in the route R_k.

Now in a colony whether VMIN or DMIN, if the insertion procedure or the QuadrupleMoves local search happens to make some moves (such as relocating, swapping nodes and/or swapping edges) to the current solution of an ant and the quality of each new neighbouring solution is measured without depending on any stored information and any push-forward and push-backward strategy PFPBS, then a lot of waste CPU time is expected as a consequence.

For that in the current solution of an ant, if a move is used between two nodes c_i and c_j located on the same route or in two different routes, the stored information of the nodes c_{i-1} and c_{j-1} are, most likely, to be applied in updating the information of the nodes c_i and c_j and any following nodes, which might be part of a route or two routes in a new neighbouring solution. Since more than one neighbouring solution could be created, the best neighbouring solution is always selected as said before. Next, if the best new neighbouring solution has to replace the current solution of an ant because of its better quality result, the information of the nodes and any following nodes involved in the move that has created that solution is stored in the ant's body instead of any old information.

As a conclusion to what is described above, storing information related to the visited nodes of each route in the current solution of an ant and later retrieving back and updating such information as required have its effects as described below. Doing that helps in saving the CPU time during the usage of the insertion procedure and

the local search of quadruple moves. Also, it assists in improving the performance and the results of any DACS system. In addition, it makes an ant have the ability of building new solutions easily by making slight moves or modifications to the current solution. Thus, using the information of some of the nodes kept, as they are located in the current solution of an ant, helps in building new information about the nodes shifted backward or forward - according to where they are newly located in the new neighbouring solutions.

4.5 Initial experimental work on the first double ant colony system - DACS 01

This section talks about the experimental work done on the first double ant colony system called as DACS 01 and the other kinds of experiments done later in order to improve its performance. In DACS 01, some of the ingredients described in Section 4.4 are not used as those mentioned in A1 to A5.

A1- The pheromone trail re-initializations.

A2- The candidate lists.

A3- The global pheromone trail diminishing of the unvisited edges in the best global solutions Ψ^{VMIN} and Ψ^{gb}.

A4- The push forward and push backward strategy PFPBS.

A5- The hybrid local search HLS.

Also, other kinds of ingredients as in B1 to B6 are used instead of some of the other ingredients described in Section 4.4.

B1- The cycle phase of the coordinator DACS 01 in Figure H.1 is used instead of the cycle phase in Figure 4.2.

B2- The routing builder of an ant in Figure H.2 is used instead of the routing builder in Figure 4.5.

B3- The XCHNG local search in Figure H.3 is used instead of the local search of the quadruple moves in Figures 4.7 and 4.8. The local search XCHNG in Figure H.3 has some similarities with the CROSS-exchanges local search talked about in the literature [8] [5] [28] and [4]. In each of the iterations in Figure H.3, two tours are selected randomly from the built tours of the solution of an ant using the basic component of XCHNG, which is regarded as the move operator. Then after selecting two tours randomly, two segments of customers are selected and exchanged as in Figure 4.9 in an effort to improve the quality of the two tours and thereafter the solution quality. Each of the two segments must have a length between 1 and 3 customer nodes. The reason for having the length of each segment limited between 1 and 3 is to give more chances for XCHNG to improve the quality of the solution.

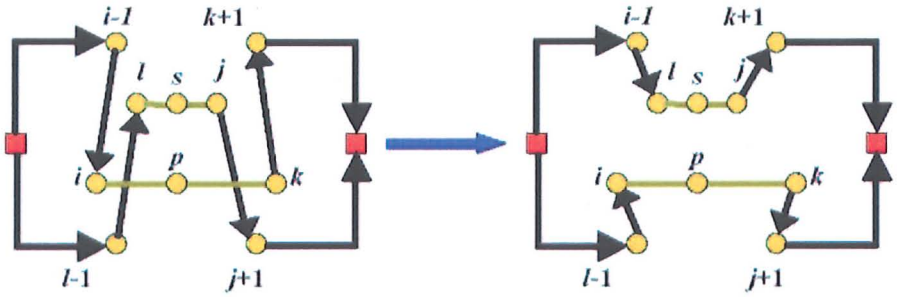


Figure 4.9: The move operator of CROSS-exchanges for inter-route improvements.

B4- Initialing the pheromone trail of each edge with $1/d_{c_i c_j}$ is used as a substitute to the pheromone trail initialization way of the term τ_o mentioned in Sections 4.4.2 and 4.4.3.

B5- The "one edge side" pheromone trail updating is used as a substitute to the arc or the "two edge side" updating one described in Sections 4.4.6 and 4.4.9.

B6- The term τ_o is initialized in each colony with the NV and TD values of a solution created using the nearest neighborhood heuristic that uses an unlimited number of vehicles. In Sections 4.4.2 and 4.4.3, the term τ_o is initialized in each colony in the same way described earlier but the nearest neighborhood heuristic uses a limited number of vehicles that depends on the kind of the colony involved.

In the system DACS 01, the parametric values in Table 4.1 like the number of ants m used in each colony, the evaporation parameter ρ, the heuristic value β and the transition mode probability p_E are chosen to be the same as in the MACS-VRPTW system [4]. The cycle phase of DACS 01 runs for a number of iterations equal to 2 and the cycle phase of each colony whether VMIN or DMIN runs for 10 iteration times. Furthermore, the XCHNG local search of DMIN is run for 150 iterations.

Table 4.1: The parametric values used in the system DACS 01.

Number of artificial ants (m)	10
Evaporation parameter (ρ)	0.1
Heuristic value (β)	1
Transition mode probability (p_E)	0.9
Maximum iterations of the cycle phase Of the system DACS 01	2
Maximum iterations of the cycle phase of each colony (VMIN or DMIN)	10
Maximum iterations of the XCHNG local search	150

At the end of running the system DACS 01, a total of 400 ants will be created from VMIN and DMIN - i.e. 2 iterations of the cycle phase of DACS 01 $\times$ 10 iterations of the cycle phase of each colony $\times$ 2 colonies. Note that in this system, the number of iterations in the cycle phases of DACS 01 and the two colonies are chosen arbitrarily because the stopping criteria used in the system MACS-VRPTW [4] and its colonies are not given clearly as explained in Section 3.1. Also, the same is true for the number of iterations of the XCHNG local search and for how XCHNG does work. As a consequence, many aspects of XCHNG and whether such aspects are working as expected, in the CROSS-exchanges local search of MACS-VRPTW [4], are guessed.

At the beginning of this section, the XCHNG local search and its effects are checked as in Section 4.5.1. Then in Section 4.5.2, a new pheromone trail initialization technique is tested. Thereafter with a new initialization way of the pheromone trails, the issue of reinitializing the pheromone trails is tried in Section 4.5.3. Later in

Sections 4.5.4 and 4.5.5, reconfiguring the cycles of certain components and threading the colonies are looked at and investigated. Finally, local searches that apply a single move operator each are experimented with as in Section 4.5.6.

4.5.1 Is the XCHNG local search doing any good?

After implementing the system DACS 01 as described in Section 4.5, this section checks to see if the XCNHG local search is doing any goodness to the performance. Consequently, DACS 01 is tested with and without XCHNG on the problem group PG100 in Section 2.2 for three runs according to the experimental methodology in Section 4.2. In each run, each problem instance is used for 300 to 400 seconds in CPU time terms.

One of the most important aspects explored afterwards is that the system DACS 01 without XCHNG is doing horribly bad. But, the performance of the system DACS 01 that uses the XCHNG local search is not performing as well as those of the systems MACS-VRPTW [4], LS [28] and LS+TA [28] in all the six problem sets as indicated from Table 4.2. In a matter of fact, DACS 01 is worse, on average by 6.37% for NV and 25.59% for TD, in terms of performance than most of the VRPTW algorithms mentioned in Table 4.3.

For example in MACS-VRPTW, the system brings, after running for 300 to 600 seconds, excellent results in terms of the number of vehicles on the problem sets R1, RC1, R2 and RC2 and these NV results are better, in a way between 6.06% to 13.82%, than the NV results obtained by DACS 01. On clustered problem sets C1 and C2, it is possible for the system DACS 01 to bring the numbers of vehicles as those achieved by MACS-VRPTW.

When it comes to the results in terms of the total of travelled distances on all the six problem sets, MACS-VRPTW is outperforming considerably DACS 01 by 17.01% to 39.61%. Also, if DACS 01 is compared with the algorithms LS and LS+TA run for 100 and 156 seconds respectively, it can be seen that LS and LS+TA manage to bring NV results on R1, RC1, R2 and RC2 that are better by 14.10% to 18.23% and to obtain TD results that are better by 17.53% to 37.91% on all the six problem sets.

Now in terms of the best computed results as in Table 2.10, DACS 01 is noticeably worse, on average by 9.68% for NV and by 24.31% for TD, than MACS-VRPTW [4],

LS+TA [28], HGA+TA [28] and LS+HGA+TA [28]. The worst results computed by DACS 01 are those of the problem sets R1 and RC1. For that, the bad performance of DACS 01 has to be investigated on one of the two problem sets R1 and RC1 so as to try to know what the possible modifications could be done in order to improve the performance. In later sections, the problem set R1 is made as the choice for the future work and experiments.

4.5.2 Does the pheromone trail initialisation and evaporation make the difference?

After the poor performance of DACS 01 in Section 4.5.1, the author has decided to look into why the results gained in Tables 4.2 are so bad when compared to the results of MACS-VRPTW [4], LS [28] and LS+TA [28] and what are the components that are in need to look at and investigate further in order possibly to improve the whole performance.

The first doubts been to see whether the initialisation and the evaporation of the pheromone trails, in the pheromone memory structures of both colonies VMIN and DMIN, are the main reason behind the poor performance of DACS 01 and therefore the author has decided to study and investigate that further on two problem sets, namely, R1-100 of the problem group PG100 in Section 2.2 and R1-200 created by the author. Each of the six problem instances of R1-200 is created from two problem instances of R1-100 and for that R1-200-01 is from R101 and R102, R1-200-02 is from R103 and R104 and so on.

One possible "pheromone trail" initialisation way thought about is to initialise each pheromone trail $\tau_{c_i c_j}$ with a number equal to the number of vehicles allocated to a colony and instead of using the inverse of the distance value, $1/d_{c_i c_j}$, between any two nodes. This way of pheromone initialisation is thought about simply because the initialisation step in each of the two colonies of MACS-VRPTW [4] has the following sentence "initialise pheromone and data structures using V". Note that in VMIN the parametric value V equals the number of vehicles of the best global solution Ψ^{gb} minus one, let us say $V = 13 - 1 = 12$, whereas in DMIN, V equals the number of vehicles of Ψ^{gb} without any reduction, $V = 13$. For example with the new way of pheromone initialisation, each pheromone trail of VMIN is initialised

Table 4.2: Comparison between the average case performances of DACS 01 and other VRPTW algorithms, after three runs, on the problem group PG100. Check Tables B.1 and B.2 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	300 - 400	20.67	1894.59	10.00	852.58	16.67	1987.33	4.00	1759.48	3.00	587.96	4.33	1899.52
SDs		0.58	24.38	0.00	2.52	0.58	27.94	0.00	22.78	0.00	1.86	0.58	112.50
% to SA+LNS [29]	1800	8.77	14.77	0.00	2.85	12.61	21.25	0.00	35.32	0.00	-0.61	8.33	28.22
02 - AVGs	300 - 400	18.67	1876.65	10.00	976.95	14.33	1834.04	4.00	1539.06	3.00	760.52	4.00	1712.56
SDs		0.58	7.07	0.00	17.32	0.58	36.57	0.00	29.06	0.00	41.49	0.00	26.02
% to SA+LNS [29]	1800	9.80	26.28	0.00	17.86	19.44	17.96	11.11	29.78	0.00	23.85	11.11	30.44
03 - AVGs	300 - 400	15.00	1612.17	10.00	1081.06	12.00	1595.49	3.00	1397.47	3.00	808.30	3.33	1534.42
SDs		0.00	6.58	0.00	80.33	0.00	70.93	0.00	32.04	0.00	44.17	0.58	22.05
% to SA+LNS [29]	1800	7.14	32.75	0.00	30.55	9.09	25.88	0.00	41.83	0.00	23.06	11.11	38.36
04 - AVGs	300 - 400	12.00	1318.10	10.00	1164.87	12.00	1472.55	3.00	1133.31	3.00	826.49	3.00	1151.89
SDs		0.00	12.99	0.00	31.36	0.00	56.92	0.00	26.47	0.00	23.32	0.00	51.65
% to SA+LNS [29]	1800	20.00	33.94	0.00	41.23	20.00	28.61	25.00	35.97	0.00	33.36	0.00	35.44
05 - AVGs	300 - 400	15.33	1729.48	10.00	853.48	16.00	1869.65	3.00	1442.10	3.00	605.60	4.33	1872.08
SDs		0.58	65.54	0.00	0.92	0.00	94.15	0.00	21.30	0.00	28.57	0.58	28.35
% to SA+LNS [29]	1800	9.52	23.37	0.00	2.96	17.65	17.80	0.00	37.34	0.00	2.84	8.33	38.27
06 - AVGs	300 - 400	14.33	1627.76	10.00	907.53	14.00	1707.08	3.00	1315.06	3.00	644.58	3.67	1623.11
SDs		0.58	82.92	0.00	38.56	0.00	29.44	0.00	49.16	0.00	6.37	0.58	27.05
% to SA+LNS [29]	1800	19.44	28.15	0.00	9.48	16.67	23.83	0.00	33.94	0.00	6.02	22.22	33.27
07 - AVGs	300 - 400	12.00	1388.84	10.00	983.97	13.00	1616.81	3.00	1255.27	3.00	658.45	4.00	1600.41
SDs		0.00	51.07	0.00	19.84	0.00	24.15	0.00	48.33	0.00	23.48	0.00	58.67
% to SA+LNS [29]	1800	17.65	25.14	0.00	18.70	18.18	31.25	36.36	37.59	0.00	8.34	33.33	43.97
08 - AVGs	300 - 400	11.00	1242.08	10.00	953.45	12.00	1474.28	3.00	1053.09	3.00	658.56	3.00	1270.88
SDs		0.00	65.04	0.00	34.58	0.00	1.39	0.00	25.04	0.00	5.31	0.00	19.53
% to SA+LNS [29]	1800	19.57	27.25	0.00	15.02	20.00	26.87	50.00	38.79	0.00	11.94	0.00	41.11
09 - AVGs	300 - 400	13.67	1527.77	10.00	1017.81	-	-	3.00	1300.85	-	-	-	-
SDs		0.58	57.46	0.00	29.37	-	-	0.00	42.31	-	-	-	-
% to SA+LNS [29]	1800	22.02	26.35	0.00	22.78	-	-	0.00	36.09	-	-	-	-
10 - AVGs	300 - 400	12.33	1386.73	-	-	-	-	3.00	1438.73	-	-	-	-
SDs		0.58	53.87	-	-	-	-	0.00	54.45	-	-	-	-
% to SA+LNS [29]	1800	20.92	23.00	-	-	-	-	0.00	46.41	-	-	-	-
11 - AVGs	300 - 400	13.00	1424.14	-	-	-	-	3.00	1191.27	-	-	-	-
SDs		0.00	47.76	-	-	-	-	0.00	4.65	-	-	-	-
% to SA+LNS [29]	1800	27.45	29.60	-	-	-	-	36.36	31.00	-	-	-	-
12 - AVGs	300 - 400	11.00	1245.41	-	-	-	-	-	-	-	-	-	-
SDs		0.00	31.13	-	-	-	-	-	-	-	-	-	-
% to SA+LNS [29]	1800	10.00	28.16	-	-	-	-	-	-	-	-	-	-
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
SDs		0.08	12.57	0.00	16.54	0.00	20.25	0.00	15.72	0.00	2.57	0.07	23.71
% to SA+LNS [29]	1800	14.97	26.02	0.00	17.92	16.53	23.63	11.64	36.57	0.00	13.85	11.36	35.63
% to MACS-VRPTW [4]	300	13.12	25.55	0.00	17.92	13.36	21.99	6.06	39.08	0.00	17.01	11.36	35.50
% to MACS-VRPTW [4]	600	13.76	25.50	0.00	17.92	13.82	22.77	6.06	39.61	0.00	17.02	11.36	36.11
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.3: Comparison between different VRPTW algorithms on the problem group PG100. The results of each algorithm are averaged over the number of runs done - check Table I.10 for more information.

		R1		C1		RC1		R2		C2		RC2	
Algorithm	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
AKRed [60]	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RTa [26]	430 - 1600	12.83	1208.43	10.00	832.59	12.75	1381.33	3.18	999.63	3.00	595.36	3.62	1207.37
	1300 - 4900	12.58	1202.31	10.00	829.01	12.50	1368.03	3.09	969.29	3.00	590.32	3.62	1155.47
	2600 - 9800	12.58	1197.42	10.00	828.45	12.38	1369.48	3.09	954.36	3.00	590.32	3.62	1139.79
RVNSa [5]	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
RVNSb [5]	4950	11.92	1222.12	10.00	828.38	11.50	1389.58	2.73	975.12	3.00	589.86	3.25	1128.38
ES [2]	1200	12.41	1203.00	10.00	830.00	12.25	1357.00	2.91	955.00	3.00	592.00	3.25	1154.00
ES4 [2]	1200	12.41	1198.00	10.00	829.00	11.88	1356.00	2.82	947.00	3.00	590.00	3.25	1144.00
ES4C [2]	1200	12.41	1201.00	10.00	829.00	12.00	1356.00	2.91	945.00	3.00	590.00	3.25	1140.00
LC03 [33]	3600	12.08	1209.19	10.00	828.38	11.50	1389.22	2.73	963.62	3.00	589.86	3.25	1143.70
LCK05 [32]	3600	11.92	1214.20	10.00	828.38	11.50	1385.30	2.73	954.32	3.00	589.86	3.25	1129.43
BBB [46]	1800	12.17	1251.40	10.00	828.50	11.88	1414.86	2.73	1056.59	3.00	590.06	3.25	1258.15
MACS-VRPTW [4]	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
TD-MACS [53]	100	12.78	1216.38	10.00	830.48	12.63	1406.58	3.15	1002.79	3.00	596.19	3.63	1187.41
	300	12.61	1209.65	10.00	828.82	12.29	1383.83	3.15	984.39	3.00	592.97	3.58	1168.63
	600	12.61	1203.05	10.00	828.41	12.25	1374.49	3.12	977.15	3.00	591.06	3.54	1155.86
	1200	12.61	1199.36	10.00	828.38	12.13	1373.18	3.09	972.31	3.00	590.49	3.46	1156.77
	1800	12.61	1196.27	10.00	828.38	12.04	1372.71	3.09	966.95	3.00	590.49	3.38	1155.74
HGA [28]	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28]	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28]	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28]	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48
KPS [15]	2900	12.67	1200.33	10.00	830.75	12.12	1388.15	3.00	966.56	3.00	592.29	3.38	1133.42
HG05a [64]	1038	12.17	1217.79	10.00	832.24	11.83	1383.70	2.85	959.84	3.00	592.12	3.25	1158.05
RTb [26]	430 - 1600	13.00	1225.62	10.00	838.93	13.00	1418.58	3.62	996.03	3.10	616.44	4.18	1249.80
	1300 - 4900	12.92	1223.74	10.00	838.00	12.88	1417.92	3.62	992.48	3.00	611.25	4.18	1245.06
	2600 - 9800	12.92	1222.36	10.00	836.87	12.77	1418.36	3.62	990.09	3.00	610.28	4.18	1244.77
TB [31]	1877 - 3372	12.64	1233.88	10.00	830.41	12.08	1404.59	3.00	1046.56	3.00	592.75	3.38	1248.34
	5632 - 10116	12.39	1230.48	10.00	828.59	12.00	1387.01	3.00	1029.65	3.00	591.14	3.38	1220.28
	11264 - 20232	12.33	1220.35	10.00	828.45	11.90	1381.31	3.00	1013.35	3.00	590.91	3.38	1198.63
HGA+EA [27]	903	12.50	1223.40	-	-	12.20	1390.80	3.00	989.50	-	-	3.50	1205.40
SA+LNS [29]	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MSLS2 [17]	132	12.11	1232.18	10.00	828.44	11.70	1403.34	2.78	982.47	3.00	590.31	3.25	1162.43
MSLS+TA2 [17]	162	12.11	1218.56	10.00	828.38	11.70	1389.68	2.78	967.19	3.00	589.86	3.25	1146.61
VGA1 [7]	13 - 87	13.73	1190.69	10.00	828.40	13.27	1363.61	5.61	885.99	3.00	589.93	6.43	1014.06
VGA2 [7]	13 - 109	13.70	1189.28	10.00	828.38	13.34	1362.38	5.64	886.60	3.00	589.95	6.44	1013.72
LSH [67]	1200	12.30	1247.23	10.00	834.44	12.04	1406.46	2.88	1025.69	3.00	591.03	3.34	1205.12
TH [25]	100 - 3749	13.20	1280.13	10.00	889.31	12.81	1467.33	3.23	1105.82	3.14	688.02	3.63	1271.05
PB [41]	600 - 2460	12.58	1296.80	10.00	838.01	12.13	1446.20	3.00	1117.70	3.00	589.93	3.38	1360.57

with 12 and each pheromone trail of DMIN is set with 13.

After changing the way of pheromone initialisation as described above, the new DACS system is tested using five different pheromone evaporation values, $\rho = 0.0$, 0.1, 0.5, 0.8 or 1.0 in order to have a better understanding of its behavior. From the results in Tables B.3 and B.4, it can be recognised that the results are rather bizarre and strange and as if they say when there is no evaporation, $\rho = 0.0$, the best results are gained in terms of the number of vehicles (13.75 for R1-100 and 23.17 for R1-200). For that, this issue is going to be investigated further in Section 4.5.3.

Additionally, it can be seen from Tables B.3 and B.4 that the pheromone initialisation of $1/d_{c_ic_j}$ is significantly better than (in terms of the total of travelled distances, on average by 8.22%) the new way of pheromone initialisation using V-1 and V in VMIN and DMIN respectively and as a result the latest way of pheromone trail initialization is discarded.

4.5.3 What pheromone trail re-initialisations can do?

Because of the controversial issue regarding the pheromone initialisation using V-1 and V in VMIN and DMIN respectively and the pheromone evaporation when it equals zero in Section 4.5.2, the issue of pheromone initialisation and evaporation is thought about for the third time. Of course, this controversial issue has led the author to manage a careful investigation for two reasons.

The first reason is to let the DACS system behave in performance in the same way as the MACS-VRPTW system in [4] and to bring results that are also comparable with the results of other VRPTW algorithms like LS [28] and LS+TA [28]. The second reason is to answer this controversial question: Why Gambardella in [4] has used the value 0.1 of the evaporation parameter in MACS-VRPTW? For that, the references in [4] [76] were reviewed to check for any sentence that refers to the way of initialising the pheromone trail structures in VMIN and DMIN but unfortunately it was not clear which way it is needed to initialise each pheromone trail $\tau_{c_ic_j}$ with and whether the term τ_o is the possibility for doing that.

Then after reviewing about the pheromone trail initialisation and re-initialisation in an ACO technique in [76] that is used for solving Quadratic Assignment Problems QAPs, the conclusion was that the pheromone re-initialisation in addition to the initialisation itself might lead into improving the performance of the DACS system.

Therefore, two major updates are made to the DACS system. The first update is to initialise each pheromone trail of the pheromone memory structures of the two colonies VMIN and DMIN with $1/(n.J_{\Psi}^{gb})$, where J_{Ψ}^{gb} is the total of travelled distances of the best global solution Ψ^{gb} found so far and n is the number of visited nodes - depot or vehicle nodes + customer nodes. The second update is to re-initialise the pheromone trails once a new best global solution, in terms of the number of vehicles in particular, is found. Here, the re-initialisation of the pheromone trails happens, as described above, but using the number of vehicles and the total of travelled distances of the new best global solution.

After making the two updates mentioned above, the new DACS system is tested on the problem set R1-100 of the problem group PG100 using the evaporation values, $\rho = 0.0$, 0.1, 0.5, 0.8 and 1.0. It can be seen from the results obtained in Table B.5 by the pheromone trail initialisation and re-initialisation with $1/(n.J_{\Psi}^{gb})$ is that the NV results are significantly better, on average by -3.31%, than the results gained in Table 4.2 by DACS 01 that initialises only the pheromone trails with $1/d_{c_i c_j}$.

Despite the significant improvement gained because of making the DACS system initialise and reinitialise with $1/(n.J_{\Psi}^{gb})$, however such way of pheromone trail initialisation and re-initialisation has not made the new system in any way behave in performance in the same way as the system MACS-VRPTW in [4] and to bring results that are comparable to those of the algorithms LS [28] and LS+TA [28].

Nonetheless, it can be recognised from Table B.5 that the results of the new DACS system with the evaporation value equal to 0.1 is the best on average, by -4.14% for NV and -0.09% for TD, over all the results of the DACS systems with other evaporation values like 0.0, 0.5, 0.8 and 1.0. But, such results suggest as well that there is not any significant difference in performance between the various DACS systems that use the five different evaporation values. Therefore in order to avoid any doubt about whether our DACS system is behaving or not in performance as MACS-VRPTW in [4], the parametric values of m, ρ, β and p_E in Table 4.1 are kept from now onwards with the same values used in MACS-VRPTW [4].

4.5.4 Is it about reconfiguring the cycle phases?

Despite the fact of improving the performance of the DACS system because of initialising and re-initialising the pheromone trails with $1/(n.J_{\Psi}^{gb})$ as indicated from

Table B.5, however that system is not behaving in performance as well as the performance of the MACS-VRPTW system [4] and many other algorithms like LS [28], LS+TA [28]...etc.

For that, one thing thought about is to try to reconfigure the cycles of the coordinator DACS, the colonies VMIN and DMIN and the XCHNG local search by changing the numbers of iterations used, as in Table 4.4, as limits to stop such components. In the new DACS system, 5 iterations is used for the cycle of the coordinator DACS, 4 iterations is used for each of the two cycles of VMIN and DMIN and 500 iterations is used for the cycle of the XCHNG local search. These limits, mentioned above, are used also in the DACS systems tested in Sections 4.5.5 and 4.5.6.

One of the main reasons behind choosing the number of iterations, as a stopping criterion, is that it was not known what are the halting criteria should be used to stop the work of each of components mentioned above as in MACS-VRPTW [4]. After reconfiguring the cycles of the components mentioned above, it can be seen from Table B.6 that the TD result 1505.82 on the problem set R1 of the problem group PG100, is significantly better on average by -1.03% but it is not really good enough when compared with the TD result 1521.50, of the DACS system that initialises and re-initialises the pheromone trails with $1/(n.J_{\Psi}^{gb})$.

Consequently also, reconfiguring the cycles of the components mentioned above has not made any real difference in performance to be as well as the systems MACS-VRPTW system [4], LS [28], LS+TA [28] and so on.

Table 4.4: The parametric values used in the DACS system with reconfigured cycles.

Number of artificial ants (m)	10
Evaporation parameter (ρ)	0.1
Heuristic value (β)	1
Transition mode probability (p_E)	0.9
Maximum iterations of the cycle phase Of the system DACS 01	5
Maximum iterations of the cycle phase of each colony (VMIN or DMIN)	4
Maximum iterations of the XCHNG local search	500

4.5.5 What about threading the colonies?

Despite the fact that the researchers of the MACS-VRPTW system [4] have not mentioned clearly whether concurrent threads are used or not, MACS-VRPTW may be a multi-threaded system and therefore the issue of concurrent programming and its ideas, like multithreaded algorithms and synchronization, are reviewed here in order to see the effects of threading the DACS system and its two colonies on the performance of the system as a whole.

In Java, a thread is a single sequential flow of control within a program and threads might be used to work on a task each. To let two or more threaded algorithms of a multithreaded system run concurrently, the PC machine must have a dual processor. Otherwise, the multithreaded system might NOT be considered as a system with two or more threads running concurrently. In this case, it may need a scheduling algorithm to yield the process of running from a thread to another, if the PC system has a single CPU.

Because our PC uses a single CPU, the deterministic scheduling algorithm (fixed priority scheduling) of Java is used in the system Threaded-DACS to help in yielding the process of running from a threaded colony to another. Now in Threaded-DACS, the system consists of two threaded colonies of VMIN and DMIN that have the same level of priority - 10 is the priority level of each thread of the two threaded colonies. To make the two colony threads of Threaded-DACS run concurrently, the Java instance method yield() of threads is used at the end of the cycle of a threaded colony especially after the ants build their solutions and applying the component of pheromone global updating. The reason is to ease yielding the process of running from a colony thread to another.

Now, whether threads can run concurrently or not, threads might or might not share resources such as using the instances variables or methods of an object. In order to avoid accessing the shared resources by two threaded colonies, like VMIN and DMIN in a Threaded-DACS system, in a conflicting way at the same time, the instance methods shared by these two threaded colonies must be declared with the Java keyword synchronized. Otherwise, some sort of illegal exceptions may come out as a result of not prefixing the Java keyword synchronized.

After threading the colonies, it can be seen from Table B.7 that the system Threaded-DACS has improved, on average, the TD results significantly by -0.94%

on the problem set R1 of the problem group PG100. But, threading the system and its colonies has not made the performance in any way closer to the performance of the MACS-VRPTW system [4] and the other VRPTW algorithms in the literature. Also, one thing recognized is that it lacks consistency in getting the same kind of performance at every time such system is run and this is indicated from the SD value in terms of NV (where SD = 0.15), which is greater than that of the DACS system where SD = 0.09. Subsequently, adding threads into DACS is disregarded because it is unnecessary and makes no real difference.

4.5.6 Is it about changing the move operator used?

One of the objectives in this research is to check the effects of using different types of local searches on improving the results and the performance of the DACS system and to discover whether a small change to the work of the local search would make any difference. Another objective is to see what are the effects of using either inter or intra-route move operators, applied individually (like 2-Opt, 3-Opt, Or-Opt and relocation [9] [7]), on the decision making of the ants.

For example, would any of these different "individually applied" move operators make the decision making of the ants more precise and the results and the performance of the systems involved much better? As a result, different types of local searches that use one move operator each are built in the process of implementation to test the five different move operators (2-Opt*, Relocate, Exchange, Or-Opt1 and Or-Opt2 [9] [7] [31]), explained below in detail, as substitutes to the move operator of the XCHNG local search.

O1- The 2-Opt* move operator replaces two old edges with two new edges in an X-like fashion as in Figure 2.2. Note that the local search that uses 2-Opt* happens between two routes chosen randomly and therefore the 2-Opt* used here is an inter-route improvement operator. The 2-Opt* is different from the 2-Opt move operator in the sense that it preserves the sequence order of the customers in each of the two routes after replacing two old edges with two new edges.

O2- The relocate move operator moves one customer from one route to another as in Figure 2.4 and therefore it is an inter-route improvement operator. It

selects two routes randomly and later it chooses again in a random way one of the two routes. Then, it chooses randomly one customer to be relocated from the randomly chosen route to a random location in the other route.

O3- The exchange move operator exchanges two customers between two routes as in Figure 2.6 and therefore it is an inter-route improvement operator. It starts by choosing two routes randomly. After that, one customer is selected randomly from a route in order to be exchanged with another customer, selected randomly also, from the other route.

O4- The Or-Opt1 move operator moves a continuous segment of customers as in Figure 4.10 from one position on a route to another position on the same route. It is an intra-route improvement operator and it starts by selecting a route randomly. Later, it chooses two random indices on the chosen route. The lowest index of the two indices is considered as the start of the segment to be removed and the highest index is considered as the end of the segment. In this operator, there is not any restriction on the length of the segment to be cut out. Next after cutting out the segment of customers, the chosen route is reduced to a number of nodes greater than or equal to one. Afterwards, the Or-Opt1 selects a random position of the reduced route in order to relocate the segment cut out into that random position.

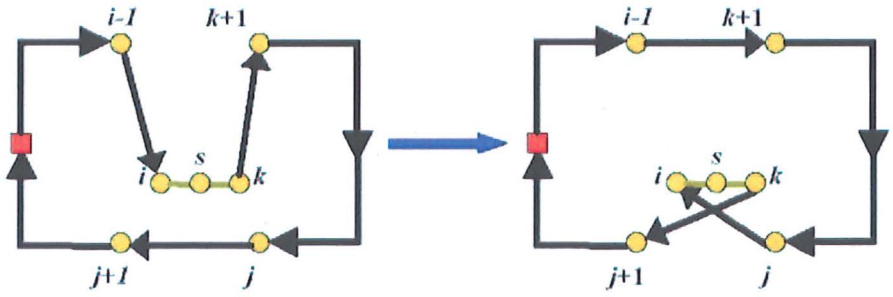


Figure 4.10: Or-Opt1 operator for intra-route improvements.

O5- The Or-Opt2 move operator moves a continuous segment of customers as in Figure 4.11 from one position on a route to another position on another route. It is an inter-route improvement operator in which two routes are selected at random. Subsequently, one of the two routes is randomly selected for removing

a segment of customers while the other route is selected for hosting the removed segment. The length of the segment to be cut out is chosen randomly and is restricted to be between 1 and 3 customers. In order to cut out a segment from a route, two random indices are chosen on the route. The lowest index of the two indices is considered as the start of the segment and the highest index is considered as the end of the segment. Later, a random position on the hosting route is selected randomly to embed the segment to be cut out from the route, chosen for removing that segment.

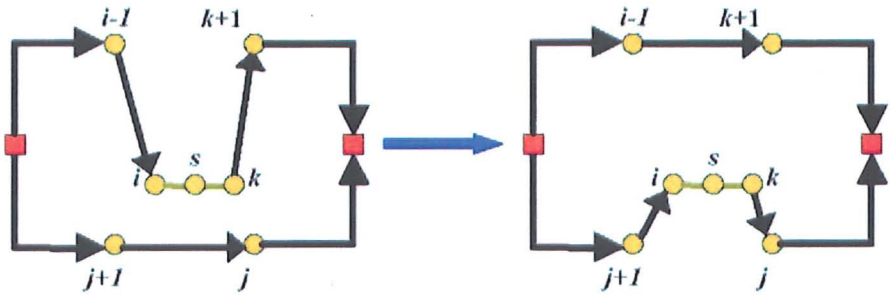


Figure 4.11: Or-Opt2 operator for inter-route improvements.

The question that comes possibly to the mind is that why the local searches that use the five neighbourhood operators described above are taken into consideration in this experimental work. The reason is that the researchers of MACS-VRPTW in [4] have said that the type of the local search procedure, they are using, has similar moves to the CROSS exchanges mentioned in [31]. In addition, some researchers [31] declares that the move operator of CROSS exchanges is a generalization of the 2-Opt* and Or-Opt move operators. Consequently, it is believed the same thing is true for the move operators explained above and it is worthwhile to check what each move can do when it is applied individually.

As indicated from Table B.8, adding a new move operator to work on its own individually instead of another one has not made the performance of the DACS system in any way near from the performance of the MACS-VRPTW system [4] and the other VRPTW algorithms in the literature. Additionally, the change of the move operator of XCHNG into a new move operator has not made the ants more accurate enough in their decision-making.

The local search of 2-Opt in Table B.8 is the only one, which has improved the TD result on average by -1.49% on the problem set R1 of the problem group PG100. But, its main obstacle is that it is not good enough for it to work on its own without the help of other move operators. For that, the idea of letting inter or intra-route move operators work on their own individually is ignored and as a result the idea is now to try to join two or more move operators to work together as intra and inter-route improvement heuristics at the same time - For more information, check Section 4.6.1.

4.6 Different configurations of the double ant colony system - DACS 02

Despite the experimental campaign done as in Sections 4.5.2, 4.5.3, 4.5.4, 4.5.5 and 4.5.6 and the fact that the results and the performance of the DACS system have improved as in Table B.5 in comparison to the results and the performance of the original system DACS 01 in Section 4.5.1, however the MACS-VRPTW system [4] and other VRPTW algorithms like LS [28] and LS+TA [28] are still outperforming the DACS system.

Consequently in this section, an experimental work on a system called DACS 02 is done and this system has all the features described in Section 4.4 but with the exception of four components, which are mentioned below. The DACS 02 system applies the parametric values in Table 4.5, which are used as well by all DACS systems tested onwards.

C1- The push forward and push backward strategy PFPBS.

C2- The hybrid local search HLS.

C3- The variant of the 2-Opt move. It is not used in the local search of quadruple moves.

C4- The candidate lists. They are not used in the probabilistic-state transition component and the insertion procedure.

For that at the start, the effects of using a local search of triple moves in DACS 02 are discussed in Sections 4.6.1 and 4.6.2 on the problem groups PG100, PG200

Table 4.5: The parametric values used in the new DACS systems onwards.

Number of ants (m)	10
Evaporation parameter (ρ)	0.1
Heuristic value (β)	1
Transition mode probability (p_E)	0.9
CPU time limit in seconds	100, 300, 400, 600, 1200, 1800, 2400 or 4800
Maximum iterations of the cycle phase of each colony (VMIN or DMIN)	1
Maximum iterations of the TripleMovesLS local search	until no improvement can be found

and PG400. Also, the search ways of ants in Section 4.6.3 and the initialisation techniques used to get the first best global solution in Section 4.6.4 and their effects are discussed. Furthermore, the kinds of candidate lists that should be used by the ants' routing builder are considered and investigated in Section 4.6.5. Finally in Section 4.6.6, some experiments with the pheromone updating and local search moves near the depot nodes are talked about.

4.6.1 What is the effect of using a local search of triple moves?

As in Sections 4.5.1 and 4.5.6, letting any move operator work, on its own individually in a local search and as either an inter or an intra-route improvement heuristic, has resulted in making the system miss many search points in the search space of solutions already available. Also, using the random strategy, in relocating and swapping customers or replacing old edges with new ones, has made the efficiency, performance and results of the DACS systems implemented so far not to be as good as the performance and results of the other VRPTW algorithms in the literature.

Moreover, such individually applied move operators suffer from the fact that they are not using the candidate lists, which may help in limiting the search space in areas that are more likely to lead to good quality solutions. In addition, the way, in which such move operators are configured individually with other elements in the

local search, makes them unable of emptying routes with small numbers of visited customers and relocating such customers into other routes.

Therefore, it might be worthwhile to try two or more move operators, which are inter and intra-route improvement heuristics at the same time, to work together within a local search and with the help of candidate lists as explained in Section 4.4.8 rather than depending only on one move operator as described earlier.

In order to check the effect what is described in the previous paragraph, a local search of triple moves (M1 to M3 in Section 4.4.8) is tested on the problem group PG100 for three runs at six different amounts of CPU time - 100, 300, 400, 600, 1200 and 1800. From Tables 4.6, 4.7 and 4.8, it can be seen that after 300 to 400 seconds the results and the performance of DACS 02 have improved dramatically, on average by -17.04% for TD and by -1.67% for NV, on all the six problem sets of PG100 (except C2) when compared with that of the system DACS 01. The best and worst-case performances of DACS 02 are much better as well.

In comparison to the other VRPTW algorithms (like MACS-VRPTW, SA+LNS, LS, LS+TA...etc) in Table 4.3, DACS 02 has on average a percentage of deviation, equal to 4.38% for NV and 3.87% for TD, which is a lot better than that of DACS 01 - 6.37% for NV and 25.59% for TD. For instance as indicated from Tables 4.6, DACS 02 is a lot closer to MACS-VRPTW and the percentage of deviation, after 300 to 400 seconds, is on average equal to 5.37% for NV and 4.39% for TD. Also when it comes to SA+LNS run for 1800 seconds, DACS 02 after 300 to 400 seconds, as in Tables 4.7 and 4.8, is closer to it on average by 7.08% for NV and 3.92% for TD.

Also from Tables 4.6 and 4.9, it can be understood when DACS 02 is run for a certain amount of short CPU time like 300 to 400 seconds or for a certain amount of long CPU time like 1200 to 1800 seconds, it does not mean that the system is going to bring out the same kind of behaviour described in such tables. However, if DACS 02 is run for longer CPU times such as 1200 and 1800 seconds on each problem instance, the system is often expected to get even further improved results and performance.

Table 4.6: Comparison between the average case performances of the algorithms DACS 02 and DACS 01, after three runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
DACS 02 - AVGs	100	13.44	1289.43	10.00	860.55	13.21	1454.84	3.27	1094.37	3.29	668.60	3.83	1330.21
SDs		0.05	13.69	0.00	11.28	0.07	8.58	0.09	25.80	0.07	9.32	0.19	27.46
% to DACS 01	300 - 400	-4.54	-15.33	0.00	-11.91	-3.94	-14.15	2.86	-18.80	9.72	-3.63	3.37	-15.97
% to MACS-VRPTW [4]	100	7.13	6.14	0.00	3.88	6.01	4.25	7.30	12.59	9.72	12.71	13.41	11.61
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
SDs		0.14	4.18	0.00	5.18	0.13	10.01	0.00	3.88	0.07	17.81	0.14	12.63
% to DACS 01	300 - 400	-5.92	-18.24	0.00	-12.38	-6.36	-16.08	0.00	-21.07	2.78	-11.94	-1.12	-19.57
% to MACS-VRPTW [4]	300	6.43	2.65	0.00	3.33	6.14	2.38	6.06	9.77	2.78	3.03	10.11	8.99
DACS 02 - AVGs	400	13.31	1247.99	10.00	844.26	12.83	1427.10	3.18	1038.20	3.08	600.10	3.71	1246.69
SDs		0.10	9.33	0.00	8.24	0.14	6.44	0.00	10.25	0.07	7.35	0.07	23.98
% to DACS 01	300 - 400	-5.52	-18.05	0.00	-13.57	-6.67	-15.79	0.00	-22.97	2.78	-13.51	0.00	-21.25
% to MACS-VRPTW [4]	300	6.87	2.89	0.00	1.92	5.80	2.73	6.06	7.13	2.78	1.20	11.36	6.71
DACS 02 - AVGs	600	13.17	1244.77	10.00	860.31	12.71	1412.59	3.18	1020.19	3.00	593.24	3.71	1199.88
SDs		0.00	4.56	0.00	18.15	0.14	12.86	0.00	5.64	0.00	1.93	0.07	9.06
% to DACS 01	300 - 400	-6.51	-18.26	0.00	-11.93	-7.58	-16.64	0.00	-24.31	0.00	-14.50	0.00	-24.21
% to MACS-VRPTW [4]	600	6.35	2.59	0.00	3.85	5.20	2.33	6.06	5.68	0.00	0.06	11.36	3.16
DACS 02 - AVGs	1200	13.14	1231.12	10.00	847.72	12.58	1414.30	3.15	994.12	3.00	592.53	3.58	1173.25
SDs		0.17	13.07	0.00	5.51	0.07	9.70	0.05	7.31	0.00	2.14	0.07	13.41
% to DACS 01	300 - 400	-6.71	-19.16	0.00	-13.22	-8.48	-16.54	-0.95	-26.24	0.00	-14.60	-3.37	-25.89
% to MACS-VRPTW [4]	1200	6.13	1.61	0.00	2.33	5.21	2.07	5.05	3.33	0.00	0.08	7.61	1.70
DACS 02 - AVGs	1800	13.00	1225.67	10.00	835.91	12.75	1404.20	3.18	983.86	3.00	591.92	3.54	1157.52
SDs		0.14	11.34	0.00	5.64	0.00	9.93	0.00	7.24	0.00	1.07	0.07	5.61
% to DACS 01	300 - 400	-7.69	-19.51	0.00	-14.43	-7.27	-17.14	0.00	-27.00	0.00	-14.69	-4.49	-26.88
% to MACS-VRPTW [4]	1800	5.01	1.23	0.00	0.91	6.96	1.16	6.06	2.45	0.00	0.01	6.36	0.72
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.7: Comparison between the average case performances of DACS 02 and other VRPTW algorithms, after three runs of 300 seconds, on the problem group PG100. Check Tables C.1 and C.2 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	300	19.67	1664.04	10.00	828.94	15.33	1750.18	4.00	1453.22	3.00	591.56	4.00	1629.50
SDs		0.58	6.57	0.00	0.00	0.58	92.76	0.00	69.12	0.00	0.00	0.00	99.57
% to DACS 01	300 - 400	-4.84	-12.17	0.00	-2.77	-8.00	-11.93	0.00	-17.41	0.00	0.61	-7.69	-14.22
02 - AVGs	300	18.00	1518.09	10.00	904.26	14.00	1531.98	4.00	1255.59	3.00	591.56	4.00	1403.23
SDs		0.00	13.62	0.00	79.18	0.00	16.85	0.00	31.07	0.00	0.00	0.00	14.95
% to DACS 01	300 - 400	-3.57	-19.11	0.00	-7.44	-2.33	-16.47	0.00	-18.42	0.00	-22.22	0.00	-18.06
03 - AVGs	300	14.00	1266.08	10.00	915.84	12.00	1366.34	3.00	1124.07	3.00	600.92	3.67	1252.26
SDs		0.00	22.85	0.00	32.10	0.00	17.35	0.00	18.37	0.00	8.70	0.58	81.01
% to DACS 01	300 - 400	-6.67	-21.47	0.00	-15.28	0.00	-14.36	0.00	-19.56	0.00	-25.66	10.00	-18.39
04 - AVGs	300	10.33	1057.89	10.00	895.12	11.00	1207.06	3.00	864.69	3.67	749.48	3.00	959.46
SDs		0.58	23.45	0.00	32.29	0.00	7.79	0.00	6.09	0.58	151.05	0.00	24.24
% to DACS 01	300 - 400	-13.89	-19.74	0.00	-23.16	-8.33	-18.03	0.00	-23.70	22.22	-9.32	0.00	-16.71
05 - AVGs	300	14.67	1439.92	10.00	828.94	15.00	1603.31	3.00	1153.54	3.00	588.88	4.00	1502.09
SDs		0.58	31.33	0.00	0.00	0.00	7.91	0.00	14.82	0.00	0.00	0.00	52.10
% to DACS 01	300 - 400	-4.35	-16.74	0.00	-2.88	-6.25	-14.25	0.00	-20.01	0.00	-2.76	-7.69	-19.76
06 - AVGs	300	13.00	1308.71	10.00	843.39	12.67	1430.74	3.00	1058.76	3.00	588.49	3.67	1255.22
SDs		0.00	1.71	0.00	25.03	0.58	24.96	0.00	16.71	0.00	0.00	0.58	20.38
% to DACS 01	300 - 400	-9.30	-19.60	0.00	-7.07	-9.52	-16.19	0.00	-19.49	0.00	-8.70	0.00	-22.67
07 - AVGs	300	11.67	1138.78	10.00	828.94	12.00	1299.51	3.00	963.50	3.00	588.29	4.00	1202.62
SDs		0.58	19.36	0.00	0.00	0.00	16.09	0.00	9.97	0.00	0.00	0.00	37.72
% to DACS 01	300 - 400	-2.78	-18.01	0.00	-15.76	-7.69	-19.63	0.00	-23.24	0.00	-10.66	0.00	-24.86
08 - AVGs	300	10.67	1006.74	10.00	828.94	11.00	1188.55	3.00	816.32	3.00	588.32	3.00	982.29
SDs		0.58	19.46	0.00	0.00	0.00	23.02	0.00	5.74	0.00	0.00	0.00	10.63
% to DACS 01	300 - 400	-3.03	-18.95	0.00	-13.06	-8.33	-19.38	0.00	-22.48	0.00	-10.67	0.00	-22.71
09 - AVGs	300	12.00	1233.19	10.00	828.94	-	-	3.00	1058.73	-	-	-	-
SDs		0.00	44.73	0.00	0.00	-	-	0.00	39.29	-	-	-	-
% to DACS 01	300 - 400	-12.20	-19.28	0.00	-18.56	-	-	0.00	-18.61	-	-	-	-
10 - AVGs	300	12.00	1152.52	-	-	-	-	3.00	1056.05	-	-	-	-
SDs		0.00	24.00	-	-	-	-	0.00	11.41	-	-	-	-
% to DACS 01	300 - 400	-2.70	-16.89	-	-	-	-	0.00	-26.60	-	-	-	-
11 - AVGs	300	12.00	1145.14	-	-	-	-	3.00	897.01	-	-	-	-
SDs		0.00	11.24	-	-	-	-	0.00	37.01	-	-	-	-
% to DACS 01	300 - 400	-7.69	-19.59	-	-	-	-	0.00	-24.70	-	-	-	-
12 - AVGs	300	11.00	1010.30	-	-	-	-	-	-	-	-	-	-
SDs		0.00	12.55	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-18.88	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
SDs		0.14	4.18	0.00	5.18	0.13	10.01	0.00	3.88	0.07	17.81	0.14	12.63
% to DACS 01	300 - 400	-5.92	-18.24	0.00	-12.38	-6.36	-16.08	0.00	-21.07	2.78	-11.94	-1.12	-19.57
% to SA+LNS [29]	1800	8.16	3.04	0.00	3.33	9.11	3.76	11.64	7.79	2.78	0.25	10.11	9.09
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.8: Comparison between the average case performances of DACS 02 and other VRPTW algorithms, after three runs of 400 seconds, on the problem group PG100. Check Tables C.3 and C.4 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	400	19.67	1671.25	10.00	828.94	15.33	1730.44	4.00	1340.90	3.00	591.56	4.00	1623.79
SDs		0.58	12.86	0.00	0.00	0.58	42.07	0.00	10.43	0.00	0.00	0.00	113.66
% to DACS 01	300 - 400	-4.84	-11.79	0.00	-2.77	-8.00	-12.93	0.00	-23.79	0.00	0.61	-7.69	-14.52
02 - AVGs	400	18.00	1503.95	10.00	843.18	14.00	1511.26	4.00	1239.77	3.00	591.56	4.00	1395.92
SDs		0.00	14.18	0.00	24.66	0.00	5.15	0.00	34.00	0.00	0.00	0.00	45.70
% to DACS 01	300 - 400	-3.57	-19.86	0.00	-13.69	-2.33	-17.60	0.00	-19.45	0.00	-22.22	0.00	-18.49
03 - AVGs	400	14.00	1279.04	10.00	922.92	12.00	1372.33	3.00	1070.75	3.00	591.17	3.67	1191.35
SDs		0.00	28.47	0.00	30.32	0.00	22.35	0.00	44.05	0.00	0.00	0.58	50.09
% to DACS 01	300 - 400	-6.67	-20.66	0.00	-14.63	0.00	-13.99	0.00	-23.38	0.00	-26.86	10.00	-22.36
04 - AVGs	400	10.33	1064.66	10.00	858.57	11.00	1246.76	3.00	872.09	3.67	672.51	3.00	929.03
SDs		0.58	35.02	0.00	29.68	0.00	61.88	0.00	7.84	0.58	58.77	0.00	21.02
% to DACS 01	300 - 400	-13.89	-19.23	0.00	-26.29	-8.33	-15.33	0.00	-23.05	22.22	-18.63	0.00	-19.35
05 - AVGs	400	15.00	1449.62	10.00	828.94	15.00	1618.24	3.00	1132.91	3.00	588.88	4.00	1449.10
SDs		0.00	41.68	0.00	0.00	0.00	10.72	0.00	15.82	0.00	0.00	0.00	26.48
% to DACS 01	300 - 400	-2.17	-16.18	0.00	-2.88	-6.25	-13.45	0.00	-21.44	0.00	-2.76	-7.69	-22.59
06 - AVGs	400	13.00	1314.18	10.00	828.94	12.67	1423.92	3.00	1066.58	3.00	588.49	4.00	1259.32
SDs		0.00	17.93	0.00	0.00	0.58	6.80	0.00	10.47	0.00	0.00	0.00	8.14
% to DACS 01	300 - 400	-9.30	-19.26	0.00	-8.66	-9.52	-16.59	0.00	-18.89	0.00	-8.70	9.09	-22.41
07 - AVGs	400	11.67	1122.12	10.00	828.94	11.67	1317.52	3.00	934.67	3.00	588.29	4.00	1164.10
SDs		0.58	15.55	0.00	0.00	1.15	28.67	0.00	19.78	0.00	0.00	0.00	19.41
% to DACS 01	300 - 400	-2.78	-19.20	0.00	-15.76	-10.26	-18.51	0.00	-25.54	0.00	-10.66	0.00	-27.26
08 - AVGs	400	10.33	1044.32	10.00	828.94	11.00	1196.29	3.00	801.97	3.00	588.32	3.00	960.93
SDs		0.58	44.04	0.00	0.00	0.00	57.84	0.00	12.09	0.00	0.00	0.00	50.75
% to DACS 01	300 - 400	-6.06	-15.92	0.00	-13.06	-8.33	-18.86	0.00	-23.85	0.00	-10.67	0.00	-24.39
09 - AVGs	400	13.00	1220.88	10.00	828.94	-	-	3.00	1000.18	-	-	-	-
SDs		0.00	15.11	0.00	0.00	-	-	0.00	11.23	-	-	-	-
% to DACS 01	300 - 400	-4.88	-20.09	0.00	-18.56	-	-	0.00	-23.11	-	-	-	-
10 - AVGs	400	11.67	1179.42	-	-	-	-	3.00	1066.26	-	-	-	-
SDs		0.58	31.15	-	-	-	-	0.00	33.02	-	-	-	-
% to DACS 01	300 - 400	-5.41	-14.95	-	-	-	-	0.00	-25.89	-	-	-	-
11 - AVGs	400	12.00	1129.21	-	-	-	-	3.00	894.12	-	-	-	-
SDs		0.00	5.83	-	-	-	-	0.00	8.89	-	-	-	-
% to DACS 01	300 - 400	-7.69	-20.71	-	-	-	-	0.00	-24.94	-	-	-	-
12 - AVGs	400	11.00	997.21	-	-	-	-	-	-	-	-	-	-
SDs		0.00	8.93	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-19.93	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	400	13.31	1247.99	10.00	844.26	12.83	1427.10	3.18	1038.20	3.08	600.10	3.71	1246.69
SDs		0.10	9.33	0.00	8.24	0.14	6.44	0.00	10.25	0.07	7.35	0.07	23.98
% to DACS 01	300 - 400	-5.52	-18.05	0.00	-13.57	-6.67	-15.79	0.00	-22.97	2.78	-13.51	0.00	-21.25
% to SA+LNS [29]	1800	8.62	3.28	0.00	1.92	8.76	4.11	11.64	5.20	2.78	-1.52	11.36	6.81
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.9: Comparison between the average case performances of DACS 02 and other VRPTW algorithms, after three runs of 1800 seconds, on the problem group PG100. Check Tables C.5 and C.6 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.33	1692.56	10.00	828.94	15.33	1707.58	4.00	1288.97	3.00	591.56	4.00	1500.98
SDs		0.58	31.86	0.00	0.00	0.58	35.55	0.00	24.50	0.00	0.00	0.00	14.64
% to DACS 01	300 - 400	-6.45	-10.66	0.00	-2.77	-8.00	-14.08	0.00	-26.74	0.00	0.61	-7.69	-20.98
02 - AVGs	1800	17.33	1499.39	10.00	828.94	13.67	1512.20	4.00	1130.58	3.00	591.56	4.00	1219.33
SDs		0.58	15.25	0.00	0.00	0.58	19.85	0.00	21.93	0.00	0.00	0.00	25.22
% to DACS 01	300 - 400	-7.14	-20.10	0.00	-15.15	-4.65	-17.55	0.00	-26.54	0.00	-22.22	0.00	-28.80
03 - AVGs	1800	14.00	1236.68	10.00	873.97	11.67	1341.39	3.00	1011.98	3.00	591.17	3.00	1129.00
SDs		0.00	6.83	0.00	30.58	0.58	23.88	0.00	10.72	0.00	0.00	0.00	13.81
% to DACS 01	300 - 400	-6.67	-23.29	0.00	-19.16	-2.78	-15.93	0.00	-27.59	0.00	-26.86	-10.00	-26.42
04 - AVGs	1800	11.00	1020.10	10.00	846.65	11.00	1184.33	3.00	825.01	3.00	605.92	3.00	866.49
SDs		0.00	7.79	0.00	20.46	0.00	8.40	0.00	11.46	0.00	10.50	0.00	3.62
% to DACS 01	300 - 400	-8.33	-22.61	0.00	-27.32	-8.33	-19.57	0.00	-27.20	0.00	-26.69	0.00	-24.78
05 - AVGs	1800	14.67	1421.98	10.00	828.94	14.67	1600.63	3.00	1060.18	3.00	588.88	4.00	1359.47
SDs		0.58	42.11	0.00	0.00	0.58	27.11	0.00	20.28	0.00	0.00	0.00	23.62
% to DACS 01	300 - 400	-4.35	-17.78	0.00	-2.88	-8.33	-14.39	0.00	-26.48	0.00	-2.76	-7.69	-27.38
06 - AVGs	1800	13.00	1312.56	10.00	828.94	12.67	1461.58	3.00	985.80	3.00	588.49	3.33	1202.83
SDs		0.00	20.93	0.00	0.00	0.58	64.09	0.00	22.94	0.00	0.00	0.58	57.10
% to DACS 01	300 - 400	-9.30	-19.36	0.00	-8.66	-9.52	-14.38	0.00	-25.04	0.00	-8.70	-9.09	-25.89
07 - AVGs	1800	11.33	1115.36	10.00	828.94	12.00	1264.60	3.00	919.95	3.00	589.44	4.00	1074.80
SDs		0.58	17.15	0.00	0.00	0.00	29.59	0.00	14.55	0.00	1.99	0.00	29.41
% to DACS 01	300 - 400	-5.56	-19.69	0.00	-15.76	-7.69	-21.78	0.00	-26.71	0.00	-10.48	0.00	-32.84
08 - AVGs	1800	10.00	979.53	10.00	828.94	11.00	1161.30	3.00	764.68	3.00	588.32	3.00	907.28
SDs		0.00	15.91	0.00	0.00	0.00	16.01	0.00	24.15	0.00	0.00	0.00	49.77
% to DACS 01	300 - 400	-9.09	-21.14	0.00	-13.06	-8.33	-21.23	0.00	-27.39	0.00	-10.67	0.00	-28.61
09 - AVGs	1800	12.00	1218.66	10.00	828.94	-	-	3.00	978.59	-	-	-	-
SDs		0.00	40.26	0.00	0.00	-	-	0.00	25.79	-	-	-	-
% to DACS 01	300 - 400	-12.20	-20.23	0.00	-18.56	-	-	0.00	-24.77	-	-	-	-
10 - AVGs	1800	11.33	1116.85	-	-	-	-	3.00	1003.90	-	-	-	-
SDs		0.58	7.84	-	-	-	-	0.00	11.04	-	-	-	-
% to DACS 01	300 - 400	-8.11	-19.46	-	-	-	-	0.00	-30.22	-	-	-	-
11 - AVGs	1800	11.67	1106.42	-	-	-	-	3.00	852.80	-	-	-	-
SDs		0.58	30.08	-	-	-	-	0.00	11.46	-	-	-	-
% to DACS 01	300 - 400	-10.26	-22.31	-	-	-	-	0.00	-28.41	-	-	-	-
12 - AVGs	1800	10.33	987.93	-	-	-	-	-	-	-	-	-	-
SDs		0.58	10.90	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	-6.06	-20.67	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	1800	13.00	1225.67	10.00	835.91	12.75	1404.20	3.18	983.86	3.00	591.92	3.54	1157.52
SDs		0.14	11.34	0.00	5.64	0.00	9.93	0.00	7.24	0.00	1.07	0.07	5.61
% to DACS 01	300 - 400	-7.69	-19.51	0.00	-14.43	-7.27	-17.14	0.00	-27.00	0.00	-14.69	-4.49	-26.88
% to SA+LNS [29]	1800	6.12	1.43	0.00	0.91	8.05	2.44	11.64	-0.31	0.00	-2.87	6.36	-0.83
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

4.6.2 Is the local search of triple moves doing well on large problem instances?

The dramatic improvement, in the performance and the outputted results of the system DACS 02 in Section 4.6.1 on the problem instances with 100 customers, has encouraged the author to test the system on larger problem instances with 200 or 400 customers each and to see whether such system is able to scale up in trying to solve such problem instances as efficient as the state-of-the-art approaches.

Therefore as in Tables 4.10 and 4.11, DACS 02 is tested on each of the problem groups PG200 and PG400 in Section 2.2 for three runs, which are stopped after 2400 or 4800 seconds. From Tables 4.12 and 4.13, it can be said that the performance and the results of DACS 02 on the six problem sets of PG200 and PG400 are not so bad and therefore they are considered as encouraging in comparison to other VRPTW algorithms like RVNSc [5], ES4C [2]...etc.

However the system DACS 02 does not outperform such algorithms. On average, DACS 02 is worse than all VRPTW algorithms on PG200 by 4.63% and 9.98% for NV and TD respectively. For instance on average, RVNSc outperforms DACS 02 on PG200 with 5.07% for NV and 9.46% for TD. Also on PG400, DACS 02 is worse, by 5.40% for NV and 22.63% for TD, on average as well. For that with 6.08% for NV and 22.78% for TD, ES4C is better on PG400 than DACS 02, on average, as an example.

4.6.3 Are the parallel ants improving anything at all?

One of the questions thought about is: How should the ants search for nodes to visit during the building up of solutions in the VMIN and DMIN colonies of the system DACS 02 in Sections 4.6.1 and 4.6.2? Should they search in parallel or sequential? Of course, these questions have triggered after reading two paragraphs in [4] and [51] respectively. One of the two paragraphs says that the ants of an ACS system, used for a TSP problem instance, should search in parallel, whereas in the other paragraph the ants are imagined to search in sequential.

Now in all the DACS systems used so far, the ants search for nodes to visit in a sequential way. Therefore, if there are ten ants queuing in order to build their solutions, then the first ant, before any other artificial ant, in the queue builds

Table 4.10: Comparison between the average case performances of DACS 02 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200. Check Tables C.7 and C.8 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time (secs)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	2400	20.33	5666.76	20.00	2713.96	19.00	4002.89	5.00	4321.64	6.00	1966.89	7.00	3440.00
SDs		0.58	88.79	0.00	16.26	0.00	83.07	0.00	106.22	0.00	19.02	0.00	58.01
% to RVNSc [5]	720 - 1680	7.02	12.78	0.00	0.35	5.56	-4.06	25.00	-10.97	0.00	1.84	16.67	8.59
02 - AVGs	2400	19.00	4511.13	19.33	3686.73	19.00	3702.85	5.00	3772.64	6.00	1914.02	5.67	3033.96
SDs		0.00	162.92	0.58	150.28	0.00	178.02	0.00	138.60	0.00	5.62	0.58	182.54
% to RVNSc [5]	720 - 1680	5.56	6.54	7.41	17.39	5.56	0.76	25.00	-1.33	0.00	2.73	13.33	5.60
03 - AVGs	2400	19.00	3909.77	19.00	3776.04	18.00	4390.49	4.00	3304.57	6.67	1993.12	5.00	2785.87
SDs		0.00	99.10	0.00	392.79	0.00	106.62	0.00	110.05	0.58	120.37	0.00	109.41
% to RVNSc [5]	720 - 1680	5.56	10.06	5.56	33.29	0.00	33.68	0.00	9.29	11.11	10.20	25.00	1.55
04 - AVGs	2400	18.00	3841.86	19.00	3083.95	18.00	3787.33	4.00	2351.49	7.00	1945.55	4.00	2513.79
SDs		0.00	21.26	0.00	169.31	0.00	189.35	0.00	54.27	0.00	22.10	0.00	21.75
% to RVNSc [5]	720 - 1680	0.00	20.90	5.56	14.38	0.00	24.41	0.00	16.36	16.67	10.87	0.00	19.09
05 - AVGs	2400	19.00	4739.68	20.00	2743.41	19.00	3824.30	4.00	3714.44	6.00	1972.67	5.33	3135.01
SDs		0.00	311.23	0.00	5.89	0.00	104.63	0.00	21.16	0.00	68.85	0.58	101.20
% to RVNSc [5]	720 - 1680	5.56	6.17	0.00	1.53	5.56	-1.62	0.00	6.56	0.00	4.97	33.33	-7.24
06 - AVGs	2400	18.67	4143.41	20.33	2818.68	19.00	3845.74	4.00	3267.52	6.00	2096.45	5.00	3051.57
SDs		0.58	261.26	0.58	192.82	0.00	106.22	0.00	7.64	0.00	106.34	0.00	54.55
% to RVNSc [5]	720 - 1680	3.70	8.33	1.67	4.36	5.56	3.78	0.00	8.58	0.00	12.87	0.00	14.45
07 - AVGs	2400	18.00	4337.51	20.00	2886.73	19.00	3725.49	4.00	2770.46	6.33	1926.75	4.67	2922.17
SDs		0.00	218.39	0.00	126.04	0.00	68.63	0.00	33.57	0.58	50.96	0.58	185.46
% to RVNSc [5]	720 - 1680	0.00	30.23	0.00	6.87	5.56	6.19	0.00	9.33	5.56	4.14	16.67	8.07
08 - AVGs	2400	18.00	3591.89	20.00	2880.11	19.00	3490.30	4.00	2298.87	6.00	1926.79	4.00	2713.62
SDs		0.00	23.81	0.00	76.70	0.00	24.18	0.00	93.17	0.00	34.66	0.00	116.72
% to RVNSc [5]	720 - 1680	0.00	16.40	5.26	2.87	5.56	4.44	0.00	23.84	0.00	5.71	0.00	14.40
09 - AVGs	2400	19.00	4412.82	19.00	2969.61	18.67	3804.97	4.00	3460.71	6.67	1988.83	4.00	2567.42
SDs		0.00	56.22	0.00	130.04	0.58	528.61	0.00	152.50	0.58	109.42	0.00	118.22
% to RVNSc [5]	720 - 1680	5.56	9.59	5.56	7.00	3.70	16.58	0.00	10.38	11.11	7.61	0.00	15.05
10 - AVGs	2400	18.00	4180.21	18.67	3054.69	18.00	4340.42	4.00	3037.70	6.00	2110.92	4.00	2347.65
SDs		0.00	114.05	0.58	240.12	0.00	261.81	0.00	60.88	0.00	92.93	0.00	26.07
% to RVNSc [5]	720 - 1680	0.00	19.66	3.70	11.71	0.00	35.72	0.00	12.12	0.00	16.71	0.00	15.37
DACS 02 - AVGs	2400	18.70	4333.50	19.53	3061.39	18.67	3891.48	4.20	3230.00	6.27	1984.20	4.87	2851.11
SDs		0.10	93.88	0.06	82.89	0.06	102.12	0.00	50.02	0.06	19.41	0.15	38.50
% to RVNSc [5]	720 - 1680	3.31	13.40	3.35	10.17	3.70	10.93	5.00	6.07	4.44	7.69	10.61	8.47
% to ES4C [2]	2400	2.75	16.96	3.35	10.04	3.70	9.46	5.00	5.73	4.44	7.49	13.18	6.58
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table 4.11: Comparison between the average case performances of DACS 02 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400. Check Tables C.9 and C.10 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	4800	41.00	13012.27	40.00	7534.32	39.00	10156.04	9.33	10669.00	12.00	4306.88	13.00	8052.91
SDs		1.00	471.31	0.00	72.38	1.00	147.40	0.58	341.39	0.00	25.19	1.00	166.62
% to RVNSc [5]	3900 - 7980	7.89	17.40	0.00	5.34	5.41	12.37	16.67	9.60	0.00	4.63	18.18	6.64
02 - AVGs	4800	38.00	12489.82	41.00	13521.01	38.00	10118.30	9.00	9486.57	12.67	5174.99	11.67	8003.99
SDs		0.00	554.93	1.00	1228.95	0.00	58.71	0.00	198.96	0.58	524.43	0.58	322.11
% to RVNSc [5]	3900 - 7980	5.56	24.09	10.81	77.91	5.56	12.17	12.50	18.71	5.56	29.83	16.67	25.91
03 - AVGs	4800	37.67	10358.99	38.00	12375.00	37.00	9794.55	8.00	7699.69	13.00	5737.35	9.33	6412.74
SDs		0.58	239.06	0.00	797.95	0.00	315.66	0.00	280.22	0.00	616.72	0.58	75.78
% to RVNSc [5]	3900 - 7980	4.63	19.66	5.56	64.89	2.78	18.71	0.00	23.05	8.33	46.22	16.67	18.15
04 - AVGs	4800	37.00	8980.87	37.00	9852.32	36.67	9835.38	8.00	5950.88	13.00	5040.46	8.00	5074.29
SDs		0.00	108.79	0.00	360.17	0.58	1210.69	0.00	23.33	0.00	234.96	0.00	93.49
% to RVNSc [5]	3900 - 7980	2.78	16.96	2.78	35.01	1.85	26.95	0.00	29.59	8.33	31.34	0.00	34.60
05 - AVGs	4800	37.67	12082.47	40.33	8257.89	38.00	10198.70	8.00	8769.74	12.67	4641.00	12.00	7384.34
SDs		0.58	347.17	0.58	594.74	0.00	414.38	0.00	228.33	0.58	217.69	0.00	32.63
% to RVNSc [5]	3900 - 7980	4.63	17.96	0.83	15.46	5.56	10.91	0.00	17.53	5.56	17.62	33.33	12.59
06 - AVGs	4800	37.33	11121.52	42.00	8527.02	38.00	10167.88	8.00	7908.02	13.00	5295.34	10.67	7214.29
SDs		0.58	125.90	1.00	433.49	1.00	187.31	0.00	194.90	0.00	193.96	0.58	245.37
% to RVNSc [5]	3900 - 7980	3.70	18.57	5.00	19.20	5.56	13.30	0.00	21.03	8.33	36.30	18.52	19.61
07 - AVGs	4800	37.00	10322.91	40.67	8424.06	38.00	10376.61	8.00	6855.46	13.33	5109.95	10.00	6739.42
SDs		0.00	184.26	0.58	433.38	0.00	180.11	0.00	81.46	0.58	99.69	0.00	146.61
% to RVNSc [5]	3900 - 7980	2.78	24.39	1.67	17.83	5.56	17.89	0.00	25.98	11.11	29.84	25.00	11.95
08 - AVGs	4800	36.00	10947.19	41.33	8704.51	37.00	10138.50	8.00	5661.35	12.00	4877.92	8.67	5994.52
SDs		0.00	128.61	0.58	407.16	0.00	325.59	0.00	235.86	0.00	149.12	0.58	83.20
% to RVNSc [5]	3900 - 7980	0.00	43.64	8.77	17.94	2.78	18.31	0.00	30.56	0.00	26.83	8.33	16.50
09 - AVGs	4800	37.33	11319.90	39.00	10125.30	37.00	10169.17	8.00	7645.82	13.33	5372.97	8.00	5822.30
SDs		0.58	199.11	0.00	1239.82	0.00	393.70	0.00	120.98	0.58	243.01	0.00	42.22
% to RVNSc [5]	3900 - 7980	3.70	17.55	5.41	39.01	2.78	20.02	0.00	10.99	11.11	35.71	0.00	19.95
10 - AVGs	4800	37.00	10428.63	38.00	10793.63	37.00	9793.17	8.00	7381.01	12.00	4753.15	8.00	5533.92
SDs		0.00	148.55	0.00	1351.13	0.00	390.70	0.00	78.00	0.00	253.05	0.00	35.30
% to RVNSc [5]	3900 - 7980	2.78	17.33	5.56	43.10	2.78	19.18	0.00	18.25	0.00	25.36	0.00	19.85
DACS 02 - AVGs	4800	37.60	11106.46	39.73	9811.51	37.57	10074.83	8.23	7802.75	12.70	5031.00	9.93	6623.27
SDs		0.10	35.77	0.32	244.57	0.06	95.07	0.06	67.87	0.10	142.18	0.15	49.83
% to RVNSc [5]	3900 - 7980	3.87	21.32	4.56	34.01	4.06	16.76	2.92	19.16	5.83	28.25	14.18	17.57
% to ES4C [2]	4800	3.58	24.44	4.56	29.37	4.06	14.97	2.92	20.01	5.83	27.85	15.50	20.03
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table 4.12: Averages of the best solutions computed by different VRPTW algorithms on the problem group PG200 - check Table I.10 for more information.

		R1		C1		RC1		R2		C2		RC2	
Algorithm	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
RVNSc [5]	720-1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
HM4 [3]	504	18.20	3808.27	18.90	2838.93	18.00	3717.96	4.00	3095.33	6.00	1852.63	4.40	2651.35
HM4C [3]	504	18.20	3855.03	18.90	2842.08	18.10	3674.91	4.00	3032.49	6.00	1856.99	4.40	2671.34
HG05a [64]	96	18.20	3890.06	19.00	2836.66	18.10	3734.32	4.10	3059.78	6.00	1898.44	4.50	2640.94
LL [62]	10926	18.30	3736.20	19.10	2728.60	18.30	3385.80	4.10	3023.00	6.00	1854.90	4.90	2518.70
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89
SA+LNS [29]	N/A	18.20	3677.96	18.90	2726.63	18.00	3279.99	4.10	3023.62	6.00	1860.17	4.50	2603.08
Best [Table I.2]	N/A	18.10	3643.20	18.80	2713.48	18.00	3196.95	4.00	2936.56	6.00	1831.76	4.30	2549.30

Table 4.13: Averages of the best solutions computed by different VRPTW algorithms on the problem group PG400 - check Table I.10 for more information.

		R1		C1		RC1		R2		C2		RC2	
Algorithm	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
HM4 [3]	1704	36.30	9396.99	38.00	7853.09	36.10	9139.82	8.00	6603.38	12.00	3969.36	8.90	5614.49
HM4C [3]	1704	36.30	9478.22	38.00	7855.82	36.10	9294.99	8.00	6650.28	12.00	3940.19	8.80	5629.43
HG05a [64]	306	36.40	9547.86	38.10	7921.19	36.10	9296.75	8.00	6683.53	12.00	4049.71	9.20	5609.88
LL [62]	21588	36.60	8912.40	38.70	7181.40	36.50	8377.90	8.00	6610.60	12.10	4017.10	9.50	5466.20
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87
SA+LNS [29]	N/A	36.40	8713.37	38.00	7220.96	36.10	8330.98	8.00	6959.75	12.00	4154.40	8.90	5631.70
Best [Table I.3]	N/A	36.20	8558.44	37.60	7261.40	36.00	7998.46	8.00	6197.61	12.00	3837.92	8.50	5318.70

its solution sequentially (i.e. the node-by-node basis) using the probabilistic-state transition component with its exploitation and exploration parts in Section 4.4.5. Once the first ant is finished from building its solution, the second ant in the queue comes into effect and so on. The sequential nature of the search way in an ant can be recognised from the do-while loop in the routing builder in Figure 4.5.

In order to see the effects of changing the sequential search way of the ants to the parallel search way, the do-while loop in the ants' routing builder in Figure 4.5 is removed and the cycles of VMIN and DMIN are redesigned to encourage the ants to search for nodes to visit simultaneously in parallel. For that in the parallel search, the first ant in the queue visits an edge connecting an already visited node (a customer or a depot) and the node to be visited using the probabilistic-state transition component with its exploitation and exploration parts and then it uses the local updating rule in Section 4.4.6 to diminish the pheromone trail of the most recently visited edge. Once a node is visited using the first ant, the second ant comes into play in order to visit a node that is not necessarily as same as the node visited by the first ant.

Also if the second ant has finished its visit, then the role of the third ant comes into play and so on until all the ten ants in the cycle of a colony visit a node each. After all the ten ants visit a node each, just then each of the ten ants starts to think about visiting the next node in its journey. The parallel search of the ten ants continues to do the procedure described above until a number of nodes equal to the number of nodes in a problem instance are tried in each ant. Of course in the parallel search of ants, an ant may build a feasible or an infeasible solution as in the sequential search way of ants. If an infeasible solution is built, then the solution may become feasible after applying the insertion procedure in Section 4.4.7 - designed to insert the remaining unvisited customers.

Subsequently in Table C.11, it can be seen from the slightly deteriorated percentages 0.64% and 0.29% that the NV and TD results of 13.06 and 1242.05 of the DACS system that uses the parallel search way of ants are not, on average, significant on the problem set R1 of the problem group PG100. Furthermore, the DACS system with the parallel search way of ants may have improved in terms of NV on some problem instances (like R103 and R110) but it has got worse on others. Thus, the general conclusion, on the whole problem set of R1, is that the change in the

search way of ants has not made any significant difference and therefore the parallel search way of ants is useless and unnecessary.

4.6.4 What if the initialisation technique is an insertion heuristic?

In the DACS system in Sections 4.6.1 and 4.6.2, the nearest neighbourhood heuristic NN is used as an initialisation technique in two ways. The first way is to use the NN heuristic, in the initialisation step of DACS in order to create an initial solution and to consider it as the best global solution before doing any search in any colony. The second way is to use the NN heuristic, in the initialisation steps of the colonies VMIN and DMIN, in order to calculate two different initial values of τ_o for the pheromone trails in the pheromone memory structures of such colonies and later to use such τ_o values differently in the local pheromone updating rule in Section 4.4.6.

Now in order to see the effects of changing an initialisation technique to another on improving the performance and the results of the DACS system, an insertion heuristic, called SI1-Like 01, is used instead of the NN heuristic. The insertion heuristic has some similarities with Solomon's I1 heuristic mentioned in [1]. In SI1-Like 01, the braced parametric values ($a_1 = 1$, $a_2 = 0$, $\lambda = 1$ and $\mu = 1$) are used. For more information about insertion heuristics, check Chapter 5.

It can be recognized, from Tables C.12, C.13 and C.14, that the performance and the results of the system DACS 2.1 that uses SI1-Like 01 have improved on many occasions on the six problem sets of the problem group PG100 and their problem instances as indicated by the numeric results (attached with the minus sign) in comparison with that of DACS 2.1 that uses NN. Despite the improvement on average by -0.04% and -0.74%, however the two systems cannot dominate each other in terms of NV and TD together.

For example, the new system with SI1-Like 01 is able sometimes to dominate (in terms of NV and TD together) the system with NN at some problem instances but not all of them and generally on at most three problem sets out of six after any amount of elapsed CPU time. Of course, this discovery is indicated in Tables C.12 and C.13 from the cells that have numeric results colored with red and blue at the same time.

Moreover, the addition of SI1-Like 01 has not made the performance and results of DACS 2.1 in any way near or as competitive as those of the MACS-VRPTW system in [4] and the other VRPTW algorithms in the literature. Because of the inability of the system DACS 2.1 + SI1-Like 01 to dominate and to bring competitive performance and results, the author has decided to abandon doing any experimentation any more with such system and to continue from now on our investigation with the system DACS 2.1 that uses the NN heuristic.

4.6.5 What about changing the kind of candidate lists used?

In this section, the aim is to see what are the effects of using different orderings of the candidate nodes within the candidate lists and the effects of changing time-oriented into distance-oriented candidate lists on the performance and the results of the DACS 2.1 system in Section 4.6.4.

In the most recent DACS systems like DACS 02 and DACS 2.1 in Sections 4.6.1, 4.6.2 and 4.6.4, the time-oriented candidate lists are used only within the local search of triple moves (M1 to M3 in Section 4.4.8) and up to 20 nearest nodes of each list are used as candidates for each visited node in the built routes of the feasible solution of an ant coming from the DMIN colony.

However, the systems tested in this section also use the candidate lists within the exploitation and exploration parts of the probabilistic state transition rule in Section 4.4.5 and in the insertion procedure described in Section 4.4.7. Behold that in the components described earlier, the number of candidate nodes is not limited to 20 as in the local search of triple moves mentioned above.

Therefore in the probabilistic state transition rule with its exploitation and exploration parts, each of the candidate nodes in a candidate list might have the chance for a visit by an ant. Also in the insertion procedure, a remaining unvisited customer c_i might have the chance to be inserted near any node of the n candidate nodes in his candidate list but the candidate node has to be visited somewhere in a route.

It can be seen from Table C.15 that changing the candidate lists from a time-oriented into a distance-oriented fashion has not made any significant difference. For that insignificantly on the problem set R1 of the problem group PG100, the NV result has deteriorated by 0.44% on average and the TD result has improved by

−0.38%. Furthermore, changing the time-oriented into distance-oriented candidate lists has not made the performance and the results of the DACS system in any way a little bit closer to the performance and the results of MACS-VRPTW in [4] and the other VRPTW algorithms in the literature. Therefore, using any particular kind of candidate lists and changing the way in which the candidates are ordered is immaterial.

4.6.6 What if the pheromone updating changed and LS moves near depots added?

In an effort to improve the performance and results of the DACS 2.1 system in Section 4.6.4 and to make DACS 2.1 as competitive as MACS-VRPTW in [4], the way, in which the pheromone trails are updated locally and globally, is changed and new local search moves near the depot nodes of a solution are tried.

In this section, the pheromone local and global updating components in Sections 4.4.6 and 4.4.9 update the pheromone trails, not just in the pheromone memory structure of the active colony, VMIN or DMIN, but also in the pheromone memory structure of the other colony that is idle and waiting for its role to start a new search for new solutions. So, whatever colony is active, the pheromone trails in the two colonies will be updated rather than only the pheromone trails of the active colony.

In addition, the local search of triple moves is redesigned to work as follows. For instance, if a visited customer c_i in a route is currently selected and the nearest node c_j in the candidate list of c_i is the depot that represents all the duplicated depots, then the local search of triple moves is going to try to relocate c_i to be near any duplicated depot (either at the start or at the end of a route) and later to choose the best feasible move (in terms of the total of traveled distances) out of all the moves tried near the duplicated depots in order to possibly improve things.

As a resultant, two different DACS systems in Table C.16 are used to check what is discussed in previous paragraphs from ideas. The important thing to recognise is the two DACS systems, tried with such ideas, have not managed to perform competitively as well as the MACS-VRPTW system in [4] and other VRPTW algorithms in the literature.

Moreover, it can be seen from the table that changing the new way the pheromone

trails are updated locally and globally has not led, on the problem set R1 of the problem group PG100, into any significant improvement on the performance and the results of the DACS 2.1 system. On average, the NV and TD results are improved on R1 trivially by -0.43% and -0.44% respectively. Also despite the significant improvement on R1, by -2.09% on average, in terms of TD because of adding local search moves near the duplicated depot nodes, however the time taken in doing such moves and the slight deterioration on average, in terms of NV by 0.43%, has discouraged the author from including them into the local search of triple moves.

4.7 Specific multiple ant colony systems

After the significant improvement of the performance and the results of the DACS system because of adding the local search of triple moves in Sections 4.6.1 and 4.6.2, the addition of the other components in Sections 4.6.3, 4.6.4, 4.6.5 and 4.6.6 have not managed to let the system look competitive in comparison to other VRPTW algorithms in the literature like MACS-VRPTW [4], SA+LNS [29], LS [28]...etc.

For that, this section investigates and studies the effects of adding a number of components, mentioned below, on improving the performance and the results of the DACS system. Those components are such as the strategy of push-forward and push-backward PFPBS in Section 4.7.1, the colonies with different objectives or goals in Section 4.7.2, the hybrid local search HLS in Section 4.7.3, the 2-Opt move variant in Section 4.7.4 and the saving ants as in Section 4.7.6 that pick up edges rather than singular nodes. Also, the elements that might be behind the synergistic effects of ants are looked at in Section 4.7.7. Finally, the more deterministic ants using simple heuristics are tried in Section 4.7.8 as a substitute to the pheromone ants.

4.7.1 What about including a push-forward and push-backward strategy PFPBS?

In the system DACS 02 in Sections 4.6.1 and 4.6.2, there is no use for the push-forward and push-backward strategy PFPBS, described in Section 4.4.11, in the probabilistic state transition rule with its exploitation and exploration parts, the insertion procedure or the local search of triple moves - M1 to M3 in Section 4.4.8.

For that, a lot of waste CPU time is expected and it would be worthwhile to see the effects of including that strategy on the performance and the results of the DACS system.

Then after adding the PFPBS strategy to DACS 02, the new system DACS 03 is tested, at six different amounts of CPU time in seconds (100, 300, 400, 600, 1200 and 1800), for thirty runs on the problem group PG100 in Section 2.2 according to the methodology of experimentation in Section 4.2. In terms of the number of vehicles firstly and the total of travelled distances secondly, it can be seen from Tables 4.14 and 4.15 that the system DACS 03 is able, at the six amounts of CPU time in seconds, to make a significant improvement on average by -3.07% for NV and -3.25% for TD on all the six problem sets of Solomon [1].

In the best case scenario at the six different amounts of CPU time, the PFPBS strategy improves on average the performance and the results by -3.34% for NV and -3.77% for TD. The worst case scenario is to get things better on average by -1.83% and -2.05% for NV and TD respectively. However in the worst case and on rare occasions only after 1800 seconds, the performance and results, on the problem sets C1, C2 and RC2, might deteriorate by 1.21% to 1.83% in terms of TD in particular in comparison with that of DACS 02.

When it comes to comparing DACS 03 to all other VRPTW algorithms in Table 4.3 on the problem group PG100, it can be said that the percentage of deviations, on average, for NV and TD is 0.94% and -0.25% respectively, which are much better than those of the system DACS 02 - for NV 4.21% and for TD 3.22%.

For instance, DACS 03 is deviated on average, by 1.84% for NV and 0.38% for TD, from MACS-VRPTW [4]. Also in relation to the algorithm SA+LNS [29] run for 1800 seconds, DACS 03 is departed on average after the same amount of CPU time, by 2.59% for NV and -0.61% for TD. However interestingly and with 7.03% for NV and 0.55% for TD, the algorithms LS [28] and LS+TA [28] after 126 to 156 seconds outperforms, on average, DACS 03 run for 100 seconds only.

Overall, adding the push-forward and the push-backward strategy PFPBS, which stores, uses and updates information as those mentioned in I1 to I6 in Section 4.4.11, is very successful and has resulted in making the performance and results very much so closer to the performance and results of other VRPTW algorithms in the literature. As a consequence, the DACS system without the strategy described

earlier is significantly worse than another DACS system that does use it.

Table 4.14: Comparison between the average case performances of the algorithms DACS 03 and DACS 02, after three runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
DACS 03 - AVGs	100	12.96	1242.95	10.00	834.85	12.61	1413.30	3.16	991.42	3.00	592.44	3.60	1169.58
SDs		0.12	9.21	0.00	4.98	0.12	15.42	0.04	7.30	0.00	2.04	0.08	17.18
% to DACS 02	100	-3.60	-3.60	0.00	-2.99	-4.51	-2.86	-3.43	-9.41	-8.86	-11.39	-6.20	-12.08
% to MACS-VRPTW [4]	100	3.28	2.32	0.00	0.78	1.22	1.28	3.63	2.00	0.00	-0.13	6.39	-1.87
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
SDs		0.09	7.51	0.00	3.54	0.12	10.23	0.07	6.56	0.00	2.07	0.08	19.32
% to DACS 02	300	-3.65	-1.04	0.00	-2.82	-3.24	-1.29	-1.62	-9.25	-2.70	-3.07	-5.68	-8.71
% to MACS-VRPTW [4]	300	2.54	1.59	0.00	0.41	2.71	1.06	4.34	-0.39	0.00	-0.14	3.85	-0.51
DACS 03 - AVGs	400	12.74	1229.23	10.00	831.32	12.40	1401.11	3.10	963.04	3.00	592.16	3.44	1162.37
SDs		0.09	6.18	0.00	3.57	0.14	8.86	0.06	7.88	0.00	2.07	0.08	19.23
% to DACS 02	400	-4.26	-1.50	0.00	-1.53	-3.34	-1.82	-2.57	-7.24	-2.70	-1.32	-7.30	-6.76
% to MACS-VRPTW [4]	300	2.32	1.34	0.00	0.36	2.26	0.86	3.33	-0.62	0.00	-0.14	3.23	-0.51
DACS 03 - AVGs	600	12.70	1225.99	10.00	831.08	12.37	1397.75	3.08	961.14	3.00	592.05	3.42	1162.62
SDs		0.08	7.18	0.00	3.66	0.14	8.84	0.07	8.17	0.00	2.08	0.07	17.98
% to DACS 02	600	-3.52	-1.51	0.00	-3.40	-2.66	-1.05	-3.24	-5.79	0.00	-0.20	-7.75	-3.11
% to MACS-VRPTW [4]	600	2.61	1.04	0.00	0.33	2.41	1.26	2.63	-0.44	0.00	-0.14	2.73	-0.04
DACS 03 - AVGs	1200	12.63	1223.57	10.00	830.75	12.32	1392.88	3.05	959.58	3.00	591.84	3.40	1159.78
SDs		0.09	6.14	0.00	3.58	0.15	7.52	0.05	8.00	0.00	2.09	0.06	16.36
% to DACS 02	1200	-3.89	-0.61	0.00	-2.00	-2.12	-1.51	-3.37	-3.47	0.00	-0.12	-5.00	-1.15
% to MACS-VRPTW [4]	1200	2.00	0.98	0.00	0.29	2.98	0.52	1.52	-0.26	0.00	-0.03	2.23	0.53
DACS 03 - AVGs	1800	12.60	1221.49	10.00	830.60	12.29	1390.92	3.04	958.53	3.00	591.82	3.39	1158.81
SDs		0.10	6.62	0.00	3.60	0.13	7.86	0.05	7.65	0.00	2.10	0.04	16.56
% to DACS 02	1800	-3.08	-0.34	0.00	-0.64	-3.59	-0.95	-4.48	-2.57	0.00	-0.02	-4.24	0.11
% to MACS-VRPTW [4]	1800	1.78	0.88	0.00	0.27	3.12	0.20	1.31	-0.19	0.00	-0.01	1.85	0.83
DACS 02 - AVGs	100	13.44	1289.43	10.00	860.55	13.21	1454.84	3.27	1094.37	3.29	668.60	3.83	1330.21
	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
	400	13.31	1247.99	10.00	844.26	12.83	1427.10	3.18	1038.20	3.08	600.10	3.71	1246.69
	600	13.17	1244.77	10.00	860.31	12.71	1412.59	3.18	1020.19	3.00	593.24	3.71	1199.88
	1200	13.14	1231.12	10.00	847.72	12.58	1414.30	3.15	994.12	3.00	592.53	3.58	1173.25
	1800	13.00	1225.67	10.00	835.91	12.75	1404.20	3.18	983.86	3.00	591.92	3.54	1157.52
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

4.7.2 What if colonies with new different objective functions are injected?

Later after the significant improvement in the performance and results of the system DACS 03 in Section 4.7.1, the plan in this section is to check and to see if adding other types of colonies with goals or objectives, different from the goals or objectives

Table 4.15: Comparison between the average case performances of DACS 03 and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100. Check Tables D.1 and D.2 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.00	1676.48	10.00	828.94	14.97	1698.55	4.00	1289.27	3.00	591.56	4.00	1478.86
SDs		0.00	14.42	0.00	0.00	0.18	35.73	0.00	21.68	0.00	0.00	0.00	36.84
% to DACS 02	1800	-1.72	-0.95	0.00	0.00	-2.39	-0.53	0.00	0.02	0.00	0.00	0.00	-1.47
02 - AVGs	1800	17.20	1491.96	10.00	835.26	13.00	1525.47	3.37	1183.22	3.00	591.56	4.00	1195.89
SDs		0.41	8.81	0.00	15.63	0.00	19.66	0.49	62.07	0.00	0.00	0.00	33.62
% to DACS 02	1800	-0.77	-0.50	0.00	0.76	-4.88	0.88	-15.83	4.66	0.00	0.00	0.00	-1.92
03 - AVGs	1800	13.50	1273.67	10.00	831.10	11.37	1308.33	3.00	967.03	3.00	594.53	3.00	1120.97
SDs		0.51	46.24	0.00	8.13	0.49	23.92	0.00	11.49	0.00	4.49	0.00	27.63
% to DACS 02	1800	-3.57	2.99	0.00	-4.90	-2.57	-2.46	0.00	-4.44	0.00	0.57	0.00	-0.71
04 - AVGs	1800	10.17	1013.27	10.00	824.92	10.53	1161.02	3.00	770.38	3.00	602.78	3.00	829.15
SDs		0.38	15.60	0.00	0.61	0.51	19.10	0.00	10.74	0.00	16.83	0.00	16.99
% to DACS 02	1800	-7.58	-0.67	0.00	-2.57	-4.24	-1.97	0.00	-6.62	0.00	-0.52	0.00	-4.31
05 - AVGs	1800	14.03	1435.36	10.00	830.05	14.40	1586.04	3.00	1062.94	3.00	588.88	4.00	1383.65
SDs		0.18	21.10	0.00	6.10	0.50	31.18	0.00	22.64	0.00	0.00	0.00	34.17
% to DACS 02	1800	-4.32	0.94	0.00	0.13	-1.82	-0.91	0.00	0.26	0.00	0.00	0.00	1.78
06 - AVGs	1800	12.20	1287.55	10.00	828.94	12.03	1432.27	3.00	940.52	3.00	588.49	3.10	1241.82
SDs		0.41	18.00	0.00	0.00	0.18	32.25	0.00	15.40	0.00	0.00	0.31	67.55
% to DACS 02	1800	-6.15	-1.91	0.00	0.00	-5.00	-2.00	0.00	-4.59	0.00	0.00	-7.00	3.24
07 - AVGs	1800	10.97	1098.79	10.00	831.26	11.20	1265.91	3.00	837.16	3.00	588.40	3.03	1146.59
SDs		0.18	16.39	0.00	8.82	0.41	28.93	0.00	11.99	0.00	0.63	0.18	48.88
% to DACS 02	1800	-3.24	-1.49	0.00	0.28	-6.67	0.10	0.00	-9.00	0.00	-0.18	-24.17	6.68
08 - AVGs	1800	10.00	964.33	10.00	836.00	10.83	1149.74	2.07	754.10	3.00	588.32	3.00	873.57
SDs		0.00	10.92	0.00	19.70	0.38	18.56	0.25	14.14	0.00	0.00	0.00	23.47
% to DACS 02	1800	0.00	-1.55	0.00	0.85	-1.52	-1.00	-31.11	-1.38	0.00	0.00	0.00	-3.72
09 - AVGs	1800	12.00	1198.78	10.00	828.94	-	-	3.00	953.58	-	-	-	-
SDs		0.00	25.90	0.00	0.00	-	-	0.00	16.58	-	-	-	-
% to DACS 02	1800	0.00	-1.63	0.00	0.00	-	-	0.00	-2.56	-	-	-	-
10 - AVGs	1800	11.10	1129.44	-	-	-	-	3.00	984.59	-	-	-	-
SDs		0.40	21.32	-	-	-	-	0.00	19.36	-	-	-	-
% to DACS 02	1800	-2.06	1.13	-	-	-	-	0.00	-1.92	-	-	-	-
11 - AVGs	1800	11.03	1101.47	-	-	-	-	3.00	801.04	-	-	-	-
SDs		0.18	18.05	-	-	-	-	0.00	12.84	-	-	-	-
% to DACS 02	1800	-5.43	-0.45	-	-	-	-	0.00	-6.07	-	-	-	-
12 - AVGs	1800	10.00	986.77	-	-	-	-	-	-	-	-	-	-
SDs		0.00	15.25	-	-	-	-	-	-	-	-	-	-
% to DACS 02	1800	-3.23	-0.12	-	-	-	-	-	-	-	-	-	-
DACS 03 - AVGs	1800	12.60	1221.49	10.00	830.60	12.29	1390.92	3.04	958.53	3.00	591.82	3.39	1158.81
SDs		0.10	6.62	0.00	3.60	0.13	7.86	0.05	7.65	0.00	2.10	0.04	16.56
% to DACS 02	1800	-3.08	-0.34	0.00	-0.64	-3.59	-0.95	-4.48	-2.57	0.00	-0.02	-4.24	0.11
% to SA+LNS [29]	1800	2.86	1.08	0.00	0.27	4.17	1.47	6.65	-2.87	0.00	-2.88	1.85	-0.72
DACS 02 - AVGs	1800	13.00	1225.67	10.00	835.91	12.75	1404.20	3.18	983.86	3.00	591.92	3.54	1157.52
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

of the colonies VMIN and DMIN, can make any difference. For that, the five colonies, mentioned below from G1-G5, are configured in order to inject anyone of them into DACS 03 and to see if any of the different types of multiple ant colony systems, may be created, can lead to better performance and new improved quality results.

G1- The colony CWTsMIN has the responsibility to minimise the total of customer waiting times for vehicles to arrive.

G2- The colony CWTsMAX has the responsibility to maximise the total of customer waiting times for vehicles to arrive.

G3- The colony VWTsMIN has the responsibility to minimise the total of vehicle waiting times for the start times of customers to open.

G4- The colony VWTsMAX has the responsibility to maximise the total of vehicle waiting times for the start times of customers to open.

G5- The colony TMIN has the responsibility to minimise the total time consumed in vehicle travelling, waiting and servicing.

In each of the previous five colonies, the insertion procedure in Section 4.4.7 and the local search of triple moves (M1 to M3 in Section 4.4.8) of the current feasible solution of each ant are modified to work in a way to accept neighbouring solutions that either minimise or maximise a value that is related to the total of customer waiting times, the total of vehicle waiting times or the total consumed time in vehicle travelling, waiting and servicing.

Also, the cycle in each of the colonies of CWTsMIN, CWTsMAX, VWTsMIN and VWTsMAX is designed to accept a solution of an ant as the new best global solution if that solution improves the solution quality of the current best global solution according to the following objective criteria: firstly reducing the number of vehicles, then secondly reducing the total of travelled distances and finally minimising or maximising the total of customer or vehicle waiting times. The only colony that is different is the TMIN colony where its cycle is designed to take into consideration only the minimization of the total consumed time in vehicle travelling, waiting and servicing as the main objective criterion to choose new best global solutions.

According to Student's t-tests and Wilcoxon signed rank tests done on four types of triple ant colony systems or TACS systems run (for thirty runs of 100

seconds each) on the problem set R1 of the problem group PG100 as in Table 4.16, the addition of each of the four colonies CWTsMIN, CWTsMAX, VWTsMIN or VWTsMAX into the system DACS 03 has not made, on average by 0.15%, any significant impact on the performance in terms of the number of vehicles. Also for the three TACS systems that have the three colonies of CWTsMIN, CWTsMAX and VWTsMAX, the results in terms of the total of travelled distances have deteriorated significantly, by 0.65% on average, according to Student's t-tests and Wilcoxon signed rank tests. However when the VWTsMIN colony is added in DACS 03, the results in terms of the total of travelled distances have not been significant as the percentage 0.11% says that on average and according to Student's t-tests and Wilcoxon signed rank tests.

Afterwards, the two colonies of CWTsMIN and CWTsMAX are added into the system DACS 03 to create a quadruple ant colony system QACS and to see if things get better. Then, the QACS system is run for thirty runs of 100 seconds each on the problem set R1 of the problem group PG100. Unfortunately, the outcomes of adding the two colonies of CWTsMIN and CWTsMAX are insignificant either.

Thereafter, the DMIN colony of the system DACS 03 is replaced with the TMIN colony mentioned in G5 in order to see if better performance and quality results can be achieved. After replacing DMIN with TMIN, the new DACS system that has VMIN and TMIN is tested on the problem set R1 of the problem group PG100 for three runs of 1200 seconds. Also, the performance and the results of the new DACS system are realized to be horrible and literally very bad.

4.7.3 What about adding a hybrid local search HLS?

Although the great improvement in performance and results because of including the push-forward and push-backward strategy PFPBS as in Section 4.7.1, however the system DACS 03 is not reducing the number of vehicles enough during the amount of CPU time allocated. For reducing the number of vehicles on average, DACS 03 should embed the hybrid local search HLS, described in Section 4.4.10, in the cycle phases of the colonies VMIN and DMIN - as in the step a in Figures 4.3 and 4.4. Then, the HLS takes part in only when there is NOT a single solution feasibly created from any of the ten ants set out to search for feasible solutions in a cycle of a colony.

Table 4.16: Four different types of multiple ant colony systems tested, after thirty runs of 100 seconds or in ten batches of three runs, on the problem set R1 of the problem group PG100.

		DACS 03 alone		DACS+ CWTsMIN		DACS+ CWTsMAX		DACS+ VWTsMIN		DACS+ VWTsMAX	
Batch No.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	100	12.89	1239.11	12.92	1246.78	13.06	1247.47	13.06	1238.35	13.06	1243.11
SDs		0.10	2.45	0.08	5.72	0.10	4.59	0.19	5.41	0.13	10.57
CVs		0.75	0.20	0.65	0.46	0.74	0.37	1.47	0.44	0.98	0.85
02 - AVGs	100	13.17	1234.01	13.00	1240.62	12.94	1249.58	13.00	1246.07	13.08	1250.72
SDs		0.00	2.36	0.08	8.07	0.13	15.93	0.08	11.70	0.08	10.65
CVs		0.00	0.19	0.64	0.65	0.98	1.28	0.64	0.94	0.64	0.85
03 - AVGs	100	12.92	1243.44	13.08	1249.54	13.08	1242.02	13.00	1242.97	13.00	1252.13
SDs		0.17	16.33	0.17	9.94	0.14	9.37	0.08	7.62	0.00	11.02
CVs		1.29	1.31	1.27	0.80	1.10	0.75	0.64	0.61	0.00	0.88
04 - AVGs	100	12.92	1256.26	13.17	1251.06	13.11	1255.43	12.94	1254.39	13.11	1251.66
SDs		0.08	10.09	0.14	3.88	0.05	3.45	0.13	3.53	0.17	4.58
CVs		0.65	0.80	1.10	0.31	0.37	0.28	0.98	0.28	1.32	0.37
05 - AVGs	100	13.00	1243.22	13.17	1244.47	12.89	1253.62	13.03	1243.19	13.00	1244.91
SDs		0.00	2.19	0.14	13.91	0.24	8.52	0.13	7.28	0.14	10.74
CVs		0.00	0.18	1.10	1.12	1.87	0.68	0.98	0.59	1.11	0.86
06 - AVGs	100	12.94	1238.41	13.11	1241.01	13.08	1242.56	13.08	1246.95	12.94	1245.29
SDs		0.10	5.03	0.10	1.71	0.08	3.59	0.08	9.26	0.10	3.23
CVs		0.74	0.41	0.73	0.14	0.64	0.29	0.64	0.74	0.74	0.26
07 - AVGs	100	13.14	1231.37	12.97	1253.61	13.03	1268.69	12.89	1239.74	12.97	1245.44
SDs		0.10	1.97	0.05	7.75	0.13	29.31	0.05	9.02	0.05	9.29
CVs		0.73	0.16	0.37	0.62	0.98	2.31	0.37	0.73	0.37	0.75
08 - AVGs	100	13.17	1237.77	12.97	1250.71	13.11	1251.93	13.00	1243.18	13.11	1247.34
SDs		0.17	5.50	0.10	12.83	0.19	2.11	0.00	0.68	0.13	1.07
CVs		1.27	0.44	0.74	1.03	1.47	0.17	0.00	0.05	0.97	0.09
09 - AVGs	100	12.97	1241.46	13.03	1248.06	13.00	1246.74	13.08	1240.45	13.03	1248.27
SDs		0.10	7.02	0.10	10.02	0.08	2.84	0.14	3.81	0.05	2.21
CVs		0.74	0.57	0.74	0.80	0.64	0.23	1.10	0.31	0.37	0.18
10 - AVGs	100	12.97	1250.17	13.06	1256.17	12.86	1270.09	13.14	1244.72	12.94	1248.55
SDs		0.13	13.68	0.13	12.59	0.10	7.22	0.05	7.63	0.05	8.38
CVs		0.98	1.09	0.98	1.00	0.75	0.57	0.37	0.61	0.37	0.67
AVGs	100	13.01	1241.52	13.05	1248.20	13.02	1252.81	13.02	1244.00	13.03	1247.74
SDs		0.11	7.36	0.08	5.10	0.09	9.76	0.07	4.55	0.06	3.08
CVs		0.83	0.59	0.65	0.41	0.70	0.78	0.56	0.37	0.49	0.25
% to DACS 03	100	0.00	0.00	0.30	0.54	0.06	0.91	0.11	0.20	0.13	0.50

Soon, the HLS catches the solution of an ant with the least unvisited customers and tries to insert such unvisited customers into the feasible parts in that infeasible solution.

According to the Student's t-test and the Wilcoxon signed rank test done on the NV results of Table 4.17 related to DACS 03 and DACS+HLS (on the problem set R1 of the problem group PG100 in Section 2.2) after 100 seconds, the addition of the hybrid local search HLS is successful at the 99.9% level and has led on average into a significant improvement. However in terms of the TD results, the statistical tests, mentioned earlier, show a significant deterioration on average. As in Figure 4.12, If a closer look is directed towards to what happens during the running of DACS 03 and DACS+HLS on R1 in the first 100 seconds of CPU time, the chart demonstrates on average that DACS+HLS is able considerably to reduce the NV values, as indicated from the light blue columns, more than the NV values (with the yellow columns) gained by DACS 03. However, reducing the NV values in DACS+HLS, on R1, comes at the expense of deteriorating the TD values with the pink lines, which are greater on average than the TD values with the dark blue lines of DACS 03.

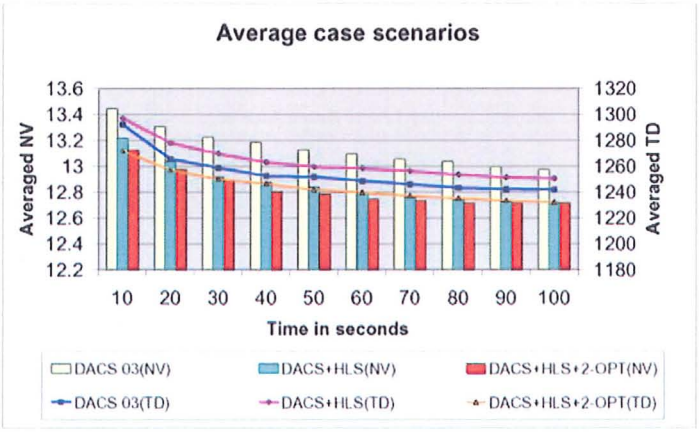


Figure 4.12: The average case scenarios of ten batches of three runs for the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt during the first 100 seconds of CPU time on the problem set R1 of the problem group PG100.

Thereafter discovering the significance of the system DACS+HLS in terms of the number of vehicles, DACS+HLS is tested for thirty runs, according to the methodology of experimentation in Section 4.2, on all the six problem sets of the problem group PG100 at six different amounts of CPU times in seconds - 100, 300, 400, 600,

Table 4.17: The systems DACS 03, DACS+HLS and DACS+HLS+2-Opt tested, after thirty runs of 100 seconds or in ten batches of three runs, on the problem set R1 of the problem group PG100.

		DACS 03		DACS+HLS		DACS+HLS+2-Opt	
Batch No.	Time(secs.)	NV	TD	NV	TD	NV	TD
01 - AVGs	100	12.92	1234.74	12.72	1242.82	12.72	1230.32
SDs		0.08	5.38	0.10	3.16	0.13	3.89
CVs		0.65	0.44	0.76	0.25	1.00	0.32
02 - AVGs	100	12.89	1243.28	12.67	1255.71	12.75	1235.05
SDs		0.05	7.76	0.00	8.36	0.14	4.77
CVs		0.37	0.62	0.00	0.67	1.13	0.39
03 - AVGs	100	12.97	1236.31	12.64	1253.91	12.67	1231.88
SDs		0.17	8.35	0.13	2.71	0.08	4.78
CVs		1.34	0.68	1.01	0.22	0.66	0.39
04 - AVGs	100	13.00	1242.98	12.72	1245.95	12.64	1227.55
SDs		0.08	3.50	0.05	6.34	0.05	1.33
CVs		0.64	0.28	0.38	0.51	0.38	0.11
05 - AVGs	100	12.94	1242.86	12.69	1248.43	12.75	1240.72
SDs		0.13	1.06	0.05	6.81	0.17	9.52
CVs		0.98	0.09	0.38	0.55	1.31	0.77
06 - AVGs	100	13.00	1241.20	12.72	1245.28	12.78	1230.42
SDs		0.08	3.81	0.05	2.77	0.05	3.23
CVs		0.64	0.31	0.38	0.22	0.38	0.26
07 - AVGs	100	12.92	1248.39	12.72	1242.63	12.69	1230.65
SDs		0.08	7.71	0.10	8.71	0.05	6.88
CVs		0.65	0.62	0.76	0.70	0.38	0.56
08 - AVGs	100	12.92	1246.08	12.67	1255.27	12.72	1226.55
SDs		0.08	6.60	0.08	3.28	0.13	7.36
CVs		0.65	0.53	0.66	0.26	1.00	0.60
09 - AVGs	100	12.86	1238.52	12.64	1253.52	12.69	1229.89
SDs		0.05	7.76	0.13	13.93	0.05	1.04
CVs		0.37	0.63	1.01	1.11	0.38	0.08
10 - AVGs	100	12.86	1239.45	12.78	1247.47	12.72	1225.22
SDs		0.13	11.42	0.10	15.85	0.05	5.16
CVs		0.99	0.92	0.75	1.27	0.38	0.42
AVGs	100	12.93	1241.38	12.70	1249.10	12.71	1230.83
SDs		0.05	4.23	0.04	5.09	0.04	4.46
CVs		0.39	0.34	0.35	0.41	0.33	0.36
% to DACS 03	100	0.00	0.00	-1.78	0.62	-1.65	-0.85
% to DACS+HLS	100	-	-	0.00	0.00	0.13	-1.46

Table 4.18: Comparison between the average case performances of the algorithms DACS+HLS and DACS 03, after thirty runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
DACS+HLS - AVGs	100	12.70	1248.05	10.00	838.26	12.32	1416.72	3.04	1003.71	3.00	593.97	3.38	1194.72
SDs		0.07	7.14	0.00	8.79	0.12	10.84	0.05	9.66	0.00	6.19	0.03	14.04
% to DACS 03	100	-1.99	0.41	0.00	0.41	-2.35	0.24	-3.74	1.24	0.00	0.26	-6.14	2.15
% to MACS-VRPTW [4]	100	1.22	2.74	0.00	1.19	-1.15	1.52	-0.25	3.27	0.00	0.13	-0.15	0.24
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
SDs		0.09	7.18	0.00	6.65	0.11	8.19	0.04	6.12	0.00	5.73	0.02	14.42
% to DACS 03	300	-1.31	0.48	0.00	0.29	-2.58	0.33	-4.16	1.42	0.00	0.18	-2.53	0.94
% to MACS-VRPTW [4]	300	1.20	2.08	0.00	0.70	0.06	1.40	0.00	1.02	0.00	0.05	1.23	0.43
DACS+HLS - AVGs	400	12.59	1234.74	10.00	832.94	12.11	1407.35	2.99	973.95	3.00	593.18	3.37	1170.73
SDs		0.09	6.95	0.00	5.87	0.11	8.18	0.04	5.93	0.00	5.76	0.02	14.59
% to DACS 03	400	-1.18	0.45	0.00	0.19	-2.38	0.45	-3.52	1.13	0.00	0.17	-1.94	0.72
% to MACS-VRPTW [4]	300	1.12	1.80	0.00	0.55	-0.18	1.31	-0.30	0.50	0.00	0.04	1.23	0.20
DACS+HLS - AVGs	600	12.57	1231.17	10.00	832.46	12.09	1405.07	2.97	970.94	3.00	593.13	3.36	1171.01
SDs		0.08	7.31	0.00	5.65	0.11	8.19	0.05	6.71	0.00	5.78	0.04	15.59
% to DACS 03	600	-1.07	0.42	0.00	0.17	-2.26	0.52	-3.44	1.02	0.00	0.18	-1.71	0.72
% to MACS-VRPTW [4]	600	1.51	1.47	0.00	0.49	0.10	1.79	-0.91	0.58	0.00	0.04	0.98	0.68
DACS+HLS - AVGs	1200	12.52	1228.43	10.00	831.81	12.07	1400.25	2.95	969.61	3.00	592.84	3.35	1171.72
SDs		0.07	8.15	0.00	4.95	0.10	8.51	0.06	8.07	0.00	5.76	0.05	19.12
% to DACS 03	1200	-0.86	0.40	0.00	0.13	-2.03	0.53	-3.08	1.05	0.00	0.17	-1.71	1.03
% to MACS-VRPTW [4]	1200	1.13	1.39	0.00	0.41	0.89	1.05	-1.62	0.78	0.00	0.14	0.48	1.57
DACS+HLS - AVGs	1800	12.51	1224.75	10.00	831.42	12.04	1398.10	2.92	970.87	3.00	592.65	3.33	1173.56
SDs		0.07	7.09	0.00	4.19	0.08	8.32	0.08	8.73	0.00	5.81	0.06	19.24
% to DACS 03	1800	-0.71	0.27	0.00	0.10	-2.03	0.52	-3.79	1.29	0.00	0.14	-1.72	1.27
% to MACS-VRPTW [4]	1800	1.06	1.15	0.00	0.37	1.02	0.72	-2.53	1.10	0.00	0.14	0.10	2.11
DACS 03 - AVGs	100	12.96	1242.95	10.00	834.85	12.61	1413.30	3.16	991.42	3.00	592.44	3.60	1169.58
	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
	400	12.74	1229.23	10.00	831.32	12.40	1401.11	3.10	963.04	3.00	592.16	3.44	1162.37
	600	12.70	1225.99	10.00	831.08	12.37	1397.75	3.08	961.14	3.00	592.05	3.42	1162.62
	1200	12.63	1223.57	10.00	830.75	12.32	1392.88	3.05	959.58	3.00	591.84	3.40	1159.78
	1800	12.60	1221.49	10.00	830.60	12.29	1390.92	3.04	958.53	3.00	591.82	3.39	1158.81
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.35	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.19: Comparison between the average case performances of DACS+HLS and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100. Check Tables E.1 and E.2 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.00	1677.74	10.00	828.94	14.97	1697.28	4.00	1287.35	3.00	591.56	4.00	1486.83
SDs		0.00	11.89	0.00	0.00	0.18	34.72	0.00	19.62	0.00	0.00	0.00	43.01
% to DACS 03	1800	0.00	0.07	0.00	0.00	0.00	-0.07	0.00	-0.15	0.00	0.00	0.00	0.54
02 - AVGs	1800	17.30	1492.95	10.00	842.42	13.00	1524.46	3.00	1219.81	3.00	591.56	3.67	1286.82
SDs		0.47	9.96	0.00	27.63	0.00	16.17	0.00	19.49	0.00	0.00	0.48	122.75
% to DACS 03	1800	0.58	0.07	0.00	0.86	0.00	-0.07	-10.89	3.09	0.00	0.00	-8.33	7.60
03 - AVGs	1800	13.37	1281.29	10.00	835.28	11.10	1307.12	3.00	967.30	3.00	599.97	3.00	1126.84
SDs		0.49	39.85	0.00	19.75	0.31	31.34	0.00	13.18	0.00	37.85	0.00	29.44
% to DACS 03	1800	-0.99	0.60	0.00	0.50	-2.35	-0.09	0.00	0.03	0.00	0.92	0.00	0.52
04 - AVGs	1800	10.00	1006.83	10.00	825.08	10.00	1173.08	2.60	818.06	3.00	604.16	3.00	825.02
SDs		0.00	12.40	0.00	0.96	0.00	20.03	0.50	71.56	0.00	29.49	0.00	14.96
% to DACS 03	1800	-1.64	-0.64	0.00	0.02	-5.06	1.04	-13.33	6.19	0.00	0.23	0.00	-0.50
05 - AVGs	1800	14.00	1440.44	10.00	830.42	14.03	1606.70	3.00	1063.76	3.00	588.88	4.00	1380.16
SDs		0.00	21.11	0.00	8.10	0.18	23.64	0.00	24.06	0.00	0.00	0.00	34.18
% to DACS 03	1800	-0.24	0.35	0.00	0.04	-2.55	1.30	0.00	0.08	0.00	0.00	0.00	-0.25
06 - AVGs	1800	12.03	1288.64	10.00	830.39	12.00	1428.97	3.00	943.69	3.00	588.49	3.00	1247.03
SDs		0.18	14.71	0.00	7.92	0.00	18.75	0.00	17.22	0.00	0.00	0.00	45.93
% to DACS 03	1800	-1.37	0.09	0.00	0.17	-0.28	-0.23	0.00	0.34	0.00	0.00	-3.23	0.42
07 - AVGs	1800	10.77	1101.59	10.00	831.26	11.00	1275.83	2.67	873.28	3.00	588.29	3.00	1166.13
SDs		0.43	19.97	0.00	8.82	0.00	38.85	0.48	57.54	0.00	0.00	0.00	37.56
% to DACS 03	1800	-1.82	0.25	0.00	0.00	-1.79	0.78	-11.11	4.31	0.00	-0.02	-1.10	1.70
08 - AVGs	1800	9.93	966.63	10.00	830.08	10.23	1171.40	2.00	760.89	3.00	588.32	3.00	869.68
SDs		0.25	10.20	0.00	6.27	0.43	23.93	0.00	19.71	0.00	0.00	0.00	22.84
% to DACS 03	1800	-0.67	0.24	0.00	-0.71	-5.54	1.88	-3.23	0.90	0.00	0.00	0.00	-0.44
09 - AVGs	1800	11.80	1212.74	10.00	828.94	-	-	3.00	956.72	-	-	-	-
SDs		0.41	40.95	0.00	0.00	-	-	0.00	17.04	-	-	-	-
% to DACS 03	1800	-1.67	1.16	0.00	0.00	-	-	0.00	0.33	-	-	-	-
10 - AVGs	1800	10.97	1134.56	-	-	-	-	3.00	975.99	-	-	-	-
SDs		0.18	22.97	-	-	-	-	0.00	12.85	-	-	-	-
% to DACS 03	1800	-1.20	0.45	-	-	-	-	0.00	-0.87	-	-	-	-
11 - AVGs	1800	11.00	1102.59	-	-	-	-	2.90	812.76	-	-	-	-
SDs		0.00	19.29	-	-	-	-	0.31	45.35	-	-	-	-
% to DACS 03	1800	-0.30	0.10	-	-	-	-	-3.33	1.46	-	-	-	-
12 - AVGs	1800	9.97	991.04	-	-	-	-	-	-	-	-	-	-
SDs		0.18	16.00	-	-	-	-	-	-	-	-	-	-
% to DACS 03	1800	-0.33	0.43	-	-	-	-	-	-	-	-	-	-
DACS+HLS - AVGs	1800	12.51	1224.75	10.00	831.42	12.04	1398.10	2.92	970.87	3.00	592.65	3.33	1173.56
SDs		0.07	7.09	0.00	4.19	0.08	8.32	0.08	8.73	0.00	5.81	0.06	19.24
% to DACS 03	1800	-0.71	0.27	0.00	0.10	-2.03	0.52	-3.79	1.29	0.00	0.14	-1.72	1.27
% to SA+LNS [29]	1800	2.13	1.35	0.00	0.37	2.05	2.00	2.60	-1.62	0.00	-2.75	0.10	0.54
DACS 03 - AVGs	1800	12.60	1221.49	10.00	830.60	12.29	1390.92	3.04	958.53	3.00	591.82	3.39	1158.81
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

1200 and 1800. From Tables 4.18 and 4.19, it can be seen that adding the hybrid local search HLS into the DACS 03 system is a significant one and has led into reducing the number of vehicles NV, on average by -2.43%, on the problem sets R1, RC1, R2 and RC2 and into improving the performance during all the six different CPU times in seconds. The improvement in the performance, especially in terms of NV, is indicated from the numeric results colored with red. However on all the six problem sets, adding HLS has led also into having bad traveled distances TDs on average with 0.59% if such results are compared with the TD results of DACS 03.

On the problem sets R1, R2 and RC2 also and in comparison to the system DACS 03, the best and worst case scenarios of the performance of the DACS+HLS system, at the six different amounts of CPU time, have improved greatly on average in terms of NV in a way that ranges between -4.33% to -4.83% and have got worse in terms of TD by 1.97% to 2.64%. In relation to the problem set RC1, only the worst case scenario is improved, by -2.27% on average, in terms of NV and the best case scenario has not changed from that of DACS 03. Moreover when it comes to the TD result of RC1 in the best case scenario, the DACS+HLS system has deteriorated it slightly by 0.22% on average at the six different amounts of CPU time. However in the worst case scenario on RC1, the TD result, on average and by 1.51%, has worsen very much so.

The addition of HLS to the DACS system has made the performance and results on the problem group PG100 much more competitive in terms of NV, by -0.73% on average, to those of other VRPTW algorithms in the literature as in Table 4.3. However in terms of TD, the situation is deteriorated on average, by 0.35% on PG100, for now. In the case where there is no use for HLS, DACS 03 is far on average, by 0.94% for NV and -0.25% for TD, from other VRPTW algorithms and this shows the difference between DACS 03 and DACS+HLS.

In comparison to the MACS+VRPTW system [4], DACS+HLS is nearer in terms of NV and departed from it on average, by 0.18% and 0.98% for NV and TD respectively. Also, it is much closer, after 1800 seconds, to the algorithms LS [28] and LS+TA [28] on average (with 2.80% for NV and -0.24% for TD) and the algorithm SA+LNS [29], with 1.15% for NV and -0.02% for TD. Subsequently, adding the hybrid local search HLS is a good thing and leaving it out would make the performance and results a lot worse, on PG100, in terms of the number of vehicles.

4.7.4 What if a variant of a 2-Opt move is inserted in local searches?

After the significant improvement in terms of the number of vehicles in Section 4.7.3 because of adding the hybrid local search HLS into the system DACS 03 in Section 4.7.1, the system DACS+HLS needs to improve on average in terms of the total of traveled distances. In order to make that improvement happen, a variant of a 2-Opt move as in Figure 2.2 or the move M4 in Section 4.4.8 is injected in the local search of triple moves in Sections 4.6.2 and 4.6.1 and the HLS component itself.

The main purpose of the 2-Opt move variant is to reduce the total of traveled distances done by two routes especially after discovering from Table B.8 on the problem set R1 of the problem group PG100 that it is possible for it on average by -1.49% to help in bringing down the traveled distances in 300 to 400 seconds. In Section 4.5.6, it was not possible for 2-Opt* to work on its own individually and to move forward the performance in a way to let the DACS system look competitive or at least as good as the other VRPTW algorithms in the literature.

As a result as in Table 4.17, inserting the 2-Opt move variant to the local searches of triple moves and HLS is successful on R1 after 100 seconds and has lead into having a significant improvement on average at the 99.9% level in terms of the traveled distances according to the Student's t-test and the Wilcoxon signed rank test on the TD results, related to the systems DACS+HLS and DACS+HLS+2-Opt.

From the orange line of DACS+HLS+2-Opt in Figure 4.12, it can be perceived during the first 100 seconds that the TD values at hand on R1 are considerably less, on average, than the TD values of the pink line achieved by DACS+HLS. Moreover after inserting 2-Opt*, the TD values in Table 4.17 are still significant on average at the 99.9% level in comparison to those of DACS 03, on R1, according to the statistical tests described earlier in the previous paragraph and this is indicated in Figure 4.12 also from the good position of the orange line beneath the dark blue line.

Of course also, the addition of this 2-Opt move variant has not led on average into any significant change or reduction on R1 in terms of the number of vehicles after 100 seconds. Therefore in Figure 4.12, the NV values with the red columns of DACS+HLS+2-Opt on R1 are still as significant on average as those with the light

blue columns of DACS+HLS at the 99.9% and this is in comparison with those of the system DACS 03 with the yellow columns and according to the statistical tests of Student's t-test and Wilcoxon signed rank on the NV values in Table 4.17.

Then, the encouraging discovery of how significant the addition of the 2-Opt move variant is has led into testing the DACS+HLS+2-Opt system on the six problem sets of the problem group PG100 in Section 2.2 for thirty runs according to the experimental methodology in Section 4.2 - at six different amounts of CPU time in seconds, which are 100, 300, 400, 600, 1200 and 1800. It can be seen from Tables 4.20 and 4.21 that the addition of the 2-Opt move variant into the DACS+HLS system has led into improving the TD results, on average by -1.14%, on all the six problem sets of PG100 at all the six different CPU times in seconds, as indicated by the numeric results colored with blue. In comparison to the TD results of DACS 03, the TD results of DACS+HLS+2-Opt are improved on average by -0.56% likewise for now.

Even though the NV results have deteriorated on average by 0.05% on PG100 after including 2-Opt*, however DACS+HLS+2-Opt is still getting NVs that are better, with -1.57%, than the NV results of DACS 03 and therefore the reduction of the number of vehicles has not significantly changed from the same kind of reduction in DACS+HLS. For instance on the problem R2, the NV result might get worse on average by 0.83% in DACS+HLS+2-Opt but in comparison to that of DACS 03 it is still improved with -2.82%.

In the DACS+HLS+2-Opt system, the best and worst case scenarios of the performance have advanced on average also in terms of TD by -0.51% and -3.57% in comparison of those scenarios of DACS+HLS. On the other hand, such case scenarios have become, on average, slightly worse in terms of NV, by 0.34% and 0.01% respectively, but they are regarded though as enhanced by -1.85% and -2.79% if compared with those scenarios of the system DACS 03. Now in comparison to the best case scenario of DACS 03 at the six different amounts of CPU time, DACS+HLS+2-Opt gets poorer in terms of TD with average equal to 0.75%, only on the problem sets R1, RC1, R2 and RC2. Also, it gets worse on the problem sets R1, RC1 and RC2, by 0.22% on average, in the worst case scenario. However on the problem sets C1, R2 and C2 only, it manages in the worst case scenario to improve the TD results by -1.95% on average.

Table 4.20: Comparison between the average case performances of the algorithms DACS+HLS+2-Opt and DACS+HLS, after thirty runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
DACS+HLS+2-Opt - AVGs	100	12.73	1233.54	10.00	830.28	12.27	1400.50	3.06	975.38	3.00	591.59	3.37	1157.51
SDs		0.07	6.04	0.00	1.53	0.12	10.61	0.05	6.72	0.00	2.12	0.02	12.53
% to DACS+HLS	100	0.17	-1.16	0.00	-0.95	-0.41	-1.14	0.50	-2.82	0.00	-0.40	-0.12	-3.11
% to MACS-VRPTW [4]	100	1.39	1.54	0.00	0.23	-1.55	0.36	0.25	0.35	0.00	-0.27	-0.27	-2.88
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36
SDs		0.09	6.43	0.00	0.48	0.09	9.43	0.05	7.35	0.00	2.09	0.02	10.97
% to DACS+HLS	300	0.29	-1.12	0.00	-0.64	-0.21	-0.92	0.81	-2.03	0.00	-0.33	0.00	-2.64
% to MACS-VRPTW [4]	300	1.49	0.93	0.00	0.06	-0.14	0.46	0.81	-1.03	0.00	-0.28	1.23	-2.22
DACS+HLS+2-Opt - AVGs	400	12.61	1223.69	10.00	828.75	12.08	1394.97	3.02	956.45	3.00	591.27	3.37	1142.05
SDs		0.10	6.81	0.00	0.41	0.09	9.64	0.06	7.03	0.00	2.09	0.03	11.56
% to DACS+HLS	400	0.18	-0.90	0.00	-0.50	-0.24	-0.88	0.91	-1.80	0.00	-0.32	-0.12	-2.45
% to MACS-VRPTW [4]	300	1.29	0.89	0.00	0.05	-0.42	0.42	0.61	-1.30	0.00	-0.29	1.10	-2.25
DACS+HLS+2-Opt - AVGs	600	12.56	1222.67	10.00	828.66	12.05	1393.25	3.01	955.06	3.00	591.17	3.35	1145.88
SDs		0.10	6.39	0.00	0.37	0.09	9.25	0.06	7.43	0.00	2.10	0.05	16.81
% to DACS+HLS	600	-0.07	-0.69	0.00	-0.46	-0.34	-0.84	1.12	-1.64	0.00	-0.33	-0.50	-2.15
% to MACS-VRPTW [4]	600	1.44	0.77	0.00	0.03	-0.25	0.93	0.20	-1.07	0.00	-0.29	0.48	-1.48
DACS+HLS+2-Opt - AVGs	1200	12.48	1221.27	10.00	828.54	12.01	1389.85	2.97	956.62	3.00	591.11	3.34	1145.70
SDs		0.09	6.29	0.00	0.27	0.03	8.06	0.05	8.62	0.00	2.12	0.06	17.37
% to DACS+HLS	1200	-0.33	-0.58	0.00	-0.39	-0.48	-0.74	0.72	-1.34	0.00	-0.29	-0.12	-2.22
% to MACS-VRPTW [4]	1200	0.79	0.79	0.00	0.02	0.40	0.30	-0.91	-0.57	0.00	-0.16	0.35	-0.69
DACS+HLS+2-Opt - AVGs	1800	12.43	1222.20	10.00	828.47	12.00	1387.21	2.95	958.23	3.00	590.69	3.34	1145.99
SDs		0.08	5.81	0.00	0.19	0.00	7.25	0.07	10.45	0.00	0.97	0.06	19.07
% to DACS+HLS	1800	-0.67	-0.21	0.00	-0.36	-0.35	-0.78	0.93	-1.30	0.00	-0.33	0.12	-2.35
% to MACS-VRPTW [4]	1800	0.39	0.94	0.00	0.01	0.67	-0.07	-1.62	-0.22	0.00	-0.20	0.23	-0.29
DACS+HLS - AVGs	100	12.70	1248.05	10.00	838.26	12.32	1416.72	3.04	1003.71	3.00	593.97	3.38	1194.72
	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
	400	12.59	1234.74	10.00	832.94	12.11	1407.35	2.99	973.95	3.00	593.18	3.37	1170.73
	600	12.57	1231.17	10.00	832.46	12.09	1405.07	2.97	970.94	3.00	593.13	3.36	1171.01
	1200	12.52	1228.43	10.00	831.81	12.07	1400.25	2.95	969.61	3.00	592.84	3.35	1171.72
	1800	12.51	1224.75	10.00	831.42	12.04	1398.10	2.92	970.87	3.00	592.65	3.33	1173.56
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

Table 4.21: Comparison between the average case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100. Check Tables F.1 and F.2 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.00	1671.08	10.00	828.94	14.97	1697.15	4.00	1266.28	3.00	591.56	4.00	1439.23
SDs		0.00	8.19	0.00	0.00	0.18	31.04	0.00	9.84	0.00	0.00	0.00	21.59
% to DACS+HLS	1800	0.00	-0.40	0.00	0.00	0.00	-0.01	0.00	-1.64	0.00	0.00	0.00	-3.20
02 - AVGs	1800	17.20	1488.93	10.00	829.13	13.00	1507.70	3.10	1208.53	3.00	591.56	3.70	1239.30
SDs		0.41	7.96	0.00	1.04	0.00	11.97	0.31	45.34	0.00	0.00	0.47	118.61
% to DACS+HLS	1800	-0.58	-0.27	0.00	-1.58	0.00	-1.10	3.33	-0.92	0.00	0.00	0.91	-3.69
03 - AVGs	1800	13.20	1286.18	10.00	828.61	11.00	1293.70	3.00	958.73	3.00	591.47	3.00	1113.27
SDs		0.41	32.28	0.00	1.43	0.00	33.51	0.00	8.67	0.00	1.65	0.00	21.42
% to DACS+HLS	1800	-1.25	0.38	0.00	-0.80	-0.90	-1.03	0.00	-0.89	0.00	-1.42	0.00	-1.20
04 - AVGs	1800	10.00	999.88	10.00	824.83	10.00	1159.59	2.80	785.14	3.00	596.92	3.00	821.71
SDs		0.00	12.12	0.00	0.19	0.00	13.28	0.41	46.66	0.00	7.73	0.00	12.89
% to DACS+HLS	1800	0.00	-0.69	0.00	-0.03	0.00	-1.15	7.69	-4.02	0.00	-1.20	0.00	-0.40
05 - AVGs	1800	14.00	1419.89	10.00	828.94	14.03	1585.50	3.00	1046.24	3.00	588.88	4.00	1342.30
SDs		0.00	15.67	0.00	0.00	0.18	25.46	0.00	19.42	0.00	0.00	0.00	22.31
% to DACS+HLS	1800	0.00	-1.43	0.00	-0.18	0.00	-1.32	0.00	-1.65	0.00	0.00	0.00	-2.74
06 - AVGs	1800	12.00	1283.39	10.00	828.94	12.00	1422.65	3.00	932.81	3.00	588.49	3.00	1215.70
SDs		0.00	14.00	0.00	0.00	0.00	15.95	0.00	13.38	0.00	0.00	0.00	37.11
% to DACS+HLS	1800	-0.28	-0.41	0.00	-0.17	0.00	-0.44	0.00	-1.15	0.00	0.00	0.00	-2.51
07 - AVGs	1800	10.50	1111.14	10.00	828.94	11.00	1250.53	2.70	860.46	3.00	588.29	3.00	1141.41
SDs		0.51	24.57	0.00	0.00	0.00	20.24	0.47	52.20	0.00	0.00	0.00	29.93
% to DACS+HLS	1800	-2.48	0.87	0.00	-0.28	0.00	-1.98	1.25	-1.47	0.00	0.00	0.00	-2.12
08 - AVGs	1800	9.93	963.32	10.00	828.94	10.00	1180.86	2.00	753.36	3.00	588.32	3.00	854.98
SDs		0.25	8.82	0.00	0.00	0.00	16.96	0.00	17.41	0.00	0.00	0.00	17.59
% to DACS+HLS	1800	0.00	-0.34	0.00	-0.14	-2.28	0.81	0.00	-0.99	0.00	0.00	0.00	-1.69
09 - AVGs	1800	11.43	1223.36	10.00	828.94	-	-	3.00	945.75	-	-	-	-
SDs		0.50	37.18	0.00	0.00	-	-	0.00	14.87	-	-	-	-
% to DACS+HLS	1800	-3.11	0.88	0.00	0.00	-	-	0.00	-1.15	-	-	-	-
10 - AVGs	1800	10.90	1131.22	-	-	-	-	3.00	971.48	-	-	-	-
SDs		0.31	24.29	-	-	-	-	0.00	8.33	-	-	-	-
% to DACS+HLS	1800	-0.61	-0.29	-	-	-	-	0.00	-0.46	-	-	-	-
11 - AVGs	1800	10.97	1097.85	-	-	-	-	2.87	811.74	-	-	-	-
SDs		0.18	26.11	-	-	-	-	0.35	53.96	-	-	-	-
% to DACS+HLS	1800	-0.30	-0.43	-	-	-	-	-1.15	-0.12	-	-	-	-
12 - AVGs	1800	10.00	990.20	-	-	-	-	-	-	-	-	-	-
SDs		0.00	18.71	-	-	-	-	-	-	-	-	-	-
% to DACS+HLS	1800	0.33	-0.08	-	-	-	-	-	-	-	-	-	-
DACS+HLS+2-Opt - AVGs	1800	12.43	1222.20	10.00	828.47	12.00	1387.21	2.95	958.23	3.00	590.69	3.34	1145.99
SDs		0.08	5.81	0.00	0.19	0.00	7.25	0.07	10.45	0.00	0.97	0.06	19.07
% to DACS+HLS	1800	-0.67	-0.21	0.00	-0.36	-0.35	-0.78	0.93	-1.30	0.00	-0.33	0.12	-2.35
% to SA+LNS [29]	1800	1.45	1.14	0.00	0.01	1.69	1.20	3.56	-2.91	0.00	-3.07	0.23	-1.82
DACS+HLS - AVGs	1800	12.51	1224.75	10.00	831.42	12.04	1398.10	2.92	970.87	3.00	592.65	3.33	1173.56
SA+LNS [29] -AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24
	7200	12.03	1213.50	10.00	828.38	11.63	1380.06	2.73	985.36	3.00	591.85	3.28	1156.39
MACS-VRPTW [4]-AVGs	100	12.55	1214.80	10.00	828.40	12.46	1395.47	3.05	971.97	3.00	593.19	3.38	1191.87
	300	12.45	1212.95	10.00	828.38	12.13	1389.15	3.00	969.09	3.00	592.97	3.33	1168.34
	600	12.38	1213.35	10.00	828.38	12.08	1380.38	3.00	965.37	3.00	592.89	3.33	1163.08
	1200	12.38	1211.64	10.00	828.38	11.96	1385.65	3.00	962.07	3.00	592.04	3.33	1153.63
	1800	12.38	1210.83	10.00	828.38	11.92	1388.13	3.00	960.31	3.00	591.85	3.33	1149.28
HGA [28] - AVGs	1800	12.17	1243.72	10.00	828.71	11.88	1399.76	2.73	1025.80	3.00	597.84	3.25	1255.22
HGA+TA [28] - AVGs	2094	12.17	1215.23	10.00	828.38	11.88	1380.55	2.73	975.93	3.00	589.86	3.25	1170.85
LS [28] - AVGs	126	12.08	1247.12	10.00	828.50	11.63	1418.53	2.73	998.70	3.00	590.30	3.25	1171.75
LS+TA [28] - AVGs	156	12.08	1222.69	10.00	828.38	11.63	1398.83	2.73	977.28	3.00	589.86	3.25	1150.48

In relation to other VRPTW algorithms in the literature as in Table 4.3, using the 2-Opt move has resulted in enhancing the traveled distances, on the problem group PG100, with average equal to -0.81% if compared with the traveled distances gained by DACS+HLS, which are 0.35% up. However in terms of the number of vehicles on PG100, the system that uses the HLS only gets a percentage of deviation equal to -0.73% on average, which is slightly better than that -0.68% of DACS+HLS+2-Opt. Furthermore, the percentage of deviations of NV and TD, related to the system using 2-Opt, are better on average than those of the system DACS 03, which are 0.94% for NV and -0.25% for TD.

For that, the results of DACS+HLS+2-Opt are getting closer in terms of TD to the results of the system MACS-VRPTW [4] and they are deviated on average by 0.22% for NV and -0.18% for TD. Also although the TD results after 1800 seconds are closer on average with -1.12% in comparison to the algorithms LS [28] and LS+TA [28], however the NV results of the system using 2-Opt are worse on average by 2.81%, which is almost similar to the percentage deviation of 2.80% of DACS+HLS. In contrast to SA+LNS [29], the system applying 2-Opt is beaten, on average by 1.16%, in terms of NV and this is nearly same as that faced by DACS+HLS but with 1.15%. Nevertheless in DACS+HLS+2-opt, the TD results are much enhanced and are departed on average by -0.91% in comparison to SA+LNS. As a conclusion, including 2-Opt improves things in terms of the traveled distances and a system without it will have the disadvantage of having poor TD results.

4.7.5 What the components PFPBS, HLS and 2-Opt can do on the problem groups PG200 and PG400?

In this section, it is worthwhile to see the effects of the push-forward and push-backward strategy PFPBS, the hybrid local search HLS and the 2-Opt move variant in Sections 4.7.1, 4.7.3 and 4.7.4 on the problem groups PG200 and PG400 in Section 2.2 after discovering their significance, in terms of performance and results as in the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt, on the problem group PG100 in Section 2.2.

The performance and the results of such components are more of the same thing but on the problem groups PG200 and PG400. Accordingly, the three systems

DACS 03, DACS+HLS and DACS+HLS+2-Opt are tested each, on each one of the problem groups PG200 and PG400, for three runs of 2400 or 4800 seconds according to the methodology of experimentation in Section 4.2.

Once the PFPBS strategy in Section 4.4.11 is added as in the system DACS 03, the performance and the results have improved, on average by -2.81% for NV and by -5.17% for TD, on all the six problem sets of PG200 as in Table 4.22. On PG400, the same kind of progress happens in Table 4.23 on all the six problem sets but on average with -2.16% for NV and -10.76% for TD. Furthermore, the best and worst case scenarios of the performance have advanced too.

In comparison to the performance and results of the DACS 02 system in Section 4.6.2 that does not use that strategy, DACS 03 is closing a lot on average to other VRPTW algorithms in the literature mentioned in Tables 4.12 and 4.13 and it is deviated now, from such algorithms, on PG200 by 1.66% and 4.29% for NV and TD respectively where as on PG400 it is distant with 3.13% and 9.19%. For instance, DACS 03 is outperformed on average by the algorithm RVNSc [5] on PG200 usually with 2.08% for NV and 3.79% for TD. Also on PG400 for example, the algorithm ES4C [2] beats DACS 03 on average by 3.80% and 9.35%, which are not as worse as those percentage of deviations gained usually by DACS 02.

Later after the addition of the HLS in Section 4.4.10 as in the system DACS+HLS, it can be seen, from Tables 4.24 and 4.25, in comparison to DACS 03 that the HLS has led into improving the performance and results in terms of the number of vehicles NVs on average, by -1.02% for PG200 and -1.92% for PG400. However in terms of the traveled distances TDs, the performance and results have deteriorated on average by 2.70% and 4.66% for PG200 and PG400 in that order. Generally speaking when it comes to the best and worst case scenarios, the same thing, described above, happens on PG200 and PG400. But in the best case scenario on PG200, the enhancement on average is done with -1.07% in terms of NV only on the problem sets C1 and RC1 while the NV values on the other problem sets are the same as in DACS 03.

Also, the HLS has led into reducing the NV results when compared to other VRPTW algorithms in the literature stated in Tables 4.12 and 4.13. The gap in terms of NV between DACS+HLS and other VRPTW algorithms is now 0.62% for PG200 and 1.09% for PG400, which are much better on average than those of DACS

03. For that in relation to the algorithm RVNSc [5], the percentage of deviation of the NV results, because of adding HLS, is on average equal to 1.04% on PG200 whereas DACS 03 manages to bring the gap equal to 2.08% in terms of NV. On PG400 in terms of NV, the gap between DACS+HLS and the algorithm ES4C [2] is reduced on average to 1.73% from the percentage deviation 3.80% of DACS 03. On the other hand, the addition of HLS have come at the expense of the TD results, which are worse (from those of DACS 03) on average by 7.11% for PG200 and 14.30% for PG400 in comparison to the other VRPTW algorithms and this is reflected again on the bad gap on average between DACS+HLS and algorithms like RVNSc and ES4C in terms of TD.

Then in order to improve the performance and results in terms of the total of traveled distances TDs, the 2-Opt move variant or the move M4, described in Section 4.4.8, is tested as in the system DACS+HLS+2-Opt on the problem groups PG200 and PG400. Of course as indicated from the numeric results colored with blue in Tables 4.26 and 4.27, the performance and results on average have improved dramatically in terms of TD, by -3.79% and -5.89% for PG200 and PG400 correspondingly and this kind of enhancement is also echoed in the best and worst case scenarios. However, the change in the number of vehicles, by -0.03% and -0.32%, on average is not that significant and therefore this is true also in the best and worst case scenarios.

When it comes to how much the performance and results is now departed, after injecting 2-Opt, from other VRPTW algorithms in the literature declared in Tables 4.12 and 4.13, it can be realized that the TD results on average are getting further down on PG200 and PG400 respectively from 7.11% and 14.30% in DACS+HLS to 3.03% and 7.53% in DACS+HLS+2-Opt. As a consequence, the algorithm RVNSc beats on average the system that uses 2-Opt on PG200 by only 2.54% for TD instead of 6.60%. As well, the algorithm ES4C outperforms on average the system that applies 2-Opt with 7.67% for TD as a substitute to 14.48%. In contrast, the NV results are down on average as well but they are insignificant.

As a conclusion to what is illustrated above, using the PFPBS strategy, the HLS local search and the 2-Opt move helps in letting the DACS system scales up to problem groups with large numbers of customers and without them things will get worse.

Table 4.22: Comparison between the average case performances of DACS 03 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200. Check Tables D.3 and D.4 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	2400	20.00	5417.47	20.00	2704.57	19.00	3760.92	5.00	4356.94	6.00	1959.23	6.00	3320.57
SDs		0.00	94.31	0.00	0.00	0.00	96.73	0.00	120.49	0.00	48.14	0.00	106.73
% to DACS 02	2400	-1.64	-4.40	0.00	-0.35	0.00	-6.04	0.00	0.82	0.00	-0.39	-14.29	-3.47
02 - AVGs	2400	18.33	4509.29	18.67	2934.04	18.00	3760.04	4.67	3572.00	6.00	1885.79	5.00	3005.75
SDs		0.58	241.32	0.58	109.73	0.00	55.45	0.58	144.43	0.00	21.19	0.00	49.21
% to DACS 02	2400	-3.51	-0.04	-3.45	-20.42	-5.26	1.54	-6.67	-5.32	0.00	-1.47	-11.76	-0.93
03 - AVGs	2400	18.00	3795.37	18.00	3047.89	18.00	3648.84	4.00	3057.36	6.00	1857.05	4.00	2751.06
SDs		0.00	9.83	0.00	123.79	0.00	54.79	0.00	27.29	0.00	16.41	0.00	58.99
% to DACS 02	2400	-5.26	-2.93	-5.26	-19.28	0.00	-16.89	0.00	-7.48	-10.00	-6.83	-20.00	-1.25
04 - AVGs	2400	18.00	3388.94	18.00	2892.72	18.00	3350.97	4.00	2151.76	6.33	1881.43	4.00	2304.99
SDs		0.00	98.22	0.00	109.76	0.00	99.30	0.00	32.03	0.58	17.56	0.00	94.44
% to DACS 02	2400	0.00	-11.79	-5.26	-6.20	0.00	-11.52	0.00	-8.49	-9.52	-3.30	0.00	-8.31
05 - AVGs	2400	18.33	4762.37	20.00	2706.01	19.00	3573.17	4.00	3780.97	6.00	1903.89	5.00	2970.99
SDs		0.58	401.49	0.00	6.86	0.00	73.84	0.00	115.31	0.00	35.23	0.00	96.87
% to DACS 02	2400	-3.51	0.48	0.00	-1.36	0.00	-6.57	0.00	1.79	0.00	-3.49	-6.25	-5.23
06 - AVGs	2400	18.00	4119.47	20.00	2758.22	18.33	4038.29	4.00	3090.24	6.00	1895.46	5.00	2875.61
SDs		0.00	87.42	0.00	54.87	0.58	373.16	0.00	17.04	0.00	30.97	0.00	34.46
% to DACS 02	2400	-3.57	-0.58	-1.64	-2.14	-3.51	5.01	0.00	-5.43	0.00	-9.59	0.00	-5.77
07 - AVGs	2400	18.00	3534.61	20.00	2705.00	18.00	4093.15	4.00	2731.60	6.00	1969.61	4.67	2795.26
SDs		0.00	33.64	0.00	6.86	0.00	106.87	0.00	101.43	0.00	59.72	0.58	218.30
% to DACS 02	2400	0.00	-18.51	0.00	-6.30	-5.26	9.87	0.00	-1.40	-5.26	2.22	0.00	-4.34
08 - AVGs	2400	18.00	3289.37	20.00	2697.66	18.00	3821.13	4.00	2096.58	6.00	1873.63	4.00	2589.05
SDs		0.00	65.22	0.00	1.61	0.00	120.79	0.00	56.79	0.00	18.92	0.00	69.97
% to DACS 02	2400	0.00	-8.42	0.00	-6.33	-5.26	9.48	0.00	-8.80	0.00	-2.76	0.00	-4.59
09 - AVGs	2400	18.00	4525.95	19.00	2735.37	18.00	3636.66	4.00	3225.64	6.00	1912.22	4.00	2440.46
SDs		0.00	100.50	0.00	40.28	0.00	91.96	0.00	20.32	0.00	87.53	0.00	66.45
% to DACS 02	2400	-5.26	2.56	0.00	-7.89	-3.57	-4.42	0.00	-6.79	-10.00	-3.85	0.00	-4.95
10 - AVGs	2400	18.00	3721.27	18.67	2755.01	18.00	3520.95	4.00	2837.72	6.00	1859.41	4.00	2308.76
SDs		0.00	77.72	0.58	114.22	0.00	60.28	0.00	61.26	0.00	26.88	0.00	52.21
% to DACS 02	2400	0.00	-10.98	0.00	-9.81	0.00	-18.88	0.00	-6.58	0.00	-11.91	0.00	-1.66
DACS 03 - AVGs	2400	18.27	4106.41	19.23	2793.65	18.23	3720.41	4.17	3090.08	6.03	1899.77	4.57	2736.25
SDs		0.06	38.00	0.06	11.17	0.06	31.35	0.06	28.34	0.06	13.88	0.06	40.48
% to DACS 02	2400	-2.32	-5.24	-1.54	-8.75	-2.32	-4.40	-0.79	-4.33	-3.72	-4.25	-6.16	-4.03
% to RVNSc [5]	720-1680	0.92	7.46	1.76	0.53	1.30	6.05	4.17	1.47	0.56	3.11	3.79	4.10
% to ES4C [2]	2400	0.37	10.83	1.76	0.42	1.30	4.65	4.17	1.15	0.56	2.91	6.20	2.29
DACS 02 - AVGs	2400	18.70	4333.50	19.53	3061.39	18.67	3891.48	4.20	3230.00	6.27	1984.20	4.87	2851.11
RVNSc [5] - AVGs	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2] - AVGs	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33] - AVGs	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48] - AVGs	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17] - AVGs	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17] - AVGs	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table 4.23: Comparison between the average case performances of DACS 03 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400. Check Tables D.5 and D.6 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	4800	40.00	11524.57	40.00	7166.97	37.00	9660.98	9.00	9986.43	12.00	4250.86	13.00	7531.97
SDs		0.00	119.62	0.00	12.65	0.00	246.58	0.00	294.60	0.00	76.69	0.00	131.78
% to DACS 02	4800	-2.44	-11.43	0.00	-4.88	-5.13	-4.87	-3.57	-6.40	0.00	-1.30	0.00	-6.47
02 - AVGs	4800	37.00	10255.54	37.33	8569.28	37.00	8772.68	8.00	8640.17	12.00	4138.15	11.33	6933.41
SDs		0.00	131.47	0.58	164.05	0.00	48.21	0.00	84.44	0.00	107.85	0.58	149.73
% to DACS 02	4800	-2.63	-17.89	-8.94	-36.62	-2.63	-13.30	-11.11	-8.92	-5.26	-20.04	-2.86	-13.38
03 - AVGs	4800	37.00	9250.18	37.33	9228.22	36.67	9231.20	8.00	7023.05	12.67	4293.78	9.33	5745.56
SDs		0.00	118.02	0.58	511.38	0.58	1036.83	0.00	29.55	0.58	176.63	0.58	123.51
% to DACS 02	4800	-1.77	-10.70	-1.75	-25.43	-0.90	-5.75	0.00	-8.79	-2.56	-25.16	0.00	-10.40
04 - AVGs	4800	36.00	9618.90	37.00	8269.47	36.00	9493.50	8.00	5549.16	12.67	4898.65	8.00	4475.40
SDs		0.00	283.40	0.00	145.41	0.00	160.27	0.00	173.58	0.58	670.44	0.00	38.73
% to DACS 02	4800	-2.70	7.10	0.00	-16.07	-1.82	-3.48	0.00	-6.75	-2.56	-2.81	0.00	-11.80
05 - AVGs	4800	37.00	10855.84	40.00	7222.70	37.00	9341.48	8.00	8255.87	12.00	4161.42	11.67	6974.40
SDs		0.00	278.40	0.00	41.42	0.00	122.18	0.00	379.87	0.00	151.52	0.58	279.97
% to DACS 02	4800	-1.77	-10.15	-0.83	-12.54	-2.63	-8.41	0.00	-5.86	-5.26	-10.33	-2.78	-5.55
06 - AVGs	4800	36.67	10611.06	40.00	7259.73	37.00	9229.06	8.00	7189.22	12.00	4216.09	10.00	6979.12
SDs		0.58	1464.56	0.00	22.41	0.00	102.52	0.00	187.80	0.00	233.74	0.00	77.60
% to DACS 02	4800	-1.79	-4.59	-4.76	-14.86	-2.63	-9.23	0.00	-9.09	-7.69	-20.38	-6.25	-3.26
07 - AVGs	4800	36.33	10280.79	40.00	7291.70	37.00	9334.31	8.00	6342.67	12.67	4139.22	9.67	6226.10
SDs		0.58	1087.03	0.00	125.85	0.00	101.88	0.00	213.90	0.58	58.20	0.58	54.91
% to DACS 02	4800	-1.80	-0.41	-1.64	-13.44	-2.63	-10.04	0.00	-7.48	-5.00	-19.00	-3.33	-7.62
08 - AVGs	4800	36.00	9174.45	39.67	7504.66	37.00	9281.23	8.00	5121.86	12.00	4052.04	9.00	5854.65
SDs		0.00	124.73	0.58	406.43	0.00	109.40	0.00	157.96	0.00	90.40	0.00	229.42
% to DACS 02	4800	0.00	-16.19	-4.03	-13.78	0.00	-8.46	0.00	-9.53	0.00	-16.93	3.85	-2.33
09 - AVGs	4800	37.00	9981.59	38.00	7451.09	37.00	9151.47	8.00	7231.14	12.67	4389.81	8.00	5767.31
SDs		0.00	7.70	0.00	194.31	0.00	222.56	0.00	194.92	0.58	133.90	0.00	115.41
% to DACS 02	4800	-0.89	-11.82	-2.56	-26.41	0.00	-10.01	0.00	-5.42	-5.00	-18.30	0.00	-0.94
10 - AVGs	4800	36.00	11183.63	37.33	7708.78	36.33	10081.68	8.00	6875.64	12.00	4148.86	8.00	5458.92
SDs		0.00	121.82	0.58	315.09	0.58	1071.68	0.00	38.66	0.00	40.07	0.00	30.02
% to DACS 02	4800	-2.70	7.24	-1.75	-28.58	-1.80	2.95	0.00	-6.85	0.00	-12.71	0.00	-1.36
DACS 03 - AVGs	4800	36.90	10273.65	38.67	7767.26	36.80	9357.76	8.10	7221.52	12.27	4268.89	9.80	6194.68
SDs		0.10	223.49	0.12	62.03	0.10	158.47	0.00	32.14	0.15	60.64	0.10	8.85
% to DACS 02	4800	-1.86	-7.50	-2.68	-20.84	-2.04	-7.12	-1.62	-7.45	-3.41	-15.15	-1.34	-6.47
% to RVNSc [5]	3900-7980	1.93	12.23	1.75	6.09	1.94	8.45	1.25	10.29	2.22	8.82	12.64	9.97
% to ES4C [2]	4800	1.65	15.11	1.75	2.42	1.94	6.79	1.25	11.07	2.22	8.49	13.95	12.26
DACS 02 - AVGs	4800	37.60	11106.46	39.73	9811.51	37.57	10074.83	8.23	7802.75	12.70	5031.00	9.93	6623.27
RVNSc [5] - AVGs	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2] - AVGs	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33] - AVGs	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48] - AVGs	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17] - AVGs	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17] - AVGs	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table 4.24: Comparison between the average case performances of DACS+HLS and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200. Check Tables E.3 and E.4 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	2400	20.00	5443.00	20.00	2726.35	18.67	3958.07	5.00	4328.39	6.00	1961.40	6.00	3369.15
SDs		0.00	19.90	0.00	8.57	0.58	314.75	0.00	24.74	0.00	33.05	0.00	4.17
% to DACS 03	2400	0.00	0.47	0.00	0.81	-1.75	5.24	0.00	-0.66	0.00	0.11	0.00	1.46
02 - AVGs	2400	18.00	4750.90	18.00	3172.77	18.00	4013.33	4.00	3998.90	6.00	1908.98	5.00	3055.89
SDs		0.00	6.96	0.00	48.18	0.00	164.39	0.00	83.87	0.00	79.36	0.00	111.99
% to DACS 03	2400	-1.82	5.36	-3.57	8.14	0.00	6.74	-14.29	11.95	0.00	1.23	0.00	1.67
03 - AVGs	2400	18.00	3792.66	18.00	3022.41	18.00	3747.10	4.00	3167.88	6.00	1999.55	4.00	2757.83
SDs		0.00	169.14	0.00	163.86	0.00	81.57	0.00	49.00	0.00	39.60	0.00	21.68
% to DACS 03	2400	0.00	-0.07	0.00	-0.84	0.00	2.69	0.00	3.61	0.00	7.67	0.00	0.25
04 - AVGs	2400	18.00	3434.05	18.00	2896.98	18.00	3451.01	4.00	2192.60	6.00	1853.09	4.00	2301.17
SDs		0.00	68.83	0.00	40.77	0.00	97.51	0.00	50.73	0.00	20.29	0.00	36.49
% to DACS 03	2400	0.00	1.33	0.00	0.15	0.00	2.99	0.00	1.90	-5.26	-1.51	0.00	-0.17
05 - AVGs	2400	18.00	5102.96	20.00	2709.98	18.00	4367.38	4.00	3631.41	6.00	1915.57	5.00	2985.59
SDs		0.00	132.08	0.00	6.86	0.00	256.93	0.00	63.68	0.00	5.60	0.00	46.41
% to DACS 03	2400	-1.82	7.15	0.00	0.15	-5.26	22.23	0.00	-3.96	0.00	0.61	0.00	0.49
06 - AVGs	2400	18.00	4141.31	20.00	2805.74	18.00	4226.19	4.00	3111.50	6.00	1997.91	5.00	2886.28
SDs		0.00	157.44	0.00	122.94	0.00	150.40	0.00	86.95	0.00	89.29	0.00	70.77
% to DACS 03	2400	0.00	0.53	0.00	1.72	-1.82	4.65	0.00	0.69	0.00	5.40	0.00	0.37
07 - AVGs	2400	18.00	3763.11	20.00	2736.98	18.00	4242.23	4.00	2690.28	6.00	1950.17	4.00	2883.55
SDs		0.00	272.02	0.00	9.93	0.00	96.50	0.00	41.36	0.00	78.10	0.00	118.76
% to DACS 03	2400	0.00	6.46	0.00	1.18	0.00	3.64	0.00	-1.51	0.00	-0.99	-14.29	3.16
08 - AVGs	2400	18.00	3336.48	19.67	2798.92	18.00	3722.18	4.00	2060.11	6.00	1870.58	4.00	2600.09
SDs		0.00	24.96	0.58	174.31	0.00	148.54	0.00	32.58	0.00	15.70	0.00	39.26
% to DACS 03	2400	0.00	1.43	-1.67	3.75	0.00	-2.59	0.00	-1.74	0.00	-0.16	0.00	0.43
09 - AVGs	2400	18.00	4628.36	18.33	2974.03	18.00	3662.72	4.00	3288.37	6.00	2172.44	4.00	2434.96
SDs		0.00	116.25	0.58	166.26	0.00	77.87	0.00	17.08	0.00	104.73	0.00	48.06
% to DACS 03	2400	0.00	2.26	-3.51	8.72	0.00	0.72	0.00	1.94	0.00	13.61	0.00	-0.23
10 - AVGs	2400	18.00	3817.12	18.00	3183.58	18.00	3570.53	4.00	2897.55	6.00	1869.41	4.00	2264.41
SDs		0.00	95.15	0.00	130.51	0.00	224.26	0.00	61.70	0.00	68.95	0.00	125.50
% to DACS 03	2400	0.00	2.58	-3.57	15.56	0.00	1.41	0.00	2.11	0.00	0.54	0.00	-1.92
DACS+HLS - AVGs	2400	18.20	4221.00	19.00	2902.78	18.07	3896.07	4.10	3136.70	6.00	1949.91	4.50	2753.89
SDs		0.00	12.25	0.00	16.68	0.06	29.53	0.00	22.68	0.00	4.05	0.00	36.73
% to DACS 03	2400	-0.36	2.79	-1.21	3.91	-0.91	4.72	-1.60	1.51	-0.55	2.64	-1.46	0.64
% to RVNSc [5]	720-1680	0.55	10.46	0.53	4.46	0.37	11.06	2.50	3.00	0.00	5.83	2.27	4.78
% to ES4C [2]	2400	0.00	13.93	0.53	4.34	0.37	9.59	2.50	2.67	0.00	5.63	4.65	2.95
DACS 03 - AVGs	2400	18.27	4106.41	19.23	2793.65	18.23	3720.41	4.17	3090.08	6.03	1899.77	4.57	2736.25
RVNSc [5] - AVGs	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2] - AVGs	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33] - AVGs	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48] - AVGs	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17] - AVGs	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17] - AVGs	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table 4.25: Comparison between the average case performances of DACS+HLS and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400. Check Tables E.5 and E.6 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	4800	40.00	11632.66	40.00	7158.83	37.00	9646.34	8.00	10706.65	12.00	4272.48	12.00	7927.89
SDs		0.00	45.53	0.00	5.74	0.00	104.07	0.00	415.79	0.00	150.86	0.00	97.27
% to DACS 03	4800	0.00	0.94	0.00	-0.11	0.00	-0.15	-11.11	7.21	0.00	0.51	-7.69	5.26
02 - AVGs	4800	36.00	12801.92	37.33	9196.26	36.33	10113.72	8.00	8718.06	12.00	4057.00	11.00	7341.58
SDs		0.00	479.72	0.58	106.02	0.58	1030.15	0.00	181.76	0.00	117.94	0.00	202.18
% to DACS 03	4800	-2.70	24.83	0.00	7.32	-1.80	15.29	0.00	0.90	0.00	-1.96	-2.94	5.89
03 - AVGs	4800	36.00	10710.43	37.00	8933.30	36.00	10464.50	8.00	7143.22	12.00	5072.78	9.00	6353.79
SDs		0.00	115.03	0.00	168.93	0.00	124.34	0.00	85.13	0.00	371.73	0.00	87.89
% to DACS 03	4800	-2.70	15.79	-0.89	-3.20	-1.82	13.36	0.00	1.71	-5.26	18.14	-3.57	10.59
04 - AVGs	4800	36.00	9478.03	36.00	9471.32	36.00	9294.51	8.00	5466.71	12.00	5107.70	8.00	4638.05
SDs		0.00	80.36	0.00	218.96	0.00	243.63	0.00	101.21	0.00	146.89	0.00	148.51
% to DACS 03	4800	0.00	-1.46	-2.70	14.53	0.00	-2.10	0.00	-1.49	-5.26	4.27	0.00	3.63
05 - AVGs	4800	36.67	11394.87	40.00	7196.63	37.00	9343.56	8.00	8242.71	12.00	4442.72	11.00	6996.60
SDs		0.58	696.88	0.00	22.67	0.00	236.65	0.00	143.60	0.00	283.01	0.00	266.36
% to DACS 03	4800	-0.90	4.97	0.00	-0.36	0.00	0.02	0.00	-0.16	0.00	6.76	-5.71	0.32
06 - AVGs	4800	36.00	11660.75	40.00	7199.81	36.67	10393.86	8.00	7224.02	12.00	4259.14	9.00	7122.16
SDs		0.00	317.31	0.00	79.74	0.58	1506.50	0.00	126.97	0.00	308.23	0.00	176.87
% to DACS 03	4800	-1.82	9.89	0.00	-0.83	-0.90	12.62	0.00	0.48	0.00	1.02	-10.00	2.05
07 - AVGs	4800	36.00	10515.45	40.00	7365.72	37.00	9500.71	8.00	6410.22	12.00	4373.84	9.00	6728.55
SDs		0.00	247.00	0.00	158.88	0.00	107.86	0.00	119.74	0.00	271.65	0.00	182.54
% to DACS 03	4800	-0.92	2.28	0.00	1.02	0.00	1.78	0.00	1.06	-5.26	5.67	-6.90	8.07
08 - AVGs	4800	36.00	9171.03	39.33	7487.01	36.00	10794.56	8.00	5296.72	12.00	4394.97	8.00	5913.24
SDs		0.00	167.89	0.58	262.38	0.00	217.08	0.00	110.52	0.00	5.60	0.00	70.91
% to DACS 03	4800	0.00	-0.04	-0.84	-0.24	-2.70	16.31	0.00	3.41	0.00	8.46	-11.11	1.00
09 - AVGs	4800	36.00	12456.31	37.00	7884.02	36.00	10374.86	8.00	7478.47	12.00	4662.31	8.00	5712.86
SDs		0.00	555.08	0.00	170.12	0.00	198.42	0.00	228.52	0.00	286.99	0.00	47.98
% to DACS 03	4800	-2.70	24.79	-2.63	5.81	-2.70	13.37	0.00	3.42	-5.26	6.21	0.00	-0.94
10 - AVGs	4800	36.00	10894.58	37.00	7714.75	36.00	10223.05	8.00	6847.29	12.00	4255.49	8.00	5476.21
SDs		0.00	274.97	0.00	232.33	0.00	469.94	0.00	119.36	0.00	142.25	0.00	100.52
% to DACS 03	4800	0.00	-2.58	-0.89	0.08	-0.92	1.40	0.00	-0.41	0.00	2.57	0.00	0.32
DACS+HLS- AVGs	4800	36.47	11071.60	38.37	7960.76	36.40	10014.97	8.00	7353.41	12.00	4489.84	9.30	6421.09
SDs		0.06	131.21	0.12	36.21	0.10	174.05	0.00	38.32	0.00	70.20	0.00	70.41
% to DACS 03	4800	-1.17	7.77	-0.78	2.49	-1.09	7.02	-1.23	1.83	-2.17	5.18	-5.10	3.65
% to RVNSc [5]	3900-7980	0.74	20.94	0.96	8.73	0.83	16.07	0.00	12.30	0.00	14.46	6.90	13.98
% to ES4C [2]	4800	0.46	24.05	0.96	4.97	0.83	14.29	0.00	13.09	0.00	14.10	8.14	16.37
DACS 03 - AVGs	4800	36.90	10273.65	38.67	7767.26	36.80	9357.76	8.10	7221.52	12.27	4268.89	9.80	6194.68
RVNSc [5] - AVGs	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2] - AVGs	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33] - AVGs	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48] - AVGs	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17] - AVGs	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17] - AVGs	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table 4.26: Comparison between the average case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200. Check Tables F.3 and F.4 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	2400	20.00	5063.16	20.00	2704.57	18.67	3916.31	5.00	4083.30	6.00	1931.44	6.00	3259.76
SDs		0.00	80.11	0.00	0.00	0.58	375.05	0.00	14.36	0.00	0.00	0.00	15.06
% to DACS+HLS	2400	0.00	-6.98	0.00	-0.80	0.00	-1.06	0.00	-5.66	0.00	-1.53	0.00	-3.25
02 - AVGs	2400	18.00	4608.76	18.00	3058.86	18.00	3750.72	4.00	3785.00	6.00	1863.16	5.00	2949.88
SDs		0.00	82.34	0.00	82.61	0.00	76.44	0.00	51.11	0.00	0.00	0.00	65.87
% to DACS+HLS	2400	0.00	-2.99	0.00	-3.59	0.00	-6.54	0.00	-5.35	0.00	-2.40	0.00	-3.47
03 - AVGs	2400	18.00	3632.38	18.00	2900.79	18.00	3449.69	4.00	3124.12	6.00	1856.14	4.00	2790.75
SDs		0.00	27.80	0.00	90.21	0.00	50.03	0.00	51.95	0.00	83.26	0.00	86.57
% to DACS+HLS	2400	0.00	-4.23	0.00	-4.02	0.00	-7.94	0.00	-1.38	0.00	-7.17	0.00	1.19
04 - AVGs	2400	18.00	3274.51	18.00	2817.17	18.00	3202.33	4.00	2104.64	6.00	1875.14	4.00	2271.65
SDs		0.00	13.87	0.00	14.70	0.00	90.69	0.00	12.05	0.00	94.62	0.00	58.72
% to DACS+HLS	2400	0.00	-4.65	0.00	-2.76	0.00	-7.21	0.00	-4.01	0.00	1.19	0.00	-1.28
05 - AVGs	2400	18.00	4916.64	20.00	2702.05	18.00	4172.54	4.00	3459.13	6.00	1883.28	5.00	2820.18
SDs		0.00	63.06	0.00	0.00	0.00	29.91	0.00	48.40	0.00	6.87	0.00	21.42
% to DACS+HLS	2400	0.00	-3.65	0.00	-0.29	0.00	-4.46	0.00	-4.74	0.00	-1.69	0.00	-5.54
06 - AVGs	2400	18.00	3872.25	20.00	2701.04	18.00	4053.36	4.00	2995.49	6.00	1863.13	5.00	2757.93
SDs		0.00	119.98	0.00	0.00	0.00	143.09	0.00	7.37	0.00	6.08	0.00	55.41
% to DACS+HLS	2400	0.00	-6.50	0.00	-3.73	0.00	-4.09	0.00	-3.73	0.00	-6.75	0.00	-4.45
07 - AVGs	2400	18.00	3474.80	20.00	2701.04	18.00	3978.33	4.00	2642.47	6.00	1873.36	4.00	2809.58
SDs		0.00	61.79	0.00	0.00	0.00	132.31	0.00	103.03	0.00	39.72	0.00	116.75
% to DACS+HLS	2400	0.00	-7.66	0.00	-1.31	0.00	-6.22	0.00	-1.78	0.00	-3.94	0.00	-2.57
08 - AVGs	2400	18.00	3215.48	19.67	2740.56	18.00	3688.00	4.00	1981.61	6.00	1835.30	4.00	2442.39
SDs		0.00	45.28	0.58	79.07	0.00	36.06	0.00	29.89	0.00	1.76	0.00	89.62
% to DACS+HLS	2400	0.00	-3.63	0.00	-2.09	0.00	-0.92	0.00	-3.81	0.00	-1.89	0.00	-6.07
09 - AVGs	2400	18.00	4327.52	18.00	3121.11	18.00	3714.83	4.00	3236.10	6.00	1853.10	4.00	2337.13
SDs		0.00	50.34	0.00	146.35	0.00	115.32	0.00	45.24	0.00	24.08	0.00	8.49
% to DACS+HLS	2400	0.00	-6.50	-1.82	4.95	0.00	1.42	0.00	-1.59	0.00	-14.70	0.00	-4.02
10 - AVGs	2400	18.00	3687.40	18.00	2904.38	18.00	3471.91	4.00	2795.28	6.00	1820.11	4.00	2188.02
SDs		0.00	160.49	0.00	94.70	0.00	61.60	0.00	72.52	0.00	5.43	0.00	42.95
% to DACS+HLS	2400	0.00	-3.40	0.00	-8.77	0.00	-2.76	0.00	-3.53	0.00	-2.64	0.00	-3.37
DACS+HLS+2-Opt - AVGs	2400	18.20	4007.29	18.97	2835.16	18.07	3739.80	4.10	3020.71	6.00	1865.42	4.50	2662.73
SDs		0.00	45.32	0.06	17.84	0.06	50.75	0.00	14.04	0.00	18.68	0.00	16.35
% to DACS+HLS	2400	0.00	-5.06	-0.18	-2.33	0.00	-4.01	0.00	-3.70	0.00	-4.33	0.00	-3.31
% to RVNSc [5]	720-1680	0.55	4.86	0.35	2.03	0.37	6.61	2.50	-0.81	0.00	1.25	2.27	1.31
% to ES4C [2]	2400	0.00	8.16	0.35	1.91	0.37	5.20	2.50	-1.12	0.00	1.05	4.65	-0.46
DACS+HLS - AVGs	2400	18.20	4221.00	19.00	2902.78	18.07	3896.07	4.10	3136.70	6.00	1949.91	4.50	2753.89
RVNSc [5] - AVGs	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2] - AVGs	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LCO3 [33] - AVGs	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48] - AVGs	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17] - AVGs	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17] - AVGs	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table 4.27: Comparison between the average case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400. Check Tables F.5 and F.6 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	4800	40.00	10925.59	40.00	7152.06	37.00	9189.29	8.00	9768.99	12.00	4119.20	12.00	7379.59
SDs		0.00	91.35	0.00	0.00	0.00	91.97	0.00	44.51	0.00	0.11	0.00	135.94
% to DACS+HLS	4800	0.00	-6.08	0.00	-0.09	0.00	-4.74	0.00	-8.76	0.00	-3.59	0.00	-6.92
02 - AVGs	4800	36.00	11058.25	37.33	8009.51	36.00	10204.17	8.00	8134.41	12.00	3959.56	11.00	6379.40
SDs		0.00	407.69	0.58	309.15	0.00	686.98	0.00	153.90	0.00	24.28	0.00	144.19
% to DACS+HLS	4800	0.00	-13.62	0.00	-12.90	-0.92	0.89	0.00	-6.69	0.00	-2.40	0.00	-13.11
03 - AVGs	4800	36.00	9892.07	36.33	9001.30	36.00	9778.42	8.00	6708.33	12.00	4240.50	8.67	5843.04
SDs		0.00	204.40	0.58	1087.84	0.00	80.24	0.00	73.01	0.00	106.82	0.58	205.39
% to DACS+HLS	4800	0.00	-7.64	-1.80	0.76	0.00	-6.56	0.00	-6.09	0.00	-16.41	-3.70	-8.04
04 - AVGs	4800	36.00	8944.64	36.00	9412.26	36.00	8940.03	8.00	5110.14	12.00	4671.09	8.00	4176.88
SDs		0.00	7.78	0.00	189.59	0.00	44.41	0.00	107.30	0.00	282.26	0.00	83.72
% to DACS+HLS	4800	0.00	-5.63	0.00	-0.62	0.00	-3.81	0.00	-6.52	0.00	-8.55	0.00	-9.94
05 - AVGs	4800	36.33	11442.41	40.00	7152.06	37.00	8974.11	8.00	7613.79	12.00	3997.00	11.00	6109.77
SDs		0.58	1244.94	0.00	0.00	0.00	45.58	0.00	66.73	0.00	68.59	0.00	163.59
% to DACS+HLS	4800	-0.91	0.42	0.00	-0.62	0.00	-3.95	0.00	-7.63	0.00	-10.03	0.00	-12.68
06 - AVGs	4800	36.00	10781.10	40.00	7153.45	36.33	9955.15	8.00	6868.97	12.00	3905.84	9.00	6441.84
SDs		0.00	229.24	0.00	0.00	0.58	1028.11	0.00	82.45	0.00	11.71	0.00	191.05
% to DACS+HLS	4800	0.00	-7.54	0.00	-0.64	-0.91	-4.22	0.00	-4.91	0.00	-8.30	0.00	-9.55
07 - AVGs	4800	36.00	9971.73	40.00	7149.43	36.00	10974.10	8.00	6021.50	12.00	4058.32	8.33	6031.39
SDs		0.00	401.87	0.00	0.00	0.00	422.05	0.00	174.62	0.00	149.55	0.58	78.13
% to DACS+HLS	4800	0.00	-5.17	0.00	-2.94	-2.70	15.51	0.00	-6.06	0.00	-7.21	-7.41	-10.36
08 - AVGs	4800	36.00	8917.81	39.00	7324.00	36.00	10337.47	8.00	4894.89	12.00	3888.62	8.00	5478.92
SDs		0.00	67.03	0.00	118.09	0.00	331.26	0.00	50.36	0.00	9.67	0.00	60.55
% to DACS+HLS	4800	0.00	-2.76	-0.85	-2.18	0.00	-4.23	0.00	-7.59	0.00	-11.52	0.00	-7.34
09 - AVGs	4800	36.00	11444.72	37.00	7702.80	36.00	10115.73	8.00	6943.37	12.00	4500.56	8.00	5255.24
SDs		0.00	485.79	0.00	177.55	0.00	269.91	0.00	107.34	0.00	176.92	0.00	53.96
% to DACS+HLS	4800	0.00	-8.12	0.00	-2.30	0.00	-2.50	0.00	-7.16	0.00	-3.47	0.00	-8.01
10 - AVGs	4800	36.00	10221.56	37.00	7447.29	36.00	9925.03	8.00	6572.67	12.00	3822.77	8.00	5004.07
SDs		0.00	151.33	0.00	190.97	0.00	220.93	0.00	66.85	0.00	42.40	0.00	23.54
% to DACS+HLS	4800	0.00	-6.18	0.00	-3.47	0.00	-2.92	0.00	-4.01	0.00	-10.17	0.00	-8.62
DACS+HLS+2-Opt - AVGs	4800	36.43	10359.99	38.27	7750.42	36.23	9839.35	8.00	6863.71	12.00	4116.35	9.20	5810.01
SDs		0.06	230.83	0.06	75.54	0.06	52.90	0.00	35.08	0.00	20.27	0.00	27.72
% to DACS+HLS	4800	-0.09	-6.43	-0.26	-2.64	-0.46	-1.75	0.00	-6.66	0.00	-8.32	-1.08	-9.52
% to RVNSc [5]	3900-7980	0.64	13.17	0.70	5.86	0.37	14.03	0.00	4.82	0.00	4.94	5.75	3.14
% to ES4C [2]	4800	0.37	16.08	0.70	2.19	0.37	12.28	0.00	5.56	0.00	4.61	6.98	5.29
DACS+HLS - AVGs	4800	36.47	11071.60	38.37	7960.76	36.40	10014.97	8.00	7353.41	12.00	4489.84	9.30	6421.09
RVNSc [5] - AVGs	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2] - AVGs	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33] - AVGs	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48] - AVGs	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17] - AVGs	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17] - AVGs	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

4.7.6 What if saving ants with two different saving functions are used?

In order to know more about the effects of choosing edges or pairs of two nodes rather singular nodes (as in the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt used in Sections 4.7.1, 4.7.3 and 4.7.4) on the performance and the results, the routing builder of ants is changed into a constructive phase that depends on a Savings algorithm that has some similarities with the Savings algorithm of Clarke and Wright [10]. The ants, which use the Savings algorithm in this thesis, are called as the saving ants.

When building the solutions in a way based on an edge-by-edge basis, the saving ants use the saving entries of the saving matrix, created before the start of the computation process of the DACS system involved, in addition to the pheromone trails of the pheromone memory structure in a colony, VMIN or DMIN. The saving entries are created using either a distance-based function as in Equation 4.7 or a time-based function as in Equation 4.8 and then they are normalized as in Equation 4.9 by dividing the value of each saving entry over the maximum saving value. The main purpose of such functions is to calculate and to know the saving value of each pair of two nodes, when served together on the same route rather than serving the two nodes separately on two different routes. Also, the reason behind the calculation of the values of the saving entries is to know the pairs of two nodes that have the largest saving values in order to be considered first during the building up of the solution of a saving ant and before the other pairs of two nodes, which have smaller saving values.

$$sv_{c_i cj} = d_{c_0 c_i} + d_{c_0 c_j} - d_{c_i c_j} \tag{4.7}$$

$$\begin{aligned} ttc_{c_0 ci} &= vw_{n_i} + s_{c_i} + (2 \times d_{c_0 c_i}) \\ ttc_{c_0 cj} &= vw_{n_j} + s_{c_j} + (2 \times d_{c_0 c_j}) \\ tt_{c_i} &= d_{c_0 c_i} + vw_{n_i} + s_{c_i} \\ ttc_{c_i cj} &= tt_{c_i} + d_{c_i c_j} + vw_{c_i c_j} + s_{c_j} + d_{c_0 c_j} \\ sv_{c_i cj} &= ttc_{c_0 c_i} + ttc_{c_0 c_j} - ttc_{c_i c_j} \end{aligned} \tag{4.8}$$

$$\eta_{c_i c_j} = \frac{sv_{c_i cj}}{\max_{i=1}^{N}[sv_{c_i c_j}]}, \qquad 1 \leq j \leq N \tag{4.9}$$

Before starting to build the solution of a saving ant, a sub list of the saving matrix is created and its length is equal to 20. The sub list is populated with twenty edges according to Equation 4.10, which has the two terms of the pheromone trail and the saving value of any edge. Once an edge is chosen to be one of the twenty edges of the sub list, it is tabooed in order not to be considered at future updates of the sub list. The reason behind using the sub list is to let that sub list be used by the probabilistic-state transition rule with its exploitation and exploration parts in Equation 4.3 and to avoid any worries about consuming a lot of CPU time when using the saving matrix, which is very large.

$$e_{c_i c_j} = \max(\tau_{c_i c_j} . [\eta_{c_i c_j}]^{\beta}) \qquad 1 \leq i, j \leq N \tag{4.10}$$

During the building up of the solution of a saving ant, the probabilistic state transition rule with its exploitation and exploration parts is used in the same way as in any other DACS system but the main difference is in the using of the visibility function in Equation 4.9. Therefore if the exploitation part is used, the saving ant picks the edge $e_{c_i c_j}$ that has the largest value, which is combined of two values - the pheromone trail and the saving value of the edge $e_{c_i c_j}$. However if the exploration part is used, the saving ant chooses in a great chance an edge $e_{c_i c_j}$ with a large value of the two values described earlier but not necessarily the edge with the maximum value. If an edge $e_{c_i c_j}$ is chosen using either the exploitation or exploration part, then it is inserted after considering one of the following three steps explained below.

RL1- If the two customers, combined by the edge $e_{c_i c_j}$, do not exist in a route, then a new route is created and they are inserted in the same order in which they exist in the edge involved.

RL2- If one of the two customers of the edge $e_{c_i c_j}$ does not exist in a route while the other customer is located in a route partially built, then the customer that does not exist in a route is inserted near the other customer located in a route under one condition, in which the other customer has to be an exterior and not an interior node. A customer can be considered as an exterior node only if the customer is located at the start or at the end of the route. There are four cases in which a customer that does not exist is inserted and those four cases are mentioned below.

(a) If the customer c_i does exist at the end of a route R_1 and the customer c_j is not located in any route, the order of the nodes visited in R_1 is kept as it is and c_j is attached at the end of R_1.

(b) If the customer c_i does exist at the start of a route R_1 and the customer c_j is not located in any route, the order of the nodes visited in R_1 is reversed as in Figure 4.13 and c_j is attached at the end of R_1.

(c) If the customer c_j does exist at the end of a route R_1 and the customer c_i is not located in any route, c_i is attached at the start of the route R_1 and then the order of the nodes visited after c_i is reversed in R_1 as in Figure 4.13.

(d) If the customer c_j does exist at the start of a route R_1 and the customer c_i is not located in any route, c_i is attached at the start of R_1 before c_j and the order of the nodes visited after c_i is kept as it is in R_1.

RL3- If the two customers of an edge $e_{c_ic_j}$ are located in two different routes and both customers are exterior either at the end or at the start of the routes involved, then the two routes that include such two customers are merged together. In order to merge two routes, there are four cases in which any two routes can be merged together and these four cases are mentioned below.

(a) If the customer c_i is located at the start of the route R_1 and the customer c_j is located at the start of the route R_2, the order of the nodes visited in R_1 is reversed as in Figure 4.13 and the new end c_i of R_1 is attached with the start node c_j of R_2 to form a new route R_3.

(b) If the customer c_i is located at the start of the route R_1 and the customer c_j is located at the end of the route R_2, the order of the nodes visited in R_1 and R_2 is reversed as in Figure 4.13 and then the new end c_i of R_1 is attached with the new start c_j of R_2 to form a new route R_3.

(c) If the customer c_i is located at the end of the route R_1 and the customer c_j is located at the start of the route R_2, the end c_i of R_1 is attached with the start c_j of R_2 to form a new route R_3.

(d) If the customer c_i is located at the end of the route R_1 and the customer node c_j is located at the end of the route R_2, the order of the nodes

visited in R_2 is reversed as in Figure 4.13 and then the end c_i of R_1 is attached with the new start c_j of R_2 to form a new route R_3.

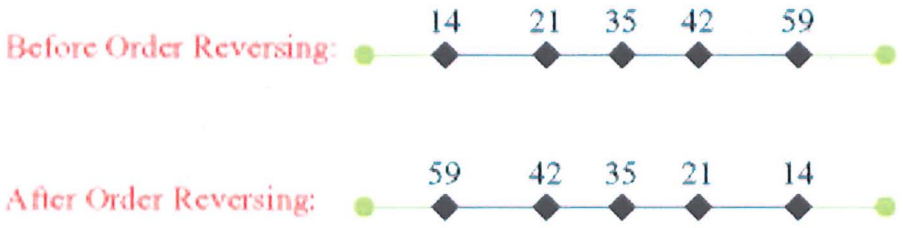


Figure 4.13: Reversing the order of a group of visited customer nodes in a route.

Furthermore because the saving matrix is very big and it is time consuming to try every single entry of it, each saving ant has the ability to update the sub list, used by the probabilistic state transition rule with its exploitation and exploration parts, by using the saving matrix for up to 1000 iterations. Now at any case, if an edge $e_{c_i c_j}$ located in that sub list is chosen at an iteration to be a part of the solution of a saving ant, then the selected edge will be tabooed in order not to be chosen later and the sub list will be updated to include a new edge instead of the one chosen to be possibly used in the next iteration.

After changing the routing builder of the ants into the new constructive phase described above, the three systems DACS 03, DACS+HLS and DACS+HLS+2-Opt with saving ants that use one of the two visibility functions in Equations 4.7 and 4.8 are tested each for three runs on the problem set R1 of the problem group PG100 in Section 2.2. In a matter of fact, the performance and the results of the DACS systems that use the saving ants are very poor and not in any way as competitive as the systems that do not use them. However, one thing recognized is that the addition of the hybrid local search HLS and the 2-Opt move variant to the system DACS 03 that uses the saving ants can improve things better but even though the performance and results are still poor and this suggests that changing the constructive phase in the way described above is not a significant idea. Nonetheless, the performance and the results of such DACS systems with the saving ants can be improved, if the behavior of the saving ants is studied and investigated further.

4.7.7 Which component is causing the synergetic effects of artificial ants?

According to the literature in [49] [50] [51] [52] [4] [54] [55] [56] [6] and [57], the usage and the update of the pheromone trails is found to be vital in encouraging the ants of a cycle to become synergetic and letting them look always for new edges and quality solutions that have not been discovered before by previous waves of ants. But also in Sections 4.6.1, 4.7.1, 4.7.3 and 4.7.4, the importance of using the local search of triple moves (M1-M3), the push-forward and push-backward strategy PFPBS, the hybrid local search HLS and the 2-Opt move is discovered and results show that a DACS system without them will lead to very bad performance.

Consequently, what is described above triggers the question of which one of these components of the DACS system in general is causing such synergetic effects. Are they the pheromone trails and how to use and update them or just simply the local searches with the help of the PFPBS strategy and how they are efficiently designed? The moment the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt are tested without any pheromone usage and update on the problem set R1 of the problem group PG100 as in Table 4.28, the performances and results of such systems have deteriorated and this deterioration is significant at the 99.9% level according to Student's t-tests and Wilcoxon signed rank tests.

However in the systems that use the HLS and the 2-Opt, one has to recognize is that the percentages of deterioration on average (with 4.46% to 6.39% for TD and 0.79% to 1.07% for NV) after switching off the usage and the update of the pheromone trails are far lesser than those of the system DACS 03 - 14.17% for TD and 1.18% for NV. In a matter of fact, using the HLS and later the 2-Opt helps a lot in recovering on average the performance and the results in terms of TD with 50.97% to almost 75.88% of what is deteriorated or lost in DACS 03 because of putting off any usage and update of the pheromone trails.

Moreover according to Student's t-tests and Wilcoxon signed rank tests, the systems that use HLS and 2-Opt and do NOT have any pheromone usage and update are still able on average to bring at the 99.9% level NV results that are better significantly by -0.75% to -0.94% in contrast with those of the system DACS 03 that use and update the pheromone trails. Here also, the DACS 03 system that

use and update the pheromone trails is still using the local search of triple moves (M1-M3) and therefore all the goodness in the TD result 1243.26 of Table 4.28 is not because of the pheromone usage and update.

Of course, the talk described above gives the indication that if the local search is fiddled with more, then that can lead into improving the performance of the DACS system even further until reaching the stage where the usage and the update of the pheromone trails are not needed any more. As a conclusion, the local search with the help of the PFPBS strategy has a great influence on the synergetic effects of the ants and switching it off will lead into performance and results that are far worse than that of a system that does not use and update the pheromone trails.

4.7.8 What if the pheromone ants replaced with more deterministic ones?

After reviewing the deterministic approaches like the reactive variable neighbourhood search in [5] and [8], it can be realized that such deterministic approaches can find very good quality solutions for the VRPTW problem without using and updating the pheromone trails as in [4]. For that, it is believed that the pheromone component is nothing but a perturbation force in ACO - Ant Colony Optimisation [49] [50] [51] [52] [4] [54] [55] [56] [6] and [57]. Therefore, the theory, here, is that very good quality solutions can be found for VRPTW by using some sort of force that chooses customers or edges rather than depending on the pheromone component in the choosing process.

Consequently in this section, the pheromone ants are converted to more deterministic ones. In order to make that conversion happen, a number of components, mentioned below, are introduced to differentiate between the more deterministic ants and the pheromone ones. For that, the perturbation force of the pheromone component in the pheromone ants is substituted by the following components from CP1 to CP4:

CP1- A deterministic ant seeds its solution with a customer in the graph, using the farthest in distance or the earliest deadline seeding strategy in Section 5.4, before using the components enumerated in CP2 and CP3. The seeding strategies are designed to be used in this order - firstly the farthest in distance and

Table 4.28: Tests that show the synergetic effects of ants in the three systems DACS 03, DACS+HLS and DACS+HLS+2-Opt on the problem set R1 of the problem group PG100.

		A) DACS 03		B) DACS 03		C) DACS+ HLS		D) DACS+ HLS		E) DACS+ HLS+ 2-Opt		F) DACS+ HLS+ 2-Opt	
Using and updating pheromone		Yes		No		Yes		No		Yes		No	
Batch No.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	100	12.92	1242.24	13.06	1422.10	12.64	1244.87	12.81	1326.21	12.72	1230.32	12.86	1284.86
SDs		0.22	5.89	0.05	15.05	0.10	10.52	0.05	17.94	0.13	3.89	0.13	22.38
CVs		1.71	0.47	0.37	1.06	0.76	0.84	0.38	1.35	1.00	0.32	0.99	1.74
02 - AVGs	100	12.94	1238.30	13.11	1401.04	12.69	1246.05	12.81	1330.42	12.75	1235.05	12.81	1287.75
SDs		0.10	13.97	0.10	11.79	0.05	8.82	0.13	12.59	0.14	4.77	0.13	8.87
CVs		0.74	1.13	0.73	0.84	0.38	0.71	0.99	0.95	1.13	0.39	0.99	0.69
03 - AVGs	100	13.03	1238.88	13.06	1411.24	12.72	1253.55	12.83	1332.36	12.67	1231.88	12.81	1282.12
SDs		0.05	4.00	0.05	12.76	0.13	1.71	0.00	17.35	0.08	4.78	0.05	3.80
CVs		0.37	0.32	0.37	0.90	1.00	0.14	0.00	1.30	0.66	0.39	0.38	0.30
04 - AVGs	100	12.89	1236.63	13.00	1447.15	12.69	1255.30	12.89	1330.09	12.64	1227.55	12.78	1288.08
SDs		0.13	6.82	0.08	23.32	0.05	6.98	0.10	4.89	0.05	1.33	0.05	9.80
CVs		0.99	0.55	0.64	1.61	0.38	0.56	0.75	0.37	0.38	0.11	0.38	0.76
05 - AVGs	100	12.92	1249.99	13.11	1424.31	12.75	1253.23	12.81	1330.41	12.75	1240.72	12.86	1285.29
SDs		0.08	13.53	0.13	36.42	0.08	16.70	0.05	12.40	0.17	9.52	0.05	3.06
CVs		0.65	1.08	0.97	2.56	0.65	1.33	0.38	0.93	1.31	0.77	0.37	0.24
06 - AVGs	100	13.00	1248.11	13.14	1414.17	12.75	1247.93	12.86	1330.59	12.78	1230.42	12.86	1282.02
SDs		0.00	8.08	0.05	19.76	0.00	6.22	0.05	5.36	0.05	3.23	0.05	1.53
CVs		0.00	0.65	0.37	1.40	0.00	0.50	0.37	0.40	0.38	0.26	0.37	0.12
07 - AVGs	100	12.92	1237.81	13.11	1426.99	12.69	1246.72	12.86	1320.44	12.69	1230.65	12.86	1283.60
SDs		0.14	8.35	0.10	31.38	0.05	2.99	0.05	7.85	0.05	6.88	0.05	5.98
CVs		1.12	0.67	0.73	2.20	0.38	0.24	0.37	0.59	0.38	0.56	0.37	0.47
08 - AVGs	100	12.92	1250.26	13.08	1426.08	12.69	1251.20	12.83	1336.97	12.72	1226.55	12.78	1289.29
SDs		0.08	6.79	0.08	25.50	0.05	9.15	0.08	6.03	0.13	7.36	0.05	5.10
CVs		0.65	0.54	0.64	1.79	0.38	0.73	0.65	0.45	1.00	0.60	0.38	0.40
09 - AVGs	100	12.94	1239.35	13.11	1427.18	12.72	1247.50	12.86	1323.14	12.69	1229.89	12.81	1286.23
SDs		0.05	1.01	0.10	43.51	0.05	4.92	0.05	9.89	0.05	1.04	0.05	5.57
CVs		0.37	0.08	0.73	3.05	0.38	0.39	0.37	0.75	0.38	0.08	0.38	0.43
10 - AVGs	100	12.89	1251.00	13.11	1394.25	12.67	1252.11	12.83	1335.88	12.72	1225.22	12.72	1288.23
SDs		0.17	16.07	0.10	30.58	0.08	6.00	0.08	2.43	0.05	5.16	0.10	12.56
CVs		1.35	1.28	0.73	2.19	0.66	0.48	0.65	0.18	0.38	0.42	0.76	0.98
AVGs	100	12.94	1243.26	13.09	1419.45	12.70	1249.85	12.84	1329.65	12.71	1230.83	12.81	1285.75
SDs		0.05	5.88	0.04	15.01	0.03	3.65	0.03	5.17	0.04	4.46	0.05	2.60
CVs		0.35	0.47	0.31	1.06	0.27	0.29	0.22	0.39	0.33	0.36	0.37	0.20
% to A, C or E.	100	0.00	0.00	1.18	14.17	0.00	0.00	1.07	6.39	0.00	0.00	0.79	4.46
% to A.	100	-	-	-	-	-	-	-0.75	6.95	-	-	-0.94	3.42

then the earliest deadline. Also, each seeding strategy is used by deterministic ants for a number of times, equal to the number of customers to be visited in a problem instance, before switching to the other seeding strategy and in each time the seeded customer can be regarded as one of these cases - farthest in distance, second farthest in distance, third farthest in distance...etc or earliest deadline, second earliest deadline, third earliest deadline...etc. The seeding component of the deterministic ants is in order to make sure that each deterministic ant starts from a start point in the graph that is different from the start points of the other deterministic ants. Of course, letting the deterministic ants seed its solutions in that way help in discovering different parts of the search space.

CP2- The exploitation mode is substituted with a new mode that does not use the pheromone trail component and depends only on the visibility function in Equation 4.1 and could be used with the distance function or any other visibility function. In 90% probability, this mode is used.

CP3- The exploration mode is substituted with a new mode that does not use the probability distribution and the pheromone trail component. Instead, it uses simple heuristics that are used in order at each time once this mode is called. For example, if a deterministic ant has a counter 0, then that refers to the first simple heuristic (a). If the same ant wants to use this mode next time, then the second heuristic number (b) is going to be used. Now, if the counter has become five, next time the deterministic ant restarts working from counter 0 and so on. In 10% probability, this mode is used.

(a) Go to the customer with the largest demand.

(b) Go to the customer with the smallest demand.

(c) Go to the customer with the earliest dead line and his due date is the neatest to be reached.

(d) Go to the nearest customer in distance in the graph.

(e) Go to the customer that causes a vehicle to have the least waiting time.

(f) Go to the customer that could wait a lot for a vehicle to arrive.

CP4- The local and global updating of the pheromone trail components are switched off. Therefore, the local pheromone updating rule in Section 4.4.6 will not be used after applying any of the two modes CP2 and CP3 enumerated above and the global pheromone update rules in Sections 4.4.9 are not going to be considered either.

Later after experimenting with the more deterministic ants in the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt as in Table 4.29 and checking of how well they are against the pheromone ants on the problem set R1 of the problem group PG100, it can be said that the performances and the results of such systems have worsen on average in a range between 0.79% to 3.14% for NV and 5.53% to 11.64% for TD.

However interestingly in comparison to the DACS 03 that uses the pheromone ants, the system DACS+HLS+2-Opt with its more deterministic ants brings the least deteriorations in terms of NV and TD together and can improve on average significantly the NV results by -0.94% according to the Student's t-test and Wilcoxon signed rank test. Also, it should be reminded that when the DACS 03 system using the pheromone ants gains better TD results by 4.48%, it is not because of using and updating the pheromone trails only.

This DACS 03 system with its pheromone ants is still using the local search of triple moves (M1 to M3) and a lot of that goodness is due to that local search rather than the pheromone ants themselves. For instance, the system DACS+HLS+2-Opt that uses more deterministic ants can recover, in terms of TD, 61.51% of what is lost because of using more deterministic ants in the system DACS 03 instead of the pheromone ants.

Of course, this kind of recovery is due to the usage of the hybrid local search HLS and the move 2-Opt. As a result, this is an indication that the system with more fiddling to the local search in particular can lead into having performance as good as the performance of a system that uses the pheromone ants.

The interesting discovery, mentioned above, proves also the point started with in which the ants with being more deterministic could also bring solutions that are competitive in terms of NV with those ants that depend alone on the usage and the update of the pheromone trail components locally and globally.

Table 4.29: Tests on ants that are more deterministic in the three systems DACS 03, DACS+HLS and DACS+HLS+2-Opt on the problem set R1 of the problem group PG100.

		A) DACS 03		B) DACS 03		C) DACS+ HLS		D) DACS+ HLS		E) DACS+ HLS+ 2-Opt		F) DACS+ HLS+ 2-Opt	
Phr. ants		Yes		No		Yes		No		Yes		No	
Det. ants		No		Yes		No		Yes		No		Yes	
Batch No.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	100	12.92	1242.24	13.31	1391.32	12.64	1244.87	12.78	1363.18	12.72	1230.32	12.81	1291.83
SDs		0.22	5.89	0.05	17.97	0.10	10.52	0.05	21.78	0.13	3.89	0.05	4.62
CVs		1.71	0.47	0.36	1.29	0.76	0.84	0.38	1.60	1.00	0.32	0.38	0.36
02 - AVGs	100	12.94	1238.30	13.36	1374.30	12.69	1246.05	12.86	1358.26	12.75	1235.05	12.81	1299.62
SDs		0.10	13.97	0.10	35.65	0.05	8.82	0.05	8.68	0.14	4.77	0.05	18.11
CVs		0.74	1.13	0.72	2.59	0.38	0.71	0.37	0.64	1.13	0.39	0.38	1.39
03 - AVGs	100	13.03	1238.88	13.36	1377.38	12.72	1253.55	12.78	1349.40	12.67	1231.88	12.83	1298.84
SDs		0.05	4.00	0.05	1.66	0.13	1.71	0.05	11.23	0.08	4.78	0.00	8.26
CVs		0.37	0.32	0.36	0.12	1.00	0.14	0.38	0.83	0.66	0.39	0.00	0.64
04 - AVGs	100	12.89	1236.63	13.25	1405.20	12.69	1255.30	12.92	1336.89	12.64	1227.55	12.81	1285.02
SDs		0.13	6.82	0.00	6.38	0.05	6.98	0.08	12.53	0.05	1.33	0.13	4.61
CVs		0.99	0.55	0.00	0.45	0.38	0.56	0.65	0.94	0.38	0.11	0.99	0.36
05 - AVGs	100	12.92	1249.99	13.36	1395.21	12.75	1253.23	12.92	1343.65	12.75	1240.72	12.81	1306.96
SDs		0.08	13.53	0.13	12.77	0.08	16.70	0.08	11.94	0.17	9.52	0.05	17.30
CVs		0.65	1.08	0.95	0.92	0.65	1.33	0.65	0.89	1.31	0.77	0.38	1.32
06 - AVGs	100	13.00	1248.11	13.42	1361.11	12.75	1247.93	12.89	1334.28	12.78	1230.42	12.78	1289.58
SDs		0.00	8.08	0.00	22.78	0.00	6.22	0.05	6.95	0.05	3.23	0.13	7.46
CVs		0.00	0.65	0.00	1.67	0.00	0.50	0.37	0.52	0.38	0.26	1.00	0.58
07 - AVGs	100	12.92	1237.81	13.28	1391.92	12.69	1246.72	12.78	1342.11	12.69	1230.65	12.89	1297.53
SDs		0.14	8.35	0.05	16.25	0.05	2.99	0.10	2.37	0.05	6.88	0.05	10.58
CVs		1.12	0.67	0.36	1.17	0.38	0.24	0.75	0.18	0.38	0.56	0.37	0.82
08 - AVGs	100	12.92	1250.26	13.36	1394.89	12.69	1251.20	12.89	1360.52	12.72	1226.55	12.78	1313.16
SDs		0.08	6.79	0.10	36.51	0.05	9.15	0.05	24.27	0.13	7.36	0.10	3.02
CVs		0.65	0.54	0.72	2.62	0.38	0.73	0.37	1.78	1.00	0.60	0.75	0.23
09 - AVGs	100	12.94	1239.35	13.33	1399.50	12.72	1247.50	12.81	1356.68	12.69	1229.89	12.83	1307.14
SDs		0.05	1.01	0.00	5.19	0.05	4.92	0.05	13.98	0.05	1.04	0.00	19.56
CVs		0.37	0.08	0.00	0.37	0.38	0.39	0.38	1.03	0.38	0.08	0.00	1.50
10 - AVGs	100	12.89	1251.00	13.39	1388.74	12.67	1252.11	12.89	1345.60	12.72	1225.22	12.81	1299.79
SDs		0.17	16.07	0.10	8.18	0.08	6.00	0.10	17.52	0.05	5.16	0.05	6.87
CVs		1.35	1.28	0.72	0.59	0.66	0.48	0.75	1.30	0.38	0.42	0.38	0.53
AVGs	100	12.94	1243.26	13.34	1387.95	12.70	1249.85	12.85	1349.05	12.71	1230.83	12.81	1298.95
SDs		0.05	5.88	0.05	13.24	0.03	3.65	0.06	10.16	0.04	4.46	0.03	8.62
CVs		0.35	0.47	0.38	0.95	0.27	0.29	0.46	0.75	0.33	0.36	0.25	0.66
% to A, C or E.	100	0.00	0.00	3.14	11.64	0.00	0.00	1.16	7.94	0.00	0.00	0.79	5.53
% to A.	100	-	-	-	-	-	-	-0.67	8.51	-	-	-0.94	4.48

4.8 A Summary of Chapter 4

Many variants of multiple ant colony systems, which have a lot of similarities with the well-known MACS-VRPTW system in [4], are studied and experimented with in this chapter and there are many motivations, as described in Section 4.3, behind the investigation of such systems. The main message in this chapter is to show that switching off the local search in a double ant colony system DACS, like in the detailed description in Section 4.4, will lead to performance and results that are far worse than the idea of putting off the usage and the update of the pheromone trails.

The start point of these experimentations has been in Section 4.5.1 by checking the effects of a local search called XCHNG, which its move operator works similarly as in Figure 4.9 to that of the well-known local search of CROSS-exchanges mentioned in [8] [5] [28] and [4], on the performance and results of a system called DACS 01. Therefore, the system DACS 01 that uses the same parametric values in MACS-VRPTW and with many guessed ones is checked with and without XCHNG on the problem group PG100 in Section 2.2 for three runs of 300 to 400 seconds according the experimental methodology in Section 4.2. Without XCHNG, DACS 01 is doing horribly bad. Although DACS 01 is doing fine with XCHNG, however the performance and results of DACS 01 on average are poor unfortunately, by 6.37% and 25.59% for NV and TD respectively, in comparison to the those of the other VRPTW algorithms of the literature in Table 4.3.

The poor performance and results have caused to do more experimental work on DACS 01 and to try to see what are the missing components should be considered. For that at the beginning in Section 4.5.2, the way in which the pheromone trails are initialized in the colonies VMIN and DMIN is changed from the inverse of the distance between any two nodes $(1/d_{c_i c_j})$ to V-1 and V, where V equals the number of vehicles of the best global solution found so far. So using the new way, each pheromone trail in VMIN is initialized with V-1 whereas in DMIN the value V is used instead. Changing the way of pheromone initialization has resulted in deteriorating the performance on the problem sets R1-100 and R1-200, on average, in terms of TD by 8.22% and later in discarding it. In addition, changing the evaporation values of the pheromone trails has not given any better performance and results either.

Then, the moment the pheromone trail initialization and re-initialization with

$1/(n.J_{\Psi}^{gb})$ is tried as in Section 4.5.3, the performance and results have improved, on average by -3.31%, significantly in terms of the number of vehicles on the problem set R1 of the problem group PG100. But, the performance and results are still not as good as those of the other VRPTW algorithms of the literature in Table 4.3. Additionally in comparison to DACS systems that use pheromone evaporation values like 0.0, 0.5, 0.8 and 1, it can be realized in Section 4.5.3 that the DACS system with the pheromone evaporation value 0.1 is the best on average, by -4.14% for NV and -0.09% for TD, in terms of performance. Therefore, the parametric values in any newly improved DACS system are kept as they are in the system MACS-VRPTW [4] in order to avoid any uncertainty about whether the DACS system is behaving or not as that system.

Later on, the reconfiguration of the cycles in the coordinator DACS 01, the colonies VMIN and DMIN and the XCHNG local search is attempted as in Section 4.5.4. As a result, reconfiguring the cycles has improved significantly, on the problem set R1 of the problem PG100, the TD results on average by -1.03% but such attempt is not good enough. Furthermore in Section 4.5.5, when the coordinator DACS 01 and its colonies has used threads and synchronization, it has improved the TD results on R1 by -0.90% on average but it does not work well enough in getting the performance at least as comparable to those of other VRPTW algorithms in the literature. Also, it has made the performance a little bit inconsistent in terms of NV as the standard deviation SD here equal to 0.15 in comparison to the SD 0.09 of DACS 01. As a consequence, this idea of using threads is ignored.

In addition, different kinds of local searches as in Section 4.5.6, which use an individual move operator each, are tried instead of XCHNG on the problem set R1 of the problem group PG100 but they did not improve the performance and results on their own in any way to be as close as to those of the other VRPTW algorithms in the literature. Although the local search of 2-Opt in Section 4.5.6 is the only one that has enhanced significantly on average the TD results by -1.49% but the main obstacle with it is that 2-Opt is not working with other move operators. Therefore, it is not good enough for the local search to let 2-Opt work on its own individually.

Later in Sections 4.6.1 and 4.6.2, a local search of triple moves (M1 to M3 in Section 4.4.8) is tested in a system called DACS 02 for three runs according to the experimental methodology in Section 4.2 on the problem groups PG100, PG200 and

PG400 in Section 2.2. Each problem instance of a group is used by DACS 02 for a limited amount of CPU time in seconds - 100, 300, 400, 600, 1200, 1800, 2400 or 4800. Here, the aim is to see the effect of trying a number of different intra and inter route improvement moves when working together rather than applying them individually.

On PG100, the performance and results of the DACS 02 system is enhanced dramatically on average by -17.04% for TD and -1.67% for NV on the problem group PG100 in comparison to the system DACS 01 but still DACS 02 is not as competitive as the other VRPTW algorithms of the literature in Table 4.3. Also, the performance and results of DACS 02 on the problem groups PG200 and PG400 are not so bad in contrast to those of the other VRPTW algorithms in the literature. On PG200, DACS 02 is outperformed on average by 4.63% and 9.98% for NV and TD respectively in comparison to other VRPTW algorithms in Table 4.12 whereas on PG400 the system is inferior to the algorithms in Table 4.13 by 5.40% for NV and 22.63% for TD.

Then in order to improve further the performance and results of DACS 02, ants that build their solutions simultaneously in parallel are tried as in Section 4.6.3 rather than sequentially as in DACS 02 but the parallel ants have not led into any significant improvement on average as indicated from the slightly deteriorated percentages 0.64% and 0.29% for NV and TD respectively on the problem set R1 of the problem PG100. Furthermore in Section 4.6.4, the nearest neighbourhood heuristic NN is changed as an initialization technique into the insertion heuristic SI1-Like 01, mentioned in Section 5.4, but that change, on average with -0.04% for NV and -0.74% for TD, is not successful enough on PG100 to make the performance as competitive as the other VRPTW algorithms in the literature to let it dominate the DACS system that uses the NN heuristic. After any amount of CPU time elapsed, the DACS system that uses SI1-Like 01 can dominate in terms of NV and TD together only on three out of six problem sets. Of course, this lack of domination is reflected also on many problem instances in which the DACS system that uses NN is able to beat in them the DACS system using SI1-Like 01. For that, using SI1-Like 01 instead of NN as an initialization technique is put down.

Also in Section 4.6.5, two kinds of candidate lists, distance-oriented and time-oriented, are tested and used in the components of the probabilistic state transition

rule with its exploitation and exploration parts, the insertion procedure and the local search of triple moves (M1 to M3). According to the percentages of deviations 0.44% for NV and −0.38% for TD on the problem set R1 of the problem PG100, there is not any significant difference seen between the usages of the two kinds of candidate lists described earlier. Later in Section 4.6.6, new local search moves to insert customers near the duplicated depot nodes of solutions are introduced and such moves have led into a better TD result on R1 by −2.09% on average. But, the deterioration, on average, in terms of NV by 0.43% and the time such moves take in trying to put the customers near depots has been the main discouraging factor in including them in the local search of triple moves. Also, the way the pheromone trails are updated locally and globally is changed but such research idea has not led, on R1, into any enough change on average, as indicated from −0.43% for NV and −0.44% for TD, in the performance and results of the DACS 02 system to be as good as those of the other VRPTW algorithms in the literature.

For that as in Section 4.7.1, a method called the push forward and push backward strategy PFPBS is included in the DACS system in a way to allow the ants store, use and update information similar to the point numbers enumerated in I1 to I6 and mentioned in Section 4.4.11. Afterwards, DACS 03 is tested for thirty runs (during the six different amounts of CPU times of 100, 300, 400, 600, 1200 and 1800 in seconds) and checked about its performance and results in the best, average and worst-case scenarios on the six problem sets of the problem group PG100. Later, it can be said that the performance and results of DACS 03 have improved successfully on average by −3.07% for NV and −3.25% for TD and are getting closer to the performance and results of the other VRPTW algorithms of the literature in Table 4.3. However, DACS 03 with the PFPBS strategy is still outperformed by many of those algorithms on all the six problem sets.

In order to improve the performance and results of the DACS 03 system further, specific types of multiple ant colony systems, which have colonies with objectives or goals that are different from the objectives or the goals pursued by the VMIN and DMIN colonies of DACS 03, are studied and investigated in Section 4.7.2. Therefore at first, four types of triple ant colony systems or TACSs with three colonies each are tried on the problem set R1 of the problem group PG100. Two of the three colonies are VMIN and DMIN regularly used while the third colony is a one that

tries to minimize or to maximize either the totals of customer waiting times as in the colonies CWTsMIN and CWTsMAX or the totals of vehicle waiting times as in the colonies VWTsMIN and VWTsMAX. The four types of TACS systems are not able to improve the performance and results in any way as indicated on average from the deteriorated percentages 0.15% and 0.54% for NV and TD respectively. For that, a quadruple ant colony system or QACS that includes the colonies VMIN, DMIN, CWTsMAX and CWTsMIN and a special type of a double ant colony system that includes a colony called TMIN (that minimises the total of time consumed in vehicle traveling, waiting and servicing) instead of DMIN are tested also on R1 but such systems are unsuccessful and ineffective.

Then, the hybrid local search HLS, which combines the local search of triple moves (M1 to M3) and the insertion procedure is added to the DACS 03 system to form a system called DACS+HLS as in Section 4.7.3 in order to improve the performance and results in terms of the number of vehicles. According to Student's t-tests and Wilcoxon signed rank tests on the problem set R1 of the problem group PG100, the addition of HLS into DACS 03 is discovered to be significant in terms of the number of vehicles at the 99.9% level. Therefore during each of the six different CPU time amounts of 100, 300, 400, 600, 1200 and 1800 in seconds, DACS+HLS is tested for thirty runs on the six problem sets of the problem group PG100 in order to see how its performance behaves and how its results are going to be in the best, average and worst case scenarios in comparison to those of DACS 03.

Later, the performance and therefore the results of the DACS+HLS system have improved in terms of the number of vehicles dramatically on the problem sets R1, RC1, R2 and RC2 of the problem group PG100 and this is on average by -2.43%. However in terms of the total traveled distances, adding the hybrid local search HLS has led, by 0.59% on average, into computing deteriorated traveled distances on all the six problem sets. In addition when looking at the VRPTW algorithms of the literature in Table 4.3, DACS+HLS for the first time is able to bring performance and results that are as good as and competitive with their performances and results.

Consequently in Section 4.7.4, the 2-Opt move variant as an inter-route improvement operator is added into the local search of triple moves (M1 to M3) and the hybrid local search HLS in a system called DACS+HLS+2-Opt in order to improve the performance and results in terms of the total of traveled distances. Then, adding

the 2-Opt move variant into the DACS+HLS system, as in DACS+HLS+2-Opt, is discovered to be significant in terms of the total of traveled distances at the 99.9% level, according to Student's t-tests and Wilcoxon signed rank tests, on the problem set R1 of the problem group PG100. Thus in order to see its performance and results in the best, average and worst case scenarios, DACS+HLS+2-Opt is examined for thirty runs on the six problem sets of the problem group PG100 during each of the six different CPU time amounts of 100, 300, 400, 600, 1200 and 1800 in seconds.

Of course, the addition of the 2-Opt move variant into the DACS+HLS system has led into improving the performance and results of the DACS+HLS+2-Opt system in the best, average and worst case scenarios in terms of the total of traveled distances on all the six problem sets when they are compared with those of DACS+HLS. At the level of all the different amounts of CPU time in seconds, the traveled distances are on average enhanced by -1.14% on all the six problem sets. Also, adding the 2-Opt move variant as in DACS+HLS+2-Opt has resulted in worsening the number of vehicles slightly on average by 0.05% from that of DACS+HLS. Therefore, the reduction in terms NV has not significantly changed a lot and DACS+HLS+2-Opt is still able, compared to the other VRPTW algorithms of the literature in Table 4.3, to get competitive performance and results.

After the experimental campaign with the three systems DACS 03, DACS+ HLS and DACS+HLS+2-Opt on the six problem sets of the problem group PG100, the three DACS systems are tested for three runs each, after either 2400 or 4800 seconds, on the six problem sets of the problem groups PG200 and PG400 as in Section 4.7.5. Then as expected, adding the push-forward and push-backward strategy as in DACS 03 has led into improving the performance and results on average by -2.81% for NV and -5.17% for TD on PG200 and by -2.16% and -10.76% on PG400.

Also, the addition of the hybrid local search HLS as in the DACS+HLS system has improved the performance and results on average in terms of the number of vehicles by -1.02% for PG200 and by -1.92% for PG400 but with a little bit of deterioration by 2.70% and 4.66% in terms of the total of traveled distances. Later on PG200 and PG400, when the 2-Opt move variant is tested as in the system DACS+HLS+2-Opt, the performance and results in terms of the total of traveled distances have improved on average with -3.79% and -5.89% for PG200 and PG400 respectively and the reduction in terms of the number of vehicles has not changed

and it is as in DACS+HLS. Of course, adding the HLS and later the 2-Opt move variant has made the performances and results of the two systems DACS+HLS and DACS+HLS+2-Opt on the problem groups PG200 and PG400 get closer to and be more competitive with those of the VRPTW algorithms of the literature in Tables 4.12 and 4.13.

Thereafter, the constructive phase of the routing builder used in the three systems DACS 03, DACS+HLS and DACS+HLS+2-Opt is changed into a constructive phase that depends on a Savings algorithm as in Section 4.7.6 that has some similarities with the Savings algorithm of Clarke and Wright mentioned in [10]. Then, the three DACS systems are tested with two saving functions, distance-oriented and time-oriented functions, on the problem set R1 of the problem PG100 but later such saving ants have not led into making any significant improvement in the performance and results of any of the three DACS systems.

Then in an effort to know which component (of the pheromone trails and the local searches) is causing the synergetic effects of the ants, the three systems DACS 03, DACS+HLS and DACS+HLS+2-Opt as in Section 4.7.7 are tested, on the problem set R1 of the problem group PG100, by switching off the usage and the updating locally and globally of the pheromone trails. Despite the significant deterioration of the performance and results at the 99.9% level after putting off the pheromone trails according to Student's t-tests and Wilcoxon signed rank tests, however interesting things are discovered as well. Firstly, the performances of the systems, which use the HLS and 2-Opt and switch off the pheromone trails, deteriorate on R1 far lesser on average than (in a range between 4.46% to 6.39% for TD and 0.79% to 1.07% for NV) that of the DACS 03 that does not use and update them.

In DACS 03 where there is no use for the HLS, the 2-Opt and the pheromone trails, the deterioration in terms of performance on R1 is on average equal to 14.17% and 1.18% for TD and NV respectively. Secondly, the systems that apply HLS and 2-Opt and do not use and update the pheromone trails recovers in terms of TD, on average by 50.97% to nearly 75.88%, what is lost in DACS 03, on R1, because of switching off the pheromone trails. Thirdly on R1, the NV results of the systems DACS+HLS and DACS+HLS+2-Opt without the pheromone trails are significantly better on average by -0.75% to -0.94% in comparison to the system DACS 03 that use and update the pheromone trails and this is according to the statistical tests of

Student's t-test and Wilcoxon signed rank.

Finally, when the system DACS 03 applies the pheromone trails, the goodness of the TD result 1243.26, achieved on average as in Table 4.28 on R1, is not because of the usage and the update of the pheromone trails since DACS 03 is still using the local search of triple moves. As a result, more fiddling with how the local search is designed can lead into having a system that gets quality performance and without any pheromone trail usage and update.

The discoveries, described above because of switching off the pheromone trails, have been encouraging to replace the pheromone ants with more deterministic ones as in Section 4.7.8, which are made after doing the four changes mentioned below. Therefore at first, two seeding strategies of the farthest in distance and the earliest deadline as mentioned in Section 5.4 are added to help in creating solutions with different starts in the search space and the exploration mode is changed into a new mode that uses the simple heuristics a to f in Section 4.7.8 instead of the probability distribution and does not use the pheromone trail component. Then, the exploitation mode is modified with a new mode that does not use the pheromone trail component and depends only on the visibility function in Equation 4.1. Furthermore, the usage and update of the pheromone trails locally and globally are switched off.

With all that described earlier in mind and on the problem set R1 of the problem group PG100, it can be seen in Table 4.29 interestingly that the DACS+HLS+2-Opt system that uses more deterministic ants deteriorates the results the least and is able significantly (according to Students t-test and the Wilcoxon signed rank test) to have on average a better NV result by -0.94% in comparison to that of the DACS 03 system, which uses the pheromone ants and still applies the local search of triple moves (M1 to M3). Also, it is able to recover, on R1, in terms of TD 61.51% of what is lost because of using the more deterministic ants instead of the pheromone ants in DACS 03. This interesting discovery has made the author to conclude that deterministic approaches, as in Chapter 5, with more fiddling to the local search in particular could bring performance as well as the approaches that use and update the pheromone trails.

Chapter 5

Deterministic Approaches

As a continuation to what is discussed in Sections 4.7.7 and 4.7.8, this chapter investigates variants of approaches, which capture under-constrained tours, with its deterministic nature in the sense of not using any random component and as a result the aim or the message to pass at this point is as follows.

> Deterministic approaches with more fiddling to its components could show at some stage signs of the ability of improving the performance of systems that use the pheromone ants.

Deterministic approaches as in [8] and [5] are algorithms that do not depend on ingredients, such as the pheromone trail components in multiple ant colony systems or the variation operators of crossover and mutation in evolutionary algorithms, in order to get competitive performance and results. Also, the deterministic algorithms in [8] and [5] do not depend on any random component that might help in searching for very good quality solutions and therefore this means that they deliver the same result every time when they are run. So, it is sufficient to run such algorithms once.

This seems like a big advantage compared to evolutionary or ant colony algorithms, which are non-deterministic and may need to be run several times in order to obtain an acceptably good result. Consequently, running a deterministic algorithm, like in [8] and [5], for a single run suggest also that such deterministic approaches are consistent and reliable in getting out good quality solutions and therefore competitive performance and results.

For that, what is described above is one of the main reasons behind the study and investigation to deterministic approaches in this chapter. Another reason is the

interesting discovery, explored in Section 4.7.8, which indicates that the artificial ants in a DACS system with being more deterministic could also lead into having performance and results that are as good as the performance and the results of another DACS system that uses the pheromone ants. The final reason is to see if there are any potential points that might lead to a successful merge between deterministic approaches and multiple ant colony systems and to check what the possible effects of such merge are - in the long term.

Thus, this raises the question mentioned below. Can a deterministic algorithm, during an allocated amount of CPU time, be as good at search for quality solutions as any of the non-deterministic approaches considered in this thesis? Since the non-deterministic approaches seem to rely heavily on some deterministic local search ingredient in order to get good results, it is worth asking whether the non-deterministic component is really needed in optimization techniques. The experiments in this chapter explore this question.

In Section 5.1, this chapter starts with a brief introduction to deterministic approaches and in Section 5.2 the methodology of experimentation in deterministic approaches, studied and investigated in this thesis, is talked about. Then, the SI1-Like deterministic approach, best explored so far as in Section 5.3, is described. SI1-Like is short for Solomon Insertion1-Like, which has some similarities in its routing builder method to that of the insertion heuristic I1 of Solomon [1].

Later, the effect of the maximization function of SI1-Like is investigated in Section 5.4. Then in Section 5.5, a seeding strategy called "No Seeding" and what it can do are introduced. Thereafter, the varying of the parametric values of a_1 and a_2, used in Equation 2.23, and its effects are explored in Section 5.6.

Also, the removing heuristic of under-constrained tours in a solution is examined in Section 5.7. Afterwards in Section 5.8, the insertion procedure of more to less constrained customers is tested. Next in Sections 5.9, 5.10 and 5.11, modifications to the hybrid local search HLS, like the handling of feasible and infeasible solutions at the same time, the ordering of the tours in an ascending way from the least to the largest size and the usage of inversion moves, are checked out.

In addition, an 'eject and insert' strategy, which ejects visited customers into other tours of a solution and inserts under-constrained customers instead, is looked at its effects on its own as in Section 5.12 and when it is merged with local searches as

in Section 5.13. Subsequently in Section 5.14, the inclusion of waiting time functions in SI1-Like are tried. Finally, this chapter is summarized in Section 5.15.

5.1 An introduction to deterministic approaches

After discussing the motivations and the reasons at the beginning of this chapter that are behind the investigation to deterministic approaches, this section talks briefly about the best deterministic approach so far (in trying to solve the VRPTW problem), which is called as Reactive Variable Neighbourhood Search or simply RVNS in [8] and [5]. The RVNS algorithm is described in detail in Section 2.7.5 and is mentioned here as a reminder.

In a deterministic approach like RVNS, initial solutions are created using route construction or cheapest insertion heuristics that use a number of combinations of parametric values and seeding schemas or scenarios. The cheapest insertion heuristics used in [8] and [5] have a lot of similarities with the studies of Solomon [1] on insertion heuristics.

Then in [8] and [5], a route elimination procedure is applied to each of the initial solutions created in order to reduce the number of routes until no more routes can be eliminated. Later, the solutions with the smallest number of vehicles are gathered so as to be improved using route improvement procedures that are designed in particular only for reducing distance. The cost function of the route improvement procedures is changed at some stage when it is necessary to escape local minima.

5.2 Experimental methodology

In this chapter, twenty SI1-Like deterministic approaches, which are related to each other according to the map mentioned in Figure 5.1, are tested. The methodology in the experimentation, used in each of the twenty SI1-Like deterministic approaches, is as follows. The approaches are implemented using the Java programming language and run on a PC machine with the following hardware features - Pentium IV with 2.66 GHz speed and 512 MB RAM.

Because of the lack of any random component, each SI1-Like is executed for a single run only on the problem group PG100 in Section 2.2, which has Solomon's

six problem sets [1] - namely R1, C1, RC1, R2, C2 and RC2. Each SI1-Like approach runs on each problem set for an averaged amount of CPU time in seconds and such averaged amount depends not only on the main features of each problem set and how they are configured but also on the structural properties of each problem instance in that problem set. Now, the performances of the different SI1-Like deterministic approaches can not be compared directly with the performances of other deterministic and non-deterministic algorithms of VRPTW, which are mentioned in the literature as in Tables 5.1 and 4.3, on the basis of CPU time due to hardware and software differences. However, such different approaches can be compared indirectly to know more about the performance, the behavior and the ability of each SI1-Like in producing its quality results.

In the SI1-Like approaches tried in Tables 5.2 to 5.6 and in Tables G.1 to G.15, the NV and TD results of each approach can be repeated on each problem instance and therefore each problem set. On each problem set, the results are averaged (over the number of the problem instances of the problem set involved) and therefore reported after taking into consideration the NV and TD results computed and reported; near the problem instance numbers 1 to 12. Since there is no random component in all the SI1-Like approaches, the standard deviation values are equal to zero. However in order to make sense of the quality of such results, percentages of deviations from some particular algorithms are reported for NV and TD where it is appropriate.

5.3 Deterministic Algorithm

This section describes in Sections 5.3.1 to 5.3.5 and in detail the deterministic algorithm best explored so far, which does not use any random component and has some similarities in its routing builder, described in Section 5.3.2, with the insertion heuristic I1 of Solomon [1]. Therefore, such algorithm is called SI1-Like as an abbreviation to Solomon Insertion1-Like.

However, it has more additional components such as the usage of a removal heuristic of under-constrained tours, a hybrid local search HLS and an 'eject and insert' strategy as mentioned in Sections 5.3.3, 5.3.4 and 5.3.5 respectively. Also in Sections 5.3.1, 5.3.2 and 5.3.4, the varying of the parametric values (of a_1 and

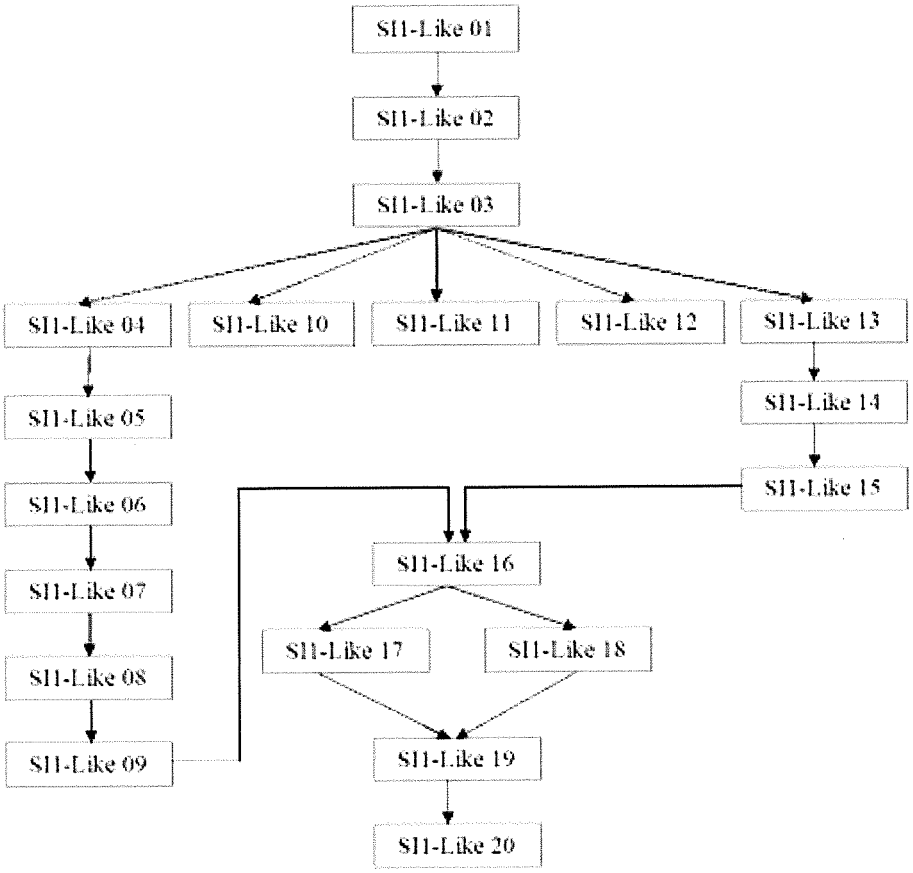


Figure 5.1: Map of SI1-Like deterministic approaches.

Table 5.1: Comparison between different deterministic algorithms of VRPTW on the problem group PG100. The results of each algorithm are averaged over the number of runs done - check Table I.10 for more information.

		R1		C1		RC1		R2		C2		RC2	
Algorithm	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
AKRed [60]	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5]	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
RVNSb [5]	4950	11.92	1222.12	10.00	828.38	11.50	1389.58	2.73	975.12	3.00	589.86	3.25	1128.38
CL1 [77]	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
I1 [1]	24 - 63	13.58	1436.70	10.00	951.90	13.50	1596.50	3.27	1402.40	3.13	692.70	3.88	1682.10
I1-AD [14]	8.18 - 40	13.83	1482.53	10.00	960.81	13.50	1610.78	3.18	1355.24	3.25	740.93	4.00	1684.43
PR1 [11]	1176	13.33	1509.04	10.67	1343.69	13.38	1723.72	3.09	1386.67	3.38	797.59	3.63	1651.05
TP [13]	108	13.00	1356.92	10.00	916.67	13.00	1514.29	3.18	1276.00	3.00	644.63	3.71	1634.43
AD [14]	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78]	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79]	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37

a_2) and the usage of a "No Seeding" strategy, the waiting time functions and a different insertion procedure are other components that do not exist in Solomon's I1 heuristic.

5.3.1 Solomon Insertion1-Like Algorithm - SI1-Like

The SI1-Like algorithm in Figure 5.2 creates each time a solution using its routing builder in Figure 5.3 and with the help of the parametric values (of μ, λ, a_1 and a_2) determined in some way and one of three seeding strategies (the farthest in distance as in Equation 5.1, the earliest deadline as in Equation 5.2 or no seeding strategy at all). As in Figure 5.2 and before getting into using the routing builder call message of step d, the algorithm chooses the kind of the seeding strategy needed in addition to the parametric value of λ, which could be either 1 or 2. The seeding strategy is used, always, to visit the first customer in each new route of a solution before inserting any remaining customers in that route.

$$value = \max_{i=1}^{N}[d_{c_0c_i}] \tag{5.1}$$

$$vw_{c_i} = \max(0, r_{c_i} - d_{c_0c_i})$$

$$value = \min_{i=1}^{N}[d_{c_i} - (d_{c_0c_i} + vw_{c_i})] \tag{5.2}$$

Then, the values of the parameters of a_1 and a_2 are determined using the steps b and c in Figure 5.2. Therefore, when a_1 is decreased by 0.1, a_2 is increased by 0.1. At each step of 0.1, if a_1 equals a value, a_2 is going to be equal to 1.0 minus that value of a_1. The values of a_1 and a_2 ranges between 0 and 1 ([0-1], inclusive).

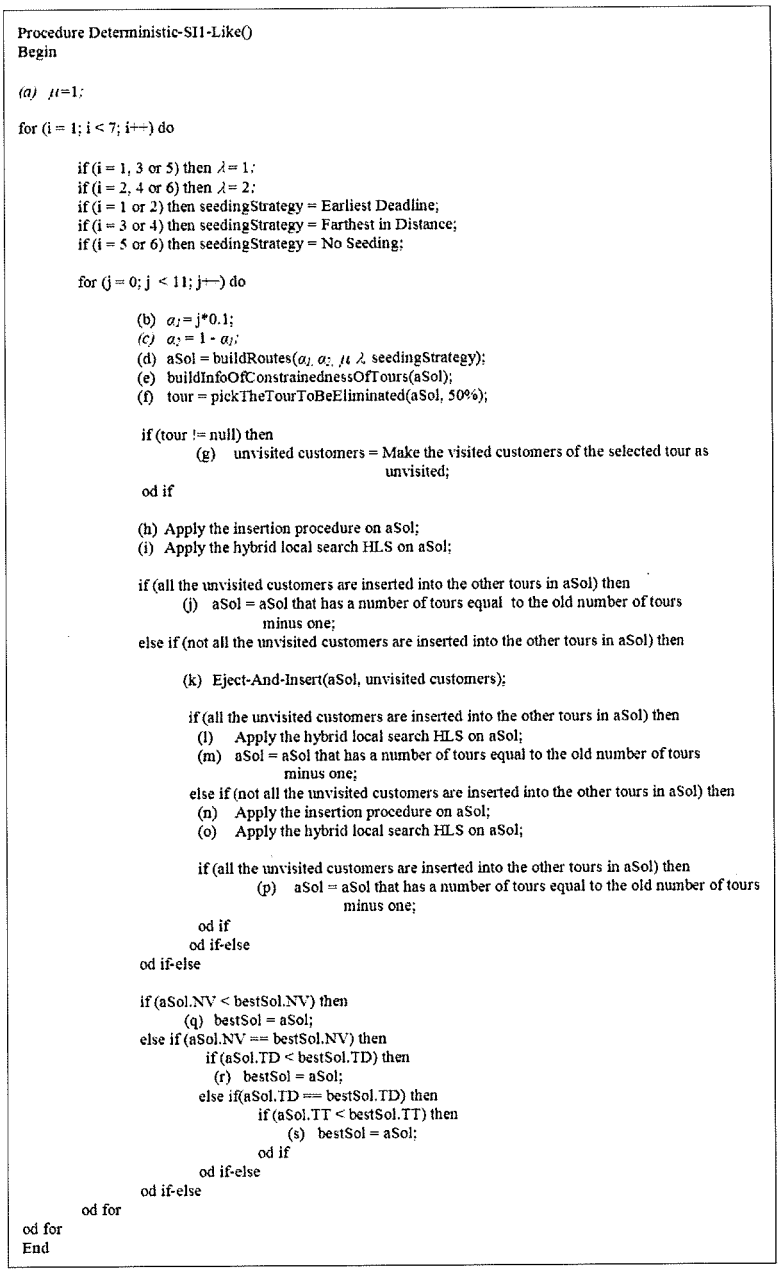
Once the parametric values and the seeding strategy are determined as explained previously, the solution is built on the route-by-route basis as described in Section 5.3.2. Thereafter as described in Section 5.3.3, SI1-Like applies a heuristic that removes an under-constrained tour with the least constrained customers. This heuristic uses some information, which is related to the routes of the built solution and how much their customers are constrained (i.e. the customers are very far from the depot, the demands are very large and/or the time windows are very tight).

On the basis of that, such heuristic helps in removing a particular "under constrained" tour so as to insert later its customers into other tours of the same solution. The customers, in such under-constrained tours, are less constrained in terms of how close they are from the depot, how small their demands are and how wide their time windows are.

Soon, the SI1-Like algorithm applies, a modified insertion procedure and an updated hybrid local search HLS as described in Section 5.3.4, to try to improve the solution created by trying to re-insert such least constrained customers back into the solution itself but in tours different from their original removed tour.

In the case where there is not any possibility to insert such least constrained customers, an 'eject and insert' strategy in Section 5.3.5 is used to try to insert them by ejecting some other visited customers into tours different from the one where they are located in. Later on, the solutions out of that strategy, whether they are feasible or not, are exposed to more local searching for quality solutions.

Any new best solution is determined in a way that depends on how much it reduces the number of vehicles as a primary goal and then the total of traveled distances as a secondary goal. If the traveled distances of two solutions are the same, then the solution with the least total of time consumed in vehicle traveling, waiting and servicing is chosen as the new best solution.



```
Procedure Deterministic-SI1-Like()
Begin

(a)  μ=1;

for (i = 1; i < 7; i++) do

        if (i = 1, 3 or 5) then λ = 1;
        if (i = 2, 4 or 6) then λ = 2;
        if (i = 1 or 2) then seedingStrategy = Earliest Deadline;
        if (i = 3 or 4) then seedingStrategy = Farthest in Distance;
        if (i = 5 or 6) then seedingStrategy = No Seeding;

        for (j = 0; j  < 11; j++) do

                (b)  α1 = j*0.1;
                (c)  α2 = 1 - α1;
                (d)  aSol = buildRoutes(α1, α2, μ, λ, seedingStrategy);
                (e)  buildInfoOfConstrainednessOfTours(aSol);
                (f)  tour = pickTheTourToBeEliminated(aSol, 50%);

                if (tour != null) then
                        (g)  unvisited customers = Make the visited customers of the selected tour as
                                                   unvisited;
                od if

                (h) Apply the insertion procedure on aSol;
                (i)  Apply the hybrid local search HLS on aSol;

                if (all the unvisited customers are inserted into the other tours in aSol) then
                        (j)  aSol = aSol that has a number of tours equal  to the old number of tours
                                    minus one;
                else if (not all the unvisited customers are inserted into the other tours in aSol) then

                        (k)  Eject-And-Insert(aSol, unvisited customers);

                        if (all the unvisited customers are inserted into the other tours in aSol) then
                          (l)   Apply the hybrid local search HLS on aSol;
                          (m)  aSol = aSol that has a number of tours equal to the old number of tours
                                      minus one;
                        else if (not all the unvisited customers are inserted into the other tours in aSol) then
                          (n)   Apply the insertion procedure on aSol;
                          (o)   Apply the hybrid local search HLS on aSol;

                          if (all the unvisited customers are inserted into the other tours in aSol) then
                                  (p)   aSol = aSol that has a number of tours equal to the old number of tours
                                                minus one;
                          od if
                        od if-else
                od if-else

                if (aSol.NV < bestSol.NV) then
                        (q)  bestSol = aSol;
                else if (aSol.NV == bestSol.NV) then
                          if (aSol.TD < bestSol.TD) then
                            (r)  bestSol = aSol;
                          else if(aSol.TD == bestSol.TD) then
                                  if (aSol.TT < bestSol.TT) then
                                      (s)  bestSol = aSol;
                                  od if
                          od if-else
                od if-else
        od for
od for
End
```

Figure 5.2: The pseudo code of the SI1-Like deterministic approach.

```
Solution buildRoutes(α1, α2, μ, λ, seedingStrategy)
Begin

(a)  tourIndex = 0;
(b)  Let maximumMeasureFlag = false and maximumMeasure = -1*largestDoubleValue;
(c)  Let size = the number of customers plus the depot in a problem instance;
If (seedingStrategy != No Seeding) then
  (d)  Put a seed customer, selected using seedingStrategy, in a tour that has an index equal to tourIndex;
od if
(e)  tourSize = Get the tour size of the tour located in tourIndex;

while (size != 0) do          //Watch the two break statements of this while loop below.

  for (i = 1; i < size; i++) do

      (f)  cu = Get an unrouted customer from the set of customers;
      (g)  visitedFlag = Check if the unrouted customer cu is visited;

      if (visitedFlag == true) then continue;

      for (j = 0; j < tourSize; j++) do          //The size of the tour is at least 1 because of the depot

        (h)  feasibleFlag = Check if inserting the unrouted customer cu at index j+1 is feasible or not;

        if (feasibleFlag == true) do

              (i)  Let ip = the node, customer or depot, that has the index j+2 and is located after the unrouted
                   customer cu;
              (j)  Let ip-1 = the node, customer or depot, that has the index j and is located before the unrouted
                   customer cu;
              (k)  Calculate c12(ip-1, cu, ip) using Equation 2.25, 5.3, 5.4 or 5.5;
              (l)  Calculate c11(ip-1, cu, ip), c12(ip-1, cu, ip) and c2(ip-1, cu, ip) according to Equations 2.23, 2.24 and
                   2.26 respectively;

              if (c2(ip-1, cu, ip) > maximumMeasure) do
                    (l)  maximumMeasure = c2(ip-1, cu, ip);
                    (m)  customerWithMaximumMeasure = unroutedCustomer;
                    (n)  maximumMeasureFlag = true;
              od if
        od if
      od for
  od for

  if (maximumMeasureFlag == true) then

        (o)  Put customerWithMaximumMeasure in a tour that has an index equal to tourIndex;
        (p)  tourSize = Get the tour size of the tour located in tourIndex;

  else if (maximumMeasureFlag == false)

        if (the number of visited customers is not equal to the number of customers in a problem instance) then

          if (the number of vehicles used is less than the number of vehicles allowed to use) then
                (q)  tourIndex = tourIndex + 1;
                If (seedingStrategy != No Seeding) then
                    (r)  Put a seed customer, selected using seedingStrategy, in a tour that has an index equal to
                         tour Index;
                od if
                (s)  tourSize = Get the tour size of the tour located in tourIndex;
          else if (the number of vehicles used is equal to the number of vehicles allowed to use) then
                (t)  break;          //Get out of the while loop.
          od if-else

        else if (the number of visited customers is equal to the number of customers in a problem instance) then
              (u)  break;          //Get out of the while loop.
        od if-else
  od if-else

  (v)  Let maximumMeasureFlag = false and maximumMeasure = -1*largestDoubleValue;
od while
(w)  return aSolution;          //All tours built with NV, TD and TT.
End
```

Figure 5.3: The pseudo code of the routing builder method that maximises the function in Equation 2.26.

5.3.2 A routing builder and its differences to the I1 heuristic

The routes of a solution are built sequentially or on the basis of a route-by-route as in Figure 5.3. In the case where the seeding strategies of the farthest in distance and the earliest deadline are used, a seed customer is first selected to create an initial route servicing this customer and in this case a vehicle leaves the depot to service the seed customer and comes back later to the depot.

The remaining customers are then inserted one by one into the initial route as follows. At each time, each unrouted customer c_u is calculated with values to the terms $c_1(i_{p-1}, c_u, i_p)$, $c_{11}(i_{p-1}, c_u, i_p)$, $c_{12}(i_{p-1}, c_u, i_p)$ and $c_2(i_{p-1}, c_u, i_p)$ as in Equations 2.23, 2.24, 2.25 and 2.26 between every two consecutive visited nodes i_{p-1} and i_p of the initial route. In some SI1-Like approaches as in Section 5.14, the value of the term $c_{12}(i_{p-1}, c_u, i_p)$ can be calculated differently also as in Equation 5.3, 5.4 or 5.5 mentioned below instead of that in Equation 2.25. The nodes i_{p-1} and i_p could be each a customer or a depot. Later, the unrouted customer that maximizes (between any two consecutive visited nodes) the value of the term $c_2(i_{p-1}, c_u, i_p)$ in Equation 2.26 is the one that will be inserted. Obviously, what is described in this paragraph is iterated until there is no more customers can be found to be feasibly injected.

At this point, another seed customer is selected to create a second route, and this route is filled using the remaining unrouted customers as described above. Of course, the route building process is repeated until all customers are routed. If the "No Seeding" strategy is used, then the route building process works similarly as described above but without any seeding at the start for new routes.

There are three main differences between the route building process, used here in Figure 5.3, and Solomon's I1 heuristic [1]. The first difference is that the route building process of the SI-Like approach uses the "No Seeding" strategy, which does not exist as a strategy in I1. Another difference is that the route building process of the SI1-Like approach uses a maximisation function only when building the routes while I1 uses at first a minimisation function and then it applies the maximization function later.

Consequently in the route building process of an SI1-Like approach, a solution is built according to the maximization function in Figure 5.3 without taking any consideration to find the minimum-cost feasible insertion place, in a currently used

route, for every unrouted customer c_u that is talked about in Section 2.5.

In a matter of fact, the route building process of an SI1-Like approach is more about maximising the value in Equation 2.26 regardless of any minimisation. Conversely in Solomon's I1 heuristic mentioned in Section 2.5, once I1 finds the best feasible place with the minimum cost between any two nodes of i_{p-1} and i_p, using Equation 2.23, for every unrouted customer c_u in a currently used route, only then the unrouted customer that maximises the value in Equation 2.26 is chosen as a visited node. Certainly in the route building process of the SI1-Like approach, Equation 2.23 is used only as a part of the calculation of the maximization function as in Equation 2.26 but without any other use or role as that (of the heuristic I1) described above.

The final difference from that of Solomon's I1 heuristic is that the route building process of the SI1-Like approach can calculate the value of the term $c_{12}(i_{p-1}, c_u, i_p)$ from the waiting time functions in Equation 5.3, 5.4 or 5.5 or from the servicing time functions $b_{i_p c_u}$ and b_{i_p} in Equation 2.25. However, the heuristic I1 uses only Equation 2.25 in calculating the value of the term $c_{12}(i_{p-1}, c_u, i_p)$. The terms $b_{i_p c_u}$ and b_{i_p} express respectively the start times of servicing at the customer i_p after and before inserting the customer c_u. In the three equations mentioned below, the two terms $tvws_{i_p c_u}$ and $tcws_{i_p c_u}$ describe each a total of (either vehicle or customer) waiting times up to node i_p in a route after inserting a customer c_u whereas the terms $tvws_{i_p}$ and $tcws_{i_p}$ represent each a total of the same kind of waiting times up to node i_p but before inserting the customer c_u.

$$c_{12}(i_{p-1}, c_u, i_p) = tvws_{i_p c_u} - tvws_{i_p} \tag{5.3}$$

$$c_{12}(i_{p-1}, c_u, i_p) = tcws_{i_p c_u} - tcws_{i_p} \tag{5.4}$$

$$c_{12}(i_{p-1}, c_u, i_p) = (tvws_{i_p c_u} + tcws_{i_p c_u}) - (tvws_{i_p} + tcws_{i_p}) \tag{5.5}$$

5.3.3 A removing heuristic of "under-constrained" tours

Next, the SI1-Like deterministic approach as in Figure 5.2 uses the concept of under-constrained tours in order to capture such tours. The general idea behind capturing such tours is somehow later to remove them and re-insert their customers via the

insertion procedure and the hybrid local search HLS; mentioned in Section 5.3.4, into the other tours of the solution. In this thesis, the under-constrained tours are the ones that have a percentage of under-constrained customers greater than or equal to 50%. Now, the under-constrained customers can be located based on three kinds of features mentioned below. If all these features are available in a group of customers in a tour, then there is a great possibility that this group is considered as a group of under-constrained customers.

i) How close the customers are from the depot?

ii) How wide the time windows of the customers are?

iii) How small in amounts the demands of the customers are?

The constrained-ness of a customer can be calculated from Equations 5.6, 5.7, 5.8 and 5.9, where n is equal to the number of customers in a problem instance. For that, Equations 5.6, 5.7 and 5.8 calculate the percentage values of the three features mentioned above of any customer. So, Equation 5.6 calculates a percentage value of how close is the customer c_i from the depot, while Equations 5.7 and 5.8 calculate the percentage values of how wide the time window of that customer is and how small in amount the demand of that customer is. In order to know if that customer is very constrained or not, the constrained-ness value can be compared with the threshold that can be calculated through Equations 5.10, 5.11, 5.12 and 5.13. Here, Equations 5.10, 5.11 and 5.12 calculate three percentage values as in Equations 5.6, 5.7 and 5.8.

If the constrained-ness value of a customer is greater than the threshold, then the customer will be regarded as a constrained customer. On the other hand, the customer will be referred to as an under-constrained customer, if the constrained-ness value of that customer is less than the threshold. Later in SI1-Like, when the tours are built, the under-constrained customers can be located easily.

$$p1_{c_i} = [d_{c_i c_0} / \max_{1 \leq i \leq n} d_{c_i c_0}] \times 100 \tag{5.6}$$

$$p2_{c_i} = -1 \times [(d_{c_i} - r_{c_i}) / \max_{1 \leq i \leq n} (d_{c_i} - r_{c_i})] \times 100 \tag{5.7}$$

$$p3_{c_i} = [q_{c_i} / \max_{1 \leq i \leq n} (q_{c_i})] \times 100 \tag{5.8}$$

$$CONSTRAINED = p1_{c_i} + p2_{c_i} + p3_{c_i}. \tag{5.9}$$

$$perc_I = [(\frac{\sum_{i=1}^{n}(d_{c_i c_0})}{n})/\max_{1\leq i\leq n}(d_{c_i c_0})] \times 100 \tag{5.10}$$

$$perc_{II} = -1 \times [(\frac{\sum_{i=1}^{n}(d_{c_i} - r_{c_i})}{n})/\max_{1\leq i\leq n}(d_{c_i} - r_{c_i})] \times 100 \tag{5.11}$$

$$perc_{III} = [(\frac{\sum_{i=1}^{n}(q_{c_i})}{n})/\max_{1\leq i\leq n}(q_{c_i})] \times 100 \tag{5.12}$$

$$THRESHOLD = perc_I + perc_{II} + perc_{III}. \tag{5.13}$$

The SI1-Like approach captures under-constrained tours using two of criteria. The first criterion is to locate those tours with a percentage value of under-constrained customers that is greater than or equal to 50%. Once the first criterion is satisfied, the second criterion comes into stage to select the tour with minimum visited customers out of the set of under-constrained tours selected. Then, the customers of the final selected tour will be removed and exposed to the insertion procedure and the HLS described in Section 5.3.4 in order to re-insert those customers into the other tours of a solution.

5.3.4 Modified versions of the insertion procedure and the hybrid local search HLS

Once an under-constrained tour is captured, its customers are tried for insertion into the other tours of a solution using a modified version of the insertion procedure in Section 4.4.7 and later an updated version of the hybrid local search HLS in Section 4.4.10.

In the insertion procedure used in SI1-Like and before doing any insertion into the other tours of a solution, the customers of an under-constrained tour are sorted in a descending order as well but according to how much constrained they are rather than to how much big their demand quantities are as in Section 4.4.7. Each customer is constrained according to how far the customer is from the depot, how much the demand of the customer is large and how much the time window of the customer is narrow.

Moreover, the hybrid local search HLS, used in SI1-Like, works in the same way described in Section 4.4.10. But here the HLS sorts at the beginning the tours

of a solution in an ascending order from the least to the largest tour size and it starts improving from the tour with the least size. Moreover, it uses the insertion procedure described in the previous paragraph and it has the ability to improve the quality of not just the infeasible solutions but also the feasible ones. In addition, it uses two extra intra-route moves as in Figures 5.4 and 5.5, called inversions of type 1 and 2, which invert each a segment of visited customers in a tour.

The difference between the two intra-route improvement operators does depend only on where a visited customer c_i and his nearest customer c_j are located in a route and whether the nearest customer c_j is located before or after the visited customer c_i. If the nearest customer c_j is located after the visited customer c_i, then the inversion operator is of type 1 and works as in Figure 5.4. On the other hand if the nearest customer c_j is located before the customer c_i, then the inversion operator becomes of type 2 and works as in Figure 5.5.

Figure 5.4: Inversion Operator - Type 1.

Figure 5.5: Inversion Operator - Type 2.

5.3.5 An 'eject and insert' strategy

In the case of not removing the under-constrained tour t_1 selected for removal purposes, an 'eject and insert' strategy is used as in Figure 5.7 to eject already visited customers from their locations in their tours to other locations in other tours and this is in order to hopefully have the chance to insert all or at least some of the already unvisited customers of the selected under-constrained tour t_1.

In order to ease what is described earlier, the 'eject and insert' strategy uses, at first, spatial or temporal regions (such as sectors, tracks and major time intervals) in which the customers to be ejected and inserted might relate to or share them together. For instance, knowing that two customers share some region might help in learning how to make rooms in routes for those under-constrained customers to be inserted.

For that, regions such as sectors, tracks and major time intervals of a problem instance are created by what are called knowledge-base algorithms before doing any computation or optimization in the SI1-Like deterministic approach. The knowledge-base algorithms are meant by to create knowledge-base data structures of a problem instance to be used possibly later within multiple ant colony systems and deterministic approaches. Those knowledge-base data structures are built using various statistical methods like average, standard deviation, frequency and percentage. The knowledge-base algorithms are:

K1- An algorithm to create 8 sectors as in Figure 5.6.

K2- An algorithm to create 10 tracks as in Figure 5.6.

K3- An algorithm to create 10 major time intervals. Behold that each customer might be located temporally in one or more major time intervals.

K4- An algorithm to know the theta angles of all nodes including the depot from the point view of a node.

K4- An algorithm to know how much each customer is being constrained. Of course, the constrained-ness of a customer depends on three features, which are how far the customer is from the depot, how tight the time window of the customer is and how large the demand of the customer is.

Then as in Figure 5.7, the 'eject and insert' strategy is run in the outer-for loop for a number of iterations equal to the size of the under-constrained tour t_1 chosen for deletion. Then in each of the iterations allocated, the unvisited customer c_i that is constrained the most, among other under-constrained customers, is picked. Later, the unvisited customer c_i, selected earlier, is tabooed in the tabu list number 1 in order not to be used in later iterations. Once the unvisited customer c_i is tabooed in the tabu list number 1, a list of visited customers that belong to the same region (i.e. sector, track or major time intervals) of the unvisited customer c_i is created.

Afterwards for a number of iterations in the inner-for loop equal to the size of the list created so far, the visited customer c_e that is constrained the least is picked for ejection from its location in a tour into a location in another tour. Also, the visited customer c_e, picked for ejection, is tabooed in the tabu list number 2 in order not to be chosen for ejection in the next iterations. Now, if the visited customer c_e selected for ejection is inserted feasibly into another route, then the insertion procedure in Section 5.3.4 is applied in an effort to insert the unvisited customers of the under-constrained tour t_1.

5.4 What the maximization function of SI1-Like can do?

As explained in Section 5.3.2, the routing builder of the approach SI1-Like and Solomon's I1 heuristic [1] are different from each other in the sense that I1 finds a minimum insertion place for every unrouted customer before maximizing the value in Equation 2.26 whereas the routing builder of SI1-Like does not care about the usage of the minimization component.

In association to other VRPTW algorithms, like I1 [1] and I1-AD [14], in the literature as in Table 5.1, this section discusses the effect of using the maximization function in Figure 5.3 that is used in the first deterministic approach called SI1-Like 01. The SI1-Like 01 approach uses only the routing builder in Section 5.3.2 among all those components mentioned in Section 5.3.1 and maximizes a value, which is calculated for every unrouted customer from Equations 2.23, 2.24, 2.25 and 2.26. In SI1-like 01, there is no use also for the following components.

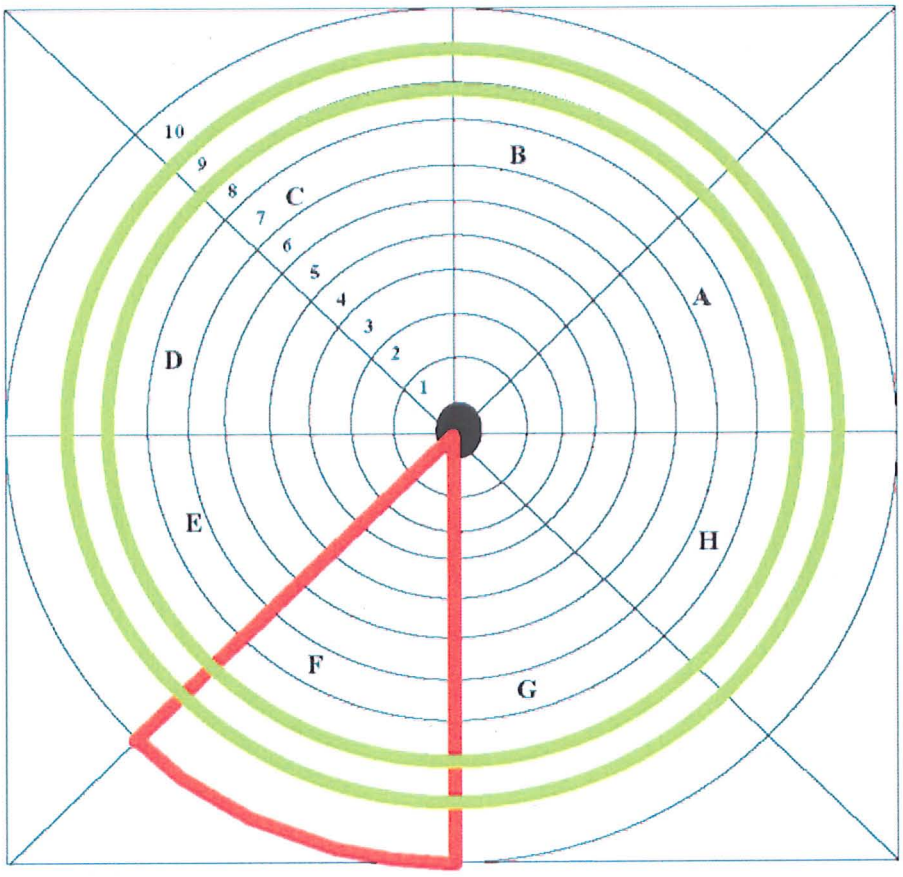


Figure 5.6: Sectors and tracks of a problem instance.

Procedure Eject-And-Insert()
Begin

(a) Region = Sector, Track or Time Interval

for (a number of iterations equal to the size of an under-constrained tour t_1 minus the depot) do

(b) Pick the under-constrained customer c_i that is constrained the most. Update the tabu list #1 with the unvisited customer c_i picked so far in order not to be selected again in the next iteration.

if (Region == Sector or Track) then
(c) Bring the sector or the track number of where the under-constrained customer c_i is located in the graph.
else-if (Region == Time Interval)
(d) Bring the start and the end of the time interval of where the under-constrained customer c_i is located in time terms.
od if-else

(e) Bring the tour ID #1 of the under-constrained customer c_i.
(f) Create a list of visited customers that are members of the sector number, track number or time interval (selected in the above if-else statement) and belong to tours other than the tour ID #1.

for (a number of iterations equal to the size of the list created in the previous step) do
(g) Pick the least constrained customer c_e of the list of visited customers (created in step f) in order to be ejected. Update the tabu list #2 with the visited customer c_e in order not to be picked again for ejection in the next iteration.
(h) Bring the tour ID #2 of the visited customer c_e to be ejected.
(i) Try to insert the visited customer c_e to be ejected into one of the tours other than the tour ID #2.
(j) If the visited customer c_e to be ejected is inserted into another tour without causing any infeasibility, then apply the insertion procedure to insert the unvisited customers of the under-constrained tour t_1, selected already for removal purposes. In the insertion procedure, the under-constrained customers are ordered in a descending order according to how much constrained they are.
od for

od for

End

Figure 5.7: The eject and insert procedure.

A1- The varying of the parametric values of a_1 and a_2. As an alternative, a_1 and a_2 either becomes equal to 0 or 1. If a_1 is equal to 0 (or 1), then a_2 becomes 1 (or 0).

A2- The "No Seeding" strategy.

A3- The waiting time functions talked about in Equations 5.3, 5.4 and 5.5. Instead, the servicing functions in Equation 2.25 are used as a part of the calculations to Equations 2.23 and 2.24 and 2.26.

A4- The removing heuristic of under-constrained tours.

A5- The modified versions of the insertion procedure and the hybrid local search.

A6- The 'eject and insert' strategy.

As in Solomon's I1 heuristic [1], SI1-Like 01 creates eight solutions for each problem instance using the two seeding strategies (of the farthest in distance and the earliest deadline) and four different sets of values of the parameters μ, λ, a_1 and a_2 - (1, 1, 1, 0), (1, 2, 1, 0), (1, 1, 0, 1) and (1, 2, 0, 1). So, two seeding strategies multiplied by four sets of parametric values equals eight solutions and the best solution created from the eight combinations is always picked according to the criteria mentioned at the end of Section 5.3.1.

After testing SI1-Like 01 on the problem group PG100 in Section 2.2 according to the experimental methodology described in Section 5.2, it can be seen as in Table 5.2 that the performance and results of the SI1-Like 01 is worse, on average by 6.38% for NV and 11.31% for TD, than those of the deterministic VRPTW algorithms in Table 5.1. For instance on the problem sets R1, C1, R2, C2 and RC2, SI1-Like 01 is beaten in terms of NV, on average with 3.23%, by the heuristic I1 [1]. Nonetheless, SI1-Like 01 manages, on RC1, to outperform I1 by -1.85% for NV.

It can be concluded, from the percentage of deviations to I1, on the six problem sets of PG100 that the maximization function is not so bad after all and this fact can be realized also if SI1-Like 01 is compared to the heuristic I1-AD, which has a lot of similarities with I1. In Table 5.2, the heuristic I1-AD [14] is better than SI1-Like 01 in terms of NV with average equal to 3.58% on the problem sets C1, R2 and C2. However on the problem sets R1, RC1 and RC2, SI1-Like is better, by

-0.81% for NV and -1.80% for TD, than I1-AD on average. Now in comparison to the deterministic approach RVNSa [5], it can be seen that SI1-Like 01 is inferior on PG100 and its results are worse on average by 14.95% and 24.91% for NV and TD respectively.

Furthermore, the algorithms in Table 4.3, which has many non-deterministic approaches, show on the problem group PG100 that they are on average significantly superior, with 9.86% for NV and 24.46% for TD, when compared to SI1-Like 01. For example, the non-deterministic approaches LS [28] and LS+TA [28], run on PG100 for 126 and 156 seconds in that order, can bring on average much enhanced results with 14.61% for NV and 24.27% for TD, which make SI1-Like 01 look so bad in terms of performance.

Also in contrast to the DACS systems that use the pheromone ants and on PG100 as in Table 5.2, SI1-Like 01 is poorer on average, by 6.46% for NV and 16.91% for TD, during 100 seconds. In particular, it is far worse in terms of performance and on average, by 7.19% for NV and 21.35% for TD, from those of the systems DACS 02, DACS 03, DACS+HLS and DACS+HLS+2-Opt. However in relation to the system DACS 01 on the problem sets R1 and RC1 only, it manages interestingly, on average by -3.00% and -4.68% respectively, to get better NV and TD.

5.5 What if an extra "No Seeding" strategy is added?

In this section, an extra "No Seeding" strategy is merged as in the SI1-Like 02 deterministic approach, which works exactly as the SI1-Like 01 approach in Section 5.4 but with a new strategy in addition to the other two seeding strategies of the farthest in distance and the earliest deadline. So, the idea of the "No Seeding" strategy is to have no seeding at all and to start the insertion of the unvisited customers into a currently used route without any need beforehand to seed that route with a seed customer.

The number of solutions should be created, in SI1-Like 02, for each problem instance is equal to 12, which is the result of multiplying the four different sets of parametric values (of μ, λ, a_1 and a_2 as mentioned in Section 5.4) by the three seeding strategies described above. In this case for each problem instance, SI1-Like

Table 5.2: Comparison between the average case performances of SI1-Like 01 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	0 - 4.91	20.00	1825.93	10.00	923.71	17.00	1826.83	4.00	1793.61	3.00	603.88	5.00	1976.18
% to RVNSa [5]	2220	5.26	10.51	0.00	11.43	21.43	7.54	0.00	42.25	0.00	2.08	25.00	38.38
02 - AVGs	0 - 4.91	18.00	1864.34	10.00	1029.44	14.00	1827.81	4.00	1572.93	4.00	795.61	4.00	1811.75
% to RVNSa [5]	2220	5.88	25.38	0.00	24.19	16.67	15.70	33.33	31.23	33.33	34.49	33.33	31.72
03 - AVGs	0 - 4.91	15.00	1560.55	11.00	1035.92	12.00	1545.31	4.00	1477.96	4.00	740.60	4.00	1534.89
% to RVNSa [5]	2220	15.38	19.03	10.00	25.10	9.09	20.71	33.33	54.43	33.33	25.28	33.33	44.46
04 - AVGs	0 - 4.91	12.00	1248.20	10.00	1147.90	11.00	1350.23	3.00	1110.09	4.00	948.61	3.00	1389.83
% to RVNSa [5]	2220	20.00	24.87	0.00	39.18	10.00	18.04	50.00	24.08	33.33	59.68	0.00	70.99
05 - AVGs	0 - 4.91	15.00	1598.40	10.00	878.78	16.00	1842.61	3.00	1436.99	3.00	685.01	5.00	1889.84
% to RVNSa [5]	2220	7.14	15.70	0.00	6.01	23.08	12.88	0.00	39.11	0.00	16.32	25.00	42.43
06 - AVGs	0 - 4.91	13.00	1559.99	10.00	898.40	13.00	1603.42	3.00	1266.40	3.00	663.19	4.00	1663.81
% to RVNSa [5]	2220	8.33	23.56	0.00	8.38	18.18	11.96	0.00	36.23	0.00	12.69	33.33	38.94
07 - AVGs	0 - 4.91	12.00	1339.41	10.00	903.91	12.00	1517.11	3.00	1137.94	3.00	686.98	4.00	1569.26
% to RVNSa [5]	2220	20.00	15.94	0.00	9.04	9.09	22.91	50.00	15.88	0.00	16.78	33.33	46.46
08 - AVGs	0 - 4.91	11.00	1191.70	10.00	950.09	11.00	1371.76	3.00	930.37	3.00	689.21	3.00	1158.86
% to RVNSa [5]	2220	22.22	22.24	0.00	14.62	10.00	19.00	50.00	26.21	0.00	17.15	0.00	35.71
09 - AVGs	0 - 4.91	14.00	1412.79	10.00	941.70	-	-	3.00	1428.80	-	-	-	-
% to RVNSa [5]	2220	27.27	14.07	0.00	13.60	-	-	0.00	51.21	-	-	-	-
10 - AVGs	0 - 4.91	12.00	1376.88	-	-	-	-	4.00	1336.50	-	-	-	-
% to RVNSa [5]	2220	20.00	21.59	-	-	-	-	33.33	37.04	-	-	-	-
11 - AVGs	0 - 4.91	12.00	1309.04	-	-	-	-	3.00	1017.42	-	-	-	-
% to RVNSa [5]	2220	20.00	14.88	-	-	-	-	50.00	4.61	-	-	-	-
12 - AVGs	0 - 4.91	11.00	1180.70	-	-	-	-	-	-	-	-	-	-
% to RVNSa [5]	2220	22.22	15.82	-	-	-	-	-	-	-	-	-	-
SI1-Like 01 - AVGs	0 - 4.91	13.75	1455.66	10.11	967.76	13.25	1610.64	3.36	1319.00	3.38	726.64	4.00	1624.30
Time(secs.)	0 - 4.91	0.08	-	0.44	-	0.00	-	4.91	-	3.50	-	4.00	-
% to RVNSa [5]	2220	14.58	18.40	1.11	16.83	15.22	15.52	23.33	33.29	12.50	23.10	23.08	42.35
% to I1 [1]	24 - 63	1.25	1.32	1.11	1.67	-1.85	0.89	2.86	-5.95	7.83	4.90	3.09	-3.44
% to I1-AD [14]	8 - 40	-0.58	-1.81	1.11	0.72	-1.85	-0.01	5.77	-2.67	3.85	-1.93	0.00	-3.57
% to DACS 01	300 - 400	-2.37	-4.41	1.11	-0.93	-3.64	-4.96	5.71	-2.14	12.50	4.73	7.87	2.60
% to DACS 02	100	2.27	12.89	1.11	12.46	0.32	10.71	2.78	20.53	2.53	8.68	4.35	22.11
% to DACS 03	100	6.09	17.11	1.11	15.92	5.05	13.96	6.42	33.04	12.50	22.65	11.24	38.88
% to DACS+HLS	100	8.24	16.63	1.11	15.45	7.58	13.69	10.56	31.41	12.50	22.34	18.52	35.96
% to DACS+HLS+2-Opt	100	8.06	18.01	1.11	16.56	8.02	15.00	10.01	35.23	12.50	22.83	18.67	40.33
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
I1 [1] - AVGs	24 - 63	13.58	1436.70	10.00	951.90	13.50	1596.50	3.27	1402.40	3.13	692.70	3.88	1682.10
I1-AD [14] - AVGs	8 - 40	13.83	1482.53	10.00	960.81	13.50	1610.78	3.18	1355.24	3.25	740.93	4.00	1684.43
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	100	13.44	1289.43	10.00	860.55	13.21	1454.84	3.27	1094.37	3.29	668.60	3.83	1330.21
DACS 03 - AVGs	100	12.96	1242.95	10.00	834.85	12.61	1413.30	3.16	991.42	3.00	592.44	3.60	1169.58
DACS+HLS - AVGs	100	12.70	1248.05	10.00	836.26	12.32	1416.72	3.04	1003.71	3.00	593.97	3.38	1194.72
DACS+HLS+2-Opt - AVGs	100	12.73	1233.54	10.00	830.28	12.27	1400.50	3.06	975.38	3.00	591.59	3.37	1157.51

02 chooses the best solution out of twelve solutions created from twelve combinations of the three seeding strategies and four different sets of parametric values.

In SI1-Like 02 and according the methodology of experimentation in Section 5.2, adding the "No Seeding" strategy has led, as in Table 5.3, into improving the performance and results significantly, by -2.49% for NV and -0.76% for TD on the problem group PG100 in Section 2.2, when compared on average with those of SI1-Like 01.

Of course in contrast to the percentage of deviations of SI1-Like 01 (6.38% for NV and 11.31% for TD) from the deterministic approaches in Table 5.1 on PG100, the addition of the "No Seeding" strategy has made SI1-Like 02 get closer on average to such approaches, with 3.68% for NV and 10.44% for TD, in terms of performance and results. For that on average and with a percentage of deviation equal to 0.99% for NV, SI1-Like 02 is getting a lot nearer to the heuristic I1 [1] on the problem sets R1, C1, R2 and C2. Also, it manages on average to beat I1 on the problem sets RC1 and RC2 by -2.60% for NV.

In comparison to the heuristic I1-AD [14], SI1-Like 02 is now on average outperforming it on the problem sets R1, RC1, C2 and RC2 by -2.47% for NV and -2.73% for TD, which are much better than those of SI1-Like 01. Likewise on average, it is a lot better by 1.46% and -1.42% for NV and TD respectively on the problem sets C1 and R2. Furthermore with percentages of deviations equal to 12.00% for NV and 23.92% for TD, SI1-Like 02 is nearer on average to RVNSa [5] and its performance now on the whole PG100 is in a way better than SI1-Like 01's performance.

Additionally, the "No Seeding" strategy has made the approach SI1-Like 02 become in a way much enhanced, on average, than the SI1-Like 01 approach when their performances are compared together with those of the VRPTW algorithms in Table 4.3 and this enhancement shows that SI1-Like 02 is deviated now, from such algorithms, with 7.06% and 23.50% for NV and TD correspondingly on the problem group PG100. For that if the performances of SI1-Like 02 and SI1-Like 01 are compared with the performances of LS [28] and LS+TA [28] on PG100, the percentages 11.66% and 23.29% of deviations of SI1-Like 02, on average for NV and TD, are much better.

In a way different from SI1-Like 01 during 100 seconds, the approach SI1-Like 02 is much nearer also in terms of performance to the average case performances of

the DACS systems that use the pheromone ants and this is on average, by 3.73% for NV and 15.99% for TD as in Table 5.3, and this improvement on PG100 is mirrored too in relation to the best and worst case scenarios of such systems. For instance, SI1-Like 02 is able on average to overcome DACS 01, by −2.40% for NV and −3.48% for TD, on the problem sets R1, C1 and RC1. As well, it is able to defeat DACS 02 on the problem sets C2 and RC2 in terms of NV with −1.72% and it is getting nearer on average to the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt by 6.05% and 22.63% for NV and TD also on PG100.

As result of what is described above, adding the "No Seeding" strategy to the SI1-Like approach is better than loosing it and therefore this strategy is kept as it is for its significance.

5.6 What about varying the parametric values of a_1 and a_2?

Thereafter as in the deterministic approach SI1-Like 03, the varying of the values, can be used in the parameters a_1 and a_2 between 0 and 1 (inclusive), is added as it is described in Figure 5.2 and Section 5.3.1. The SI1-Like 03 approach works as in the approach SI1-Like 02 in Section 5.5 but with the varying process mentioned earlier. The purpose behind this varying is to discover what sort of neighborhood solutions that could be gained between the two extreme values 0 and 1. After the varying process, it is believed that 66 solutions are going to be created and those solutions are the result of multiplying 6 iterations of the outer-for loop in Figure 5.2 by 11 steps of 0.1 between the values 0 and 1 of a_1 and a_2.

As in Table 5.4, the result of the varying process, according to the experimental methodology mentioned in Section 5.2, is a significant improvement, on average by −2.97% for NV and −2.59% for TD, on all the six problem sets of the problem group PG100 in Section 2.2 and this can be realized if SI1-Like 03 and SI1-Like 02 are compared together.

The significant improvement, on PG100, has led into enhancing the performance and results on average also in relation to the other deterministic and non-deterministic algorithms in the literature. SI1-Like 03 is now less departed on average in terms of NV and TD respectively, by 0.57% and 7.57%, from the deterministic

Table 5.3: Comparison between the average case performances of SI1-Like 02 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1 - 7.91	20.00	1825.93	10.00	878.36	17.00	1826.83	4.00	1793.61	3.00	591.56	4.00	1976.01
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	-4.91	0.00	0.00	0.00	0.00	0.00	-2.04	-20.00	-0.01
02 - AVGs	1 - 7.91	18.00	1864.34	10.00	984.35	14.00	1822.75	4.00	1572.93	3.00	731.49	4.00	1811.75
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	-4.38	0.00	-0.28	0.00	0.00	-25.00	-8.06	0.00	0.00
03 - AVGs	1 - 7.91	15.00	1560.55	10.00	1097.78	12.00	1545.31	4.00	1397.73	4.00	740.60	4.00	1534.89
% to SI1-Like 01	0 - 4.91	0.00	0.00	-9.09	5.97	0.00	0.00	0.00	-5.43	0.00	0.00	0.00	0.00
04 - AVGs	1 - 7.91	11.00	1258.95	10.00	1147.90	11.00	1350.23	3.00	1110.09	4.00	948.61	3.00	1216.88
% to SI1-Like 01	0 - 4.91	-8.33	0.86	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-12.44
05 - AVGs	1 - 7.91	15.00	1598.40	10.00	878.78	16.00	1842.61	3.00	1436.99	3.00	606.28	4.00	1971.66
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-11.49	-20.00	4.33
06 - AVGs	1 - 7.91	13.00	1559.99	10.00	898.40	13.00	1603.42	3.00	1266.40	3.00	663.19	4.00	1596.29
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-4.06
07 - AVGs	1 - 7.91	12.00	1339.41	10.00	903.91	12.00	1517.11	3.00	1137.94	3.00	686.98	4.00	1569.26
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
08 - AVGs	1 - 7.91	11.00	1191.70	10.00	950.09	11.00	1371.76	3.00	930.37	3.00	689.21	3.00	1158.86
% to SI1-Like 01	0 - 4.91	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
09 - AVGs	1 - 7.91	13.00	1443.67	10.00	941.70	-	-	3.00	1356.80	-	-	-	-
% to SI1-Like 01	0 - 4.91	-7.14	2.19	0.00	0.00	-	-	0.00	-5.04	-	-	-	-
10 - AVGs	1 - 7.91	12.00	1376.88	-	-	-	-	3.00	1406.77	-	-	-	-
% to SI1-Like 01	0 - 4.91	0.00	0.00	-	-	-	-	-25.00	5.26	-	-	-	-
11 - AVGs	1 - 7.91	12.00	1309.04	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 01	0 - 4.91	0.00	0.00	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	1 - 7.91	11.00	1180.70	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 01	0 - 4.91	0.00	0.00	-	-	-	-	-	-	-	-	-	-
SI1-Like 02 - AVGs	1 - 7.91	13.58	1459.13	10.00	964.59	13.25	1610.00	3.27	1311.55	3.25	707.24	3.75	1604.45
Time(secs.)	1 - 7.91	1.00	-	1.00	-	1.00	-	7.91	-	5.12	-	6.25	-
% to SI1-Like 01	0 - 4.91	-1.21	0.24	-1.10	-0.33	0.00	-0.04	-2.70	-0.56	-3.70	-2.67	-6.25	-1.22
% to RVNSa [5]	2220	13.19	18.68	0.00	16.44	15.22	15.47	20.00	32.53	8.33	19.81	15.38	40.61
% to I1 [1]	24 - 63	0.02	1.56	0.00	1.33	-1.85	0.85	0.08	-6.48	3.83	2.10	-3.35	-4.62
% to I1-AD [14]	8 - 40	-1.78	-1.58	0.00	0.39	-1.85	-0.05	2.92	-3.22	0.00	-4.55	-6.25	-4.75
% to DACS 01	300 - 400	-3.55	-4.18	0.00	-1.26	-3.64	-5.00	2.86	-2.69	8.33	1.94	1.12	1.35
% to DACS 02	100	1.03	13.16	0.00	12.09	0.32	10.67	0.00	19.84	-1.27	5.78	-2.17	20.62
% to DACS 03	100	4.80	17.39	0.00	15.54	5.05	13.92	3.55	32.29	8.33	19.38	4.29	37.18
% to DACS+HLS	100	6.93	16.91	0.00	15.07	7.58	13.64	7.57	30.67	8.33	19.07	11.11	34.30
% to DACS+HLS+2-Opt	100	6.75	18.29	0.00	16.18	8.02	14.96	7.04	34.47	8.33	19.55	11.25	38.61
SI1-Like 01 - AVGs	0 - 4.91	13.75	1455.66	10.11	967.76	13.25	1610.64	3.36	1319.00	3.38	726.64	4.00	1624.30
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
I1 [1] - AVGs	24 - 63	13.58	1436.70	10.00	951.90	13.50	1596.50	3.27	1402.40	3.13	692.70	3.88	1682.10
I1-AD [14] - AVGs	8 - 40	13.83	1482.53	10.00	960.81	13.50	1610.78	3.18	1355.24	3.25	740.93	4.00	1684.43
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	100	13.44	1289.43	10.00	860.55	13.21	1454.84	3.27	1094.37	3.29	668.60	3.83	1330.21
DACS 03 - AVGs	100	12.96	1242.95	10.00	834.85	12.61	1413.30	3.16	991.42	3.00	592.44	3.60	1169.58
DACS+HLS - AVGs	100	12.70	1248.05	10.00	838.26	12.32	1416.72	3.04	1003.71	3.00	593.97	3.38	1194.72
DACS+HLS+2-Opt - AVGs	100	12.73	1233.54	10.00	830.28	12.27	1400.50	3.06	975.38	3.00	591.59	3.37	1157.51

algorithms in Table 5.1 and by 3.86% and 20.26% from the VRPTW algorithms in Table 4.3.

For instance on PG100, the algorithms LS [28] and LS+TA [28] are able on average to outperform SI1-Like 03 by 8.32% for NV and 20.06% for TD, which are as percentages of deviations better than those of SI1-Like 02. Also as a result of the varying process, SI1-Like 03 is getting closer on average to the algorithm RVNSa [5] as in Table 5.4 and its performance is deviated now by 8.66% for NV and 20.68% for TD. Moreover as never seen before on all the six problem sets of PG100, it is able to get performance and results that are better, on average by -3.65% for NV and -4.12% for TD, than those of the heuristics I1 [1] and I1-AD [14].

Furthermore as in Table 5.4, SI1-Like 03 is now beating on average, with its varying component and by -4.16% and -12.82% for NV and TD respectively, the deterministic approach PR1 [11] on the problem sets R1, C1, RC1, C2 and RC2 and this is with the exception of R2, since PR1 is better on average by 2.97% on R2 in terms of NV. Additionally, it can be seen from Table 5.4 that it has become competitive in terms of NV and therefore better, on average by -7.65% for NV, than the deterministic approaches PS [79] and PR2 [78] when it comes to trying to solve the problem sets of R1, RC1, R2, C2 and RC2.

In relation to the average case scenarios of the DACS systems that use the pheromone ants and during 100 seconds as in Table 5.4, the varying process has made SI1-Like 03 on average, by 0.61% for NV and 12.96% for TD, to be in a closer position in terms of performance and results on all the six problem sets of the problem group PG100. What's more, the SI1-Like performance has advanced closely too to those of the best and worst case scenarios of such DACS systems.

As a result and for the first time, SI1-Like 03 is able, during 100 seconds, to succeed in defeating DACS 01 on all the six problem sets and on average by -2.22% and -4.19% for NV and TD correspondingly. Moreover on the problem sets R1, RC1, R2, C2 and RC2, it is able to overcome DACS 02 on average by -3.95% in terms of NV. When it comes to the systems DACS 03, DACS+HLS, DACS+HLS+2-Opt, the varying process has made the performance and results with percentages of deviations equal on average to 2.86% for NV and 19.42% for TD, which are far better than those achieved by SI1-Like 02.

Overall, the varying of the values of the parameters a_1 and a_2 has made SI1-Like

03 get better and significant performance and results and an approach without it will lead definitely into worsening the situation reached so far.

5.7 What if a removing heuristic of "under constrained" tours is considered?

Once the removing heuristic of "under-constrained" tours is added, as in the deterministic approach SI1-Like 04 in this section, the significant improvement in terms of the performance and results will be recognized instantly. The removing heuristic of under-constrained tours, which relocates the under-constrained customers of such tours (using an insertion procedure and a hybrid local search HLS) into other tours of the same solution as explained in Section 5.3.3, is the only thing that distinguishes SI1-Like 04 from its predecessor SI1-Like 03, talked about in Section 5.6.

With the removing heuristic described earlier and according to the experimental methodology in Section 5.2 on the problem group PG100 in Section 2.2, SI1-Like 04 outperforms SI1-Like 03 as in Table 5.5, on average, with -2.42% for NV and -7.49% for TD on all the six problem sets of Solomon [1]. This kind of improvement is reflected also on the relationship now between the approach that uses such removing heuristic and other non-deterministic and deterministic algorithms in the literature as stated in Tables 4.3 and 5.1.

For that on average, the percentages of deviation 1.28% and 11.01%, for NV and TD respectively from the algorithms in Table 4.3, indicate that SI1-Like 04 with its removing heuristic of under-constrained tours is much closer on the problem group PG100 to such algorithms in terms of performance than that of SI1-Like 03, which has its percentages of deviation equal to 3.86% for NV and 20.26% for TD from the same algorithms. For instance, the performance of an approach with the removing heuristic described earlier; in comparison to that of an approach without that particular heuristic and against the performances of the algorithms LS [28] and LS+TA [28], is very much so less deviated on PG100 and as a result the new percentages of deviation, on average, to such algorithms is now 5.60% for NV and 10.84% for TD.

Also on the problem group PG100, if SI1-Like 04 with its removing heuristic is compared to the deterministic algorithms in Table 5.1, the percentages of deviations

Table 5.4: Comparison between the average case performances of SI1-Like 03 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	5.88 - 46	20.00	1825.93	10.00	878.36	16.00	1803.80	4.00	1751.55	3.00	591.56	4.00	1926.97
% to SI1-Like 02	1 - 7.91	0.00	0.00	0.00	0.00	-5.88	-1.26	0.00	-2.34	0.00	0.00	0.00	-2.48
02 - AVGs	5.88 - 46	18.00	1864.34	10.00	978.30	14.00	1701.87	4.00	1488.07	3.00	713.83	4.00	1642.33
% to SI1-Like 02	1 - 7.91	0.00	0.00	0.00	-0.61	0.00	-6.63	0.00	-5.40	0.00	-2.41	0.00	-9.35
03 - AVGs	5.88 - 46	14.00	1642.49	10.00	1092.72	12.00	1545.31	3.00	1413.53	3.00	753.97	3.00	1572.29
% to SI1-Like 02	1 - 7.91	-6.67	5.25	0.00	-0.46	0.00	0.00	-25.00	1.13	-25.00	1.81	-25.00	2.44
04 - AVGs	5.88 - 46	11.00	1258.95	10.00	1134.71	11.00	1341.12	3.00	1027.94	3.00	920.99	3.00	1179.59
% to SI1-Like 02	1 - 7.91	0.00	0.00	0.00	-1.15	0.00	-0.67	0.00	-7.40	-25.00	-2.91	0.00	-3.06
05 - AVGs	5.88 - 46	14.00	1523.92	10.00	878.78	16.00	1744.58	3.00	1388.65	3.00	606.28	4.00	1882.80
% to SI1-Like 02	1 - 7.91	-6.67	-4.66	0.00	0.00	0.00	-5.32	0.00	-3.36	0.00	0.00	0.00	-4.51
06 - AVGs	5.88 - 46	13.00	1443.40	10.00	898.40	13.00	1542.89	3.00	1223.62	3.00	663.19	4.00	1519.09
% to SI1-Like 02	1 - 7.91	0.00	-7.47	0.00	0.00	0.00	-3.78	0.00	-3.38	0.00	0.00	0.00	-4.84
07 - AVGs	5.88 - 46	12.00	1315.21	10.00	903.91	12.00	1496.77	3.00	1137.94	3.00	633.27	4.00	1482.67
% to SI1-Like 02	1 - 7.91	0.00	-1.81	0.00	0.00	0.00	-1.34	0.00	0.00	0.00	-7.82	0.00	-5.52
08 - AVGs	5.88 - 46	10.00	1155.63	10.00	853.25	11.00	1331.62	3.00	930.37	3.00	634.38	3.00	1156.87
% to SI1-Like 02	1 - 7.91	-9.09	-3.03	0.00	-10.19	0.00	-2.93	0.00	0.00	0.00	-7.96	0.00	-0.17
09 - AVGs	5.88 - 46	13.00	1375.65	10.00	884.16	-	-	3.00	1297.73	-	-	-	-
% to SI1-Like 02	1 - 7.91	0.00	-4.71	0.00	-6.11	-	-	0.00	-4.35	-	-	-	-
10 - AVGs	5.88 - 46	12.00	1296.45	-	-	-	-	3.00	1389.47	-	-	-	-
% to SI1-Like 02	1 - 7.91	0.00	-5.84	-	-	-	-	0.00	-1.23	-	-	-	-
11 - AVGs	5.88 - 46	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 02	1 - 7.91	-8.33	-2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	5.88 - 46	10.00	1189.46	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 02	1 - 7.91	-9.09	0.74	-	-	-	-	-	-	-	-	-	-
SI1-Like 03 -AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
Time(secs.)	5.88 - 46	6.00	-	7.33	-	5.88	-	46.00	-	30.12	-	35.25	-
% to SI1-Like 02	1 - 7.91	-3.07	-1.93	0.00	-2.06	-0.94	-2.89	-2.78	-2.50	-7.69	-2.48	-3.33	-3.68
% to RVNSa [5]	2220	9.72	16.39	0.00	14.05	14.13	12.14	16.55	29.22	0.00	16.84	11.54	35.43
% to I1 [1]	24 - 63	-3.04	-0.39	0.00	-0.75	-2.78	-2.07	-2.70	-8.82	-4.15	-0.44	-6.57	-8.13
% to I1-AD [14]	8 - 40	-4.80	-3.47	0.00	-1.67	-2.78	-2.94	0.06	-5.64	-7.69	-6.92	-9.38	-8.26
% to DACS 01	300 - 400	-6.51	-6.03	0.00	-3.29	-4.55	-7.74	0.00	-5.12	0.00	-0.59	-2.25	-2.39
% to DACS 02	100	-2.07	10.98	0.00	9.78	-0.63	7.47	-2.78	16.85	-8.86	3.15	-5.43	16.17
% to DACS 03	100	1.59	15.13	0.00	13.16	4.06	10.63	0.67	28.98	0.00	16.41	0.81	32.13
% to DACS+HLS	100	3.65	14.66	0.00	12.70	6.56	10.36	4.58	27.40	0.00	16.11	7.41	29.35
% to DACS+HLS+2-Opt	100	3.47	16.01	0.00	13.79	7.00	11.64	4.06	31.10	0.00	16.58	7.54	33.50
SI1-Like 02 - AVGs	1 - 7.91	13.58	1459.13	10.00	964.59	13.25	1610.00	3.27	1311.55	3.25	707.24	3.75	1604.45
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
I1 [1] - AVGs	24 - 63	13.58	1436.70	10.00	951.90	13.50	1596.50	3.27	1402.40	3.13	692.70	3.88	1682.10
I1-AD [14] - AVGs	8 - 40	13.83	1482.53	10.00	960.81	13.50	1610.78	3.18	1355.24	3.25	740.93	4.00	1684.43
PR1 [11] - AVGs	1176	13.33	1509.04	10.67	1343.69	13.38	1723.72	3.09	1386.67	3.38	797.59	3.63	1651.05
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	100	13.44	1289.43	10.00	860.55	13.21	1454.84	3.27	1094.37	3.29	668.60	3.83	1330.21
DACS 03 - AVGs	100	12.96	1242.95	10.00	834.85	12.61	1413.30	3.16	991.42	3.00	592.44	3.60	1169.58
DACS+HLS - AVGs	100	12.70	1248.05	10.00	838.26	12.32	1416.72	3.04	1003.71	3.00	593.97	3.38	1194.72
DACS+HLS+2-Opt - AVGs	100	12.73	1233.54	10.00	830.28	12.27	1400.50	3.06	975.38	3.00	591.59	3.37	1157.51

-1.87% and -0.58% for NV and TD correspondingly shows that it is able to defeat on average many of such algorithms in terms of performance. For instance in addition to beating the algorithms I1 [1] and I1-AD [14] on PG100, the approach using the removing heuristic of under-constrained tours is able now, as never been before in any SI1-Like approach, to outperform the deterministic approaches PR1 [11] and TP [13] in terms of NV and TD together on all the six problem sets and this is on average by -3.89% and -12.03%.

Furthermore, the removing heuristic of under-constrained tours is now able to make the approach SI1-Like 04 get closer very much so to other deterministic algorithms like PR2 [78], AD [14], PS [79] and RVNSa [5]. On the problem sets R1, RC1, R2, C2 and RC2 only, an SI1-Like approach with that removing heuristic is better in terms of performance, on average by -5.90% for NV and -7.04% for TD, than PR2. In relation to AD, it manages in terms of performance also to defeat it, on average with -0.03% and -11.62% for NV and TD respectively, on the problem sets C1, R2, C2 and RC2. When it comes comparing an approach that uses the removing heuristic with the deterministic algorithm PS [79], the performance of such approach, like SI1-Like 04 for instance, is much better on average than that of PS by -12.11% for NV and it is much nearer to its performance in terms of TD by 10.96% on the entire six problem sets. As well on the same six problem sets and by 5.93% for NV and 11.40% for TD, the SI1-Like performance is much nearer on average to that of RVNSa [5].

As opposed to the performance of SI1-Like 03 on average and during 300 seconds, the addition of the removing heuristic of under-constrained tours has led as in Table 5.5 also into making the SI1-Like 04 performance on PG100 to be very close, by -0.80% and 5.72%, to the average case performances of the DACS systems that use the pheromone ants. Of course, such enhancement of the SI1-Like performance is repeated too once it is looked at in relation to the best and worst case scenarios of such DACS systems and this can be verified by checking what is going to happen if an SI1-Like approach is left without using that removing heuristic.

For that during 300 seconds and in contrast to the system DACS 01, the SI1-Like 04 approach is able, on average by -4.57% for NV and -11.39% for TD, to bring even far better results on PG100 than that of SI1-Like 03, which does not use such removing heuristic. In addition, SI1-Like 04 is able to defeat the system DACS 02

in terms of NV and by -3.40% on average instead of -0.51% and in a way that is far greater than that of SI1-Like 03 on the problem sets R1, RC1, R2, C2 and RC2. It is for the first time to see that an SI1-Like approach is capable of overcoming, on average by -1.84%, the system DACS 03 in terms of NV on the problem sets R2 and RC2 during 300 seconds. Moreover on PG100, the approach that uses the removing heuristic is closing in terms of performance, a lot on average by 1.71% for NV and 11.27% for TD, to the systems DACS+HLS and DACS+HLS+2-Opt.

Finally, it can be seen according to what is explained above that an SI1-Like approach without the removing heuristic of under-constrained tours will lead without doubt into worsening the performance and results.

5.8 Is the insertion procedure of more to less constrained customers doing any good?

The effects of changing, the working way of the insertion procedure in Section 4.4.7 on the performance and results of the SI1-Like approach and therefore the modified version of the hybrid local search HLS described in Section 4.4.10, are discovered in this section. Here, the way or the order, in which the insertion procedure re-inserts the visited customers of an under-constrained tour (chosen for removal) into the other tours of a solution, is modified as in the deterministic approach S1-Like 05, which works as in SI1-Like 04 but with the single modification mentioned earlier. So, the under-constrained customers of that tour are ordered in a descending order according to how much constrained they are as described in Section 5.3.4 rather than to how large their demand quantities are.

After the update of the working way of the insertion procedure as described above and testing it on the problem group PG100 in Section 2.2 according to the methodology of experimentation mentioned in Section 5.2, the SI1-Like 05 approach is found on average as in Table G.1 to bring the same number of vehicles on PG100 as in SI1-Like 04. Also, it is found on average to deteriorate the TD results by 1.35% on the problem sets C1, R2, C2 and RC2 and to improve them insignificantly by -0.12% on the problem sets R1 and RC1. However, the SI1-Like 05 approach is still outperforming SI1-Like 03 on PG100 by -2.42 and -6.72 for NV and TD respectively.

Table 5.5: Comparison between the average case performances of SI1-Like 04 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	15.78 - 137.64	20.00	1722.93	10.00	833.24	15.00	1754.91	4.00	1456.35	3.00	591.56	4.00	1597.21
% to SI1-Like 03	5.88 - 46	0.00	-5.64	0.00	-5.14	-6.25	-2.71	0.00	-16.85	0.00	0.00	0.00	-17.11
02 - AVGs	15.78 - 137.64	18.00	1579.59	10.00	968.25	14.00	1617.26	4.00	1336.32	3.00	591.56	4.00	1638.54
% to SI1-Like 03	5.88 - 46	0.00	-15.27	0.00	-1.03	0.00	-4.97	0.00	-10.20	0.00	-17.13	0.00	-0.23
03 - AVGs	15.78 - 137.64	14.00	1422.88	10.00	1005.73	12.00	1425.88	3.00	1090.19	3.00	715.18	3.00	1232.53
% to SI1-Like 03	5.88 - 46	0.00	-13.37	0.00	-7.96	0.00	-7.73	0.00	-22.87	0.00	-5.14	0.00	-21.61
04 - AVGs	15.78 - 137.64	10.00	1136.89	10.00	1134.71	11.00	1293.48	3.00	1027.94	3.00	816.30	3.00	1032.55
% to SI1-Like 03	5.88 - 46	-9.09	-9.70	0.00	0.00	0.00	-3.55	0.00	0.00	0.00	-11.37	0.00	-12.47
05 - AVGs	15.78 - 137.64	14.00	1439.41	10.00	832.27	15.00	1704.79	3.00	1199.94	3.00	606.28	4.00	1552.60
% to SI1-Like 03	5.88 - 46	0.00	-5.55	0.00	-5.29	-6.25	-2.28	0.00	-13.59	0.00	0.00	0.00	-17.54
06 - AVGs	15.78 - 137.64	13.00	1354.97	10.00	828.94	12.00	1521.72	3.00	1168.83	3.00	588.49	3.00	1335.46
% to SI1-Like 03	5.88 - 46	0.00	-6.13	0.00	-7.73	-7.69	-1.37	0.00	-4.48	0.00	-11.26	-25.00	-12.09
07 - AVGs	15.78 - 137.64	11.00	1392.23	10.00	828.94	12.00	1362.18	3.00	1137.94	3.00	606.06	3.00	1364.75
% to SI1-Like 03	5.88 - 46	-8.33	5.86	0.00	-8.29	0.00	-8.99	0.00	0.00	0.00	-4.30	-25.00	-7.95
08 - AVGs	15.78 - 137.64	10.00	1064.04	10.00	853.25	11.00	1254.11	2.00	909.27	3.00	615.08	3.00	1156.87
% to SI1-Like 03	5.88 - 46	0.00	-7.93	0.00	0.00	0.00	-5.82	-33.33	-2.27	0.00	-3.04	0.00	0.00
09 - AVGs	15.78 - 137.64	12.00	1281.02	10.00	884.16	-	-	3.00	1113.30	-	-	-	-
% to SI1-Like 03	5.88 - 46	-7.69	-6.88	0.00	0.00	-	-	0.00	-14.21	-	-	-	-
10 - AVGs	15.78 - 137.64	12.00	1230.79	-	-	-	-	3.00	1132.10	-	-	-	-
% to SI1-Like 03	5.88 - 46	0.00	-5.06	-	-	-	-	0.00	-18.52	-	-	-	-
11 - AVGs	15.78 - 137.64	11.00	1226.06	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 03	5.88 - 46	0.00	-4.28	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	15.78 - 137.64	10.00	1084.49	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 03	5.88 - 46	0.00	-8.83	-	-	-	-	-	-	-	-	-	-
SI1-Like 04 -AVGs	15.78 - 137.64	12.92	1327.94	10.00	907.72	12.75	1491.79	3.09	1144.51	3.00	641.31	3.38	1363.81
Time(secs.)	15.78 - 137.64	18.08	-	15.78	-	16.12	-	137.64	-	64.25	-	101.12	-
% to SI1-Like 03	5.88 - 46	-1.90	-7.20	0.00	-3.92	-2.86	-4.59	-2.86	-10.50	0.00	-7.01	-6.90	-11.75
% to RVNSa [5]	2220	7.64	8.01	0.00	9.58	10.87	7.00	13.33	15.65	0.00	8.64	3.85	19.52
% to PR1 [11]	1176	-3.10	-12.00	-6.28	-32.45	-4.71	-13.46	0.03	-17.46	-11.24	-19.59	-7.02	-17.40
% to TP [13]	108	-0.64	-2.14	0.00	-0.98	-1.92	-1.49	-2.80	-10.30	0.00	-0.51	-9.03	-16.56
% to DACS 01	300 - 400	-8.28	-12.80	0.00	-7.08	-7.27	-11.97	-2.86	-15.08	0.00	-7.57	-8.99	-13.85
% to DACS 02	300	-2.52	6.65	0.00	6.05	-0.97	4.89	-2.86	7.59	-2.70	4.97	-7.95	7.11
% to DACS 03	300	1.17	7.77	0.00	9.13	2.34	6.26	-1.26	18.56	0.00	8.30	-2.41	17.33
% to DACS+HLS	300	2.51	7.25	0.00	8.81	5.05	5.91	3.03	16.90	0.00	8.10	0.12	16.23
% to DACS+HLS+2-Opt	300	2.22	8.47	0.00	9.51	5.26	6.89	2.20	19.33	0.00	8.46	0.12	19.39
SI1-Like 03 -AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
PR1 [11] - AVGs	1176	13.33	1509.04	10.67	1343.69	13.38	1723.72	3.09	1386.67	3.38	797.59	3.63	1651.05
TP [13] - AVGs	108	13.00	1356.92	10.00	916.67	13.00	1514.29	3.18	1276	3.00	644.63	3.71	1634.43
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Despite the insignificant enhancement of SI1-Like 05 to the TD results on R1 and RC1 in comparison to SI1-Like 04, however the new way of ordering the under-constrained customers is significantly improving the TD results, on some particular problem instances like R104, R108, RC104 and RC205, on average by -1.91%. This discovery of significant improvement on the problem instances mentioned earlier has encouraged the author to do more experimentation with the current approach of SI1-Like 05 as indicated in the later sections of this chapter.

5.9 What if the quality of feasible solutions is improved as in the infeasible ones?

In Sections 5.7 and 5.8, the hybrid local search HLS; used in the deterministic approaches SI1-Like 04 and SI1-Like 05, concentrates only on inserting the removed customers of an under-constrained tour into the other tours of the infeasible solution involved and once such customers are inserted, the solution becomes feasible and the HLS stops working without trying to improve that feasible solution in terms of the total of traveled distances any further. As a result, the HLS in Sections 5.7 and 5.8 tries only to improve the infeasible solutions without any care about the solutions that are feasible. In this section, the approach SI1-Like 06, which extends SI1-Like 05, does not stop working once all the customers of an under-constrained tour are re-inserted and therefore it continues improving on the feasible solutions as well as the infeasible ones.

Later as in Table 5.6 and according to the experimental methodology in Section 5.2, the idea of improving the quality of feasible solutions in addition to the infeasible ones is tried, as in SI1-Like 06, on the problem group PG100 in Section 2.2. Such idea has shown that it is able to enhance significantly the traveled distances, on average by -6.85%, on all the six problem sets of PG100 in contrast to the TD results of SI1-Like 04 and SI1-Like 05. The reduction of the number of vehicles is the same on average in all the SI1-Like approaches mentioned in this paragraph.

The significant improvement in terms of the traveled distances on PG100 has made the performance of the SI1-Like approach get near or very closely to the performances of other non-deterministic and deterministic algorithms. In terms of performance, the percentage of deviation of the SI1-Like 06 approach, from the

algorithms in Table 4.3, is away by 3.74% for TD, which is much better on average than the percentages 11.98% and 11.01% of SI1-Like 05 and SI1-Like 04 respectively. As well in relation to the deterministic algorithms in Table 5.1, the percentage of deviation of SI1-Like 06 for TD is equal to -7.01% and this shows that the performance here is much enhanced if compared to those performances (of SI1-Like 05 and SI1-Like 04) with percentages of deviation equal to 0.26% and -0.58%.

For that in comparison to the algorithms LS [28] and LS+TA [28] in Table 4.3, the performance of SI1-Like 06, on the problem group PG100, is a lot better on average now but it is still, by 3.57%, worse and therefore departed in terms of TD from the performances of such algorithms. Even though the traveled distances are enhanced because of the idea of improving the quality of the feasible and the infeasible solutions, however the performances of the deterministic algorithms CL1 [77], AKRed [60], RVNSa [5] and RVNSb [5] in Table 5.1, on PG100, are still better on average, by 3.71% in terms of TD, than that of SI1-like 06.

Nonetheless as never been before, the approach SI1-Like 06 outperforms the deterministic algorithm PR2 [78] on average, by -11.77% for TD, on all the six problem sets of PG100 besides its outstanding performance and dominance by -4.92% in terms of NV. PR2 is the fifth deterministic algorithm to be outperformed by SI1-Like 06 and this is in addition to the deterministic algorithms I1 [1], I1-AD [14], PR1 [11] and TP [13]. Also when it comes to the deterministic heuristic AD, SI1-Like 06 does better on average than such heuristic in terms of TD, by -18.50%, on the problem sets C1, R2, C2 and RC2.

In relation to the average case scenarios of the DACS systems using the pheromone ants, the traveled distances of SI1-Like 06, during 300 seconds, have got better very much so as in Table 5.6 and closer on average by -1.18% (instead of 6.18%) if compared with those of SI1-Like 04 and SI1-Like 05 on PG100. However on average, the percentage -0.80% of deviation for NV to such average case scenarios of the DACS systems is the same in all the three SI1-Like approaches mentioned earlier. As soon as it comes to the best and worst scenarios of such DACS systems, the SI1-Like performance also, without any reduction for the number of vehicles, is advancing towards them in terms of the traveled distances even further.

On DACS 01 and in addition to overcoming it by -4.57% in terms of NV, the SI1-Like 06 approach manages to reduce the results on the problem group PG100,

further in terms of TD and on average by −17.10% instead of −11.02%, as in SI1-Like 05 and SI1-Like 04. During the first 100 seconds, it is able also to beat DACS 02 on average by −3.28% for TD on the problem sets R1, C1, RC1 and C2 and this is on top of what is achieved by −4.07% in terms of NV. Moreover on the problem sets R2 and RC2, it outperforms DACS 02 on average by −2.84% for TD as never been before and −5.41% for NV in the time interval between 100 and 300 seconds.

The SI1-Like 06 approach has the ability still, during the 300 seconds of CPU time, to defeat DACS 03 on the problem sets R2 and RC2 in terms of NV by −1.84% but the new thing here is that the traveled distances are further reduced as indicated from the percentage of deviation 6.76% compared to this 19.00% of SI1-Like 05 and SI1-Like 04. Generally speaking in terms of performance on the problem group PG100, it is closer on average to the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt by 3.95% and 1.13% for TD and NV respectively instead of 11.75% and 1.13% but it is still not as good as such systems.

As a conclusion to what is described, the idea of improving the quality of feasible solutions as well as the infeasible ones is better to use within the hybrid local search HLS and therefore leaving it out would mean that the performance of the SI1-Like approach is going to be downhill or very bad.

5.10 What if the order of the tours within HLS is changed?

The order of tours in a solution and whether its change has any effects is looked at in this section briefly. In a new deterministic approach called SI1-Like 07, its entire components behave in the same way as in SI1-Like 06 mentioned in Section 5.9 but the main difference here is that the order of the tours of the solution to be optimized using the HLS is sorted to be in an ascending order from the least to the largest tour size. For that, the time the HLS decides to improve the quality, it would start from the tour with the least size. The tours of a solution is sorted in this way before using the HLS in order to see the effects of the tour order on the performance of an SI1-Like approach.

Once the order of the tours (from the least to the largest tour size) is considered as described above, the SI1-Like 07 approach is tested on the problem group

Table 5.6: Comparison between the average case performances of SI1-Like 06 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	19.75 - 141.91	20.00	1696.09	10.00	828.94	15.00	1701.50	4.00	1338.10	3.00	591.56	4.00	1516.82
% to SI1-Like 05	15.22 - 114.55	0.00	-1.56	0.00	-0.52	0.00	-3.04	0.00	-8.12	0.00	0.00	0.00	-5.03
02 - AVGs	19.75 - 141.91	18.00	1565.48	10.00	896.84	14.00	1530.69	4.00	1171.56	3.00	591.56	4.00	1273.30
% to SI1-Like 05	15.22 - 114.55	0.00	-0.89	0.00	-8.33	0.00	-5.35	0.00	-18.73	0.00	0.00	0.00	-22.47
03 - AVGs	19.75 - 141.91	14.00	1297.87	10.00	916.16	12.00	1360.63	3.00	995.73	3.00	591.17	3.00	1183.24
% to SI1-Like 05	15.22 - 114.55	0.00	-8.79	0.00	-11.74	0.00	-4.58	0.00	-23.34	0.00	-19.89	0.00	-3.93
04 - AVGs	19.75 - 141.91	10.00	1082.73	10.00	949.62	11.00	1197.60	3.00	866.04	3.00	634.41	3.00	927.07
% to SI1-Like 05	15.22 - 114.55	0.00	-4.07	0.00	-16.31	0.00	-5.99	0.00	-15.75	0.00	-26.23	0.00	-21.41
05 - AVGs	19.75 - 141.91	14.00	1412.88	10.00	828.94	15.00	1639.13	3.00	1118.31	3.00	588.88	4.00	1400.45
% to SI1-Like 05	15.22 - 114.55	0.00	-1.84	0.00	-0.40	0.00	-3.85	0.00	-6.80	0.00	-2.87	0.00	-5.12
06 - AVGs	19.75 - 141.91	13.00	1329.41	10.00	828.94	12.00	1473.69	3.00	1024.76	3.00	588.49	3.00	1275.64
% to SI1-Like 05	15.22 - 114.55	0.00	-1.89	0.00	0.00	0.00	-3.16	0.00	-12.33	0.00	0.00	0.00	-4.48
07 - AVGs	19.75 - 141.91	11.00	1273.22	10.00	828.94	12.00	1334.79	3.00	1137.94	3.00	588.29	3.00	1243.06
% to SI1-Like 05	15.22 - 114.55	0.00	-8.55	0.00	0.00	0.00	-2.01	0.00	0.00	0.00	-2.93	0.00	-8.92
08 - AVGs	19.75 - 141.91	10.00	1029.94	10.00	828.94	11.00	1206.33	2.00	824.03	3.00	588.32	3.00	973.26
% to SI1-Like 05	15.22 - 114.55	0.00	-2.74	0.00	-2.85	0.00	-3.81	0.00	-10.40	0.00	-4.35	0.00	-15.87
09 - AVGs	19.75 - 141.91	12.00	1263.45	10.00	828.94	-	-	3.00	989.84	-	-	-	-
% to SI1-Like 05	15.22 - 114.55	0.00	-1.37	0.00	-6.25	-	-	0.00	-14.21	-	-	-	-
10 - AVGs	19.75 - 141.91	12.00	1179.64	-	-	-	-	3.00	1006.50	-	-	-	-
% to SI1-Like 05	15.22 - 114.55	0.00	-4.16	-	-	-	-	0.00	-11.09	-	-	-	-
11 - AVGs	19.75 - 141.91	11.00	1207.90	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 05	15.22 - 114.55	0.00	-1.48	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	19.75 - 141.91	10.00	1075.41	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 05	15.22 - 114.55	0.00	-0.84	-	-	-	-	-	-	-	-	-	-
SI1-Like 06 -AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
Time(secs.)	19.75 - 141.91	22.67	-	21.78	-	19.75	-	141.91	-	76.25	-	115.50	-
% to SI1-Like 05	15.22 - 114.55	0.00	-3.19	0.00	-5.79	0.00	-3.95	0.00	-11.30	0.00	-8.36	0.00	-10.84
% to SI1-Like 04	15.78 - 137.64	0.00	-3.27	0.00	-5.30	0.00	-4.11	0.00	-8.73	0.00	-7.17	0.00	-10.24
% to RVNSa [5]	2220	7.64	4.48	0.00	3.77	10.87	2.60	13.33	5.55	0.00	0.85	3.85	7.28
% to AD [14]	132 - 253	0.68	-7.35	0.00	-10.03	2.00	-7.46	0.03	-23.56	0.00	-17.00	-0.15	-23.40
% to PR2 [78]	180	-3.10	-7.05	0.00	-4.80	-3.77	-7.43	-5.48	-19.24	-4.15	-8.86	-13.02	-23.26
% to DACS 01	300 - 400	-8.28	-15.65	0.00	-12.00	-7.27	-15.58	-2.86	-22.50	0.00	-14.19	-8.99	-22.68
% to DACS 02	300	-2.52	3.16	0.00	0.43	-0.97	0.59	-2.86	-1.81	-2.70	-2.55	-7.95	-3.87
% to DACS 03	300	1.17	4.25	0.00	3.34	2.34	1.90	-1.26	8.21	0.00	0.54	-2.41	5.31
% to DACS+HLS	300	2.51	3.74	0.00	3.04	5.05	1.56	3.03	6.70	0.00	0.35	0.12	4.32
% to DACS+HLS+2-Opt	300	2.22	4.92	0.00	3.71	5.26	2.51	2.20	8.91	0.00	0.68	0.12	7.16
SI1-Like 05 - AVGs	15.22 - 114.55	12.92	1326.84	10.00	912.42	12.75	1489.35	3.09	1177.68	3.00	649.61	3.38	1372.98
SI1-Like 04 - AVGs	15.78 - 137.64	12.92	1327.94	10.00	907.72	12.75	1491.79	3.09	1144.51	3.00	641.31	3.38	1363.81
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

PG100 in Section 2.2 according to the experimental methodology in Section 5.2. As a consequence, the results in Table G.2 shows that the reduction of the traveled distances on PG100 is still better on average by -6.83% than the traveled distances of SI1-Like 04 and SI1-Like 05 that can be achieved. Of course, this -6.83% is almost same as the -6.85% of SI1-Like 06, which does not use any particular order for the tours.

Therefore, what is described recently about the percentages (-6.83% and -6.85%) gives the indication that the latest change of the tours' order in the structure of the HLS is not really significant. In comparison to the approach SI1-Like 06 on PG100, the number of vehicles has not changed on average in the approach SI1-Like 07. Moreover, the performance and results of SI1-Like 07 have deteriorated insignificantly, on average by 0.35%, in terms of the total of traveled distances on the four problem sets of R1, RC1, R2 and C2.

Nonetheless on average by -0.64%, SI1-Like 07 is able on the problem sets C1 and RC2 to improve the performance and the TD results and therefore to outperform SI1-Like 06. Despite the insignificance of the idea of changing the order of the tours in general in this section, however in a way the improvement on C1 and RC2 with -0.64% has motivated the author to keep fiddling with the latest SI1-Like approach.

5.11 What is the effect of including the inversion segment moves?

In the hybrid local search HLS used in the SI1-Like 07 approach in Section 5.10, there is no use for two moves called the inversion operators of type 1 and type 2 that are talked about in Section 5.3.4. At the beginning, type 1 is added alone as in an approach called SI1-Like 08, which is the extension of the SI1-Like 07 approach. Then as in another approach named SI1-Like 09, type 2 is added in addition to type 1. Both approaches are later tested as in Tables G.3 and G.4 on the problem group PG100 in Section 2.2 and according to the methodology of experimentation in Section 5.2.

The latest experiments of both approaches on PG100, as in Tables G.3 and G.4, indicate on average that the percentage of deviation in terms of the number of vehicles is zero to the SI1-Like 07 and SI1-like 06 approaches. In comparison to

SI1-Like 07 and SI1-like 06 in terms of the traveled distances on PG100, SI1-Like 09 with its usage of the two inversion types is shown to have a percentage of deviation equal to -0.45%, which is better on average than this -0.26% of SI1-Like 08 that uses only type 1 of the two inversion types.

These percentages (-0.45% and -0.26%) of SI1-Like 09 and SI1-Like 08 respectively point to improvements that are happening somewhere. As a result of the difference between these percentages of -0.45% and -0.26%, SI1-Like 09 is chosen over SI1-Like 08 to do the comparisons with as described below.

In contrast to the performance of SI1-like 07, the experiments of SI1-Like 09 in Table G.4 show that adding the inversion moves of type 1 and later type 2 has led on average into improving the performance in terms of the traveled distances TD by -0.46% on all the sex problem sets of PG100. Furthermore on the problem sets C1, R2, C2 and RC2, the SI1-Like approach with such inversion moves has the ability to overcome SI1-Like 06 on average by -0.83% in terms of TD. But however, it has got beaten on average by SI1-Like 06 with 0.33% for TD on the problem sets R1 and RC1.

Since SI1-Like 06 is better also on average by 0.44% than SI1-Like 07 on R1 and RC1 as in Table G.4, the inversion moves in SI1-Like 09 are considered to have succeeded only in improving the TD results on the problem sets C1, R2, C2 and RC2. Such improvement on the four problem sets mentioned earlier has resulted in a way in making the SI1-Like performance a little bit enhanced and closer, on average by 0.10% for TD instead of 0.74% to 0.94% before, to the performances in Tables 5.1 and 4.3 achieved by other deterministic and non-deterministic approaches of the literature and this is as never been before in any of the previous SI1-Like approaches. For instance on RC2, the deterministic approach CL1 [77] is now defeated on average by -0.15 for NV and -1.16% for TD whereas on C2 it is beaten in terms of TD by -0.33 (% of deviation to NV is zero here).

Accordingly, such improvement in terms of SI1-Like performance on such problem sets of C1, R2, C2 and RC2 is regarded as an encouraging point to do more experimentation with the latest SI1-Like approach that has such inversion moves.

5.12 What the 'eject and insert' strategy can do on its own?

The 'eject and insert' strategy described in Section 5.3.5 is tested here to see what it can do on its own and without the usage of the insertion procedure and the local search mentioned in Figure 5.2 and in Section 5.3.4. Therefore, four versions of the strategy, which are mentioned in S1 to S4, are tried on the problem group PG100 in Section 2.2 according to our methodology in experimentation mentioned in Section 5.2. The four versions are different from each other in the sense of the kind of region (whether sector, track, major time interval or the whole graph) used to eject the least under-constrained customer of a route so as to give the opportunity for some unvisited customer to be inserted later.

S1- The strategy using sectors as in the SI1-Like 10 deterministic approach tested in Table G.5.

S2- The strategy using tracks as in the SI1-Like 11 deterministic approach tested in Table G.6.

S3- The strategy using major time intervals as in the SI1-Like 12 deterministic approach tested in Table G.7.

S4- The strategy using the whole graph as in as in the SI1-Like 13 deterministic approach tested in Table G.8.

Using the 'eject and insert' strategy alone on the problem group PG100, as in the latest SI1-Like approaches declared earlier, deteriorates the performance and results significantly, on average and in a range between 1.71% to 2.27% for NV and 16.35% to 17.08% for TD, if compared with those of the SI1-Like 09 deterministic approach.

Nonetheless in terms of NV, the SI1-Like 13 approach is discovered on PG100 to bring, on average and by 1.71%, the least deteriorated percentage of deviation to SI1-Like 09 and to be the only approach so far to improve the NV result on the problem instance R101 by -5.00% after getting 19 vehicles. As a result and for the reasons mentioned previously, SI1-Like 13 is chosen for more experimentation as in Section 5.13.

5.13 What is the effect of hybridizing the 'eject and insert' strategy with local searches?

As a continuation to the experimental work on the 'eject and insert' strategy in Section 5.12 and after discovering its inability to work on its own, the insertion procedure and the hybrid local search HLS of the approach SI1-Like 09 as in Section 5.3.4 are hybridized with that strategy of the approach SI1-Like 13 and applied here to see the effects of using them in improving the quality of the solutions (whether feasible or infeasible) gained out of that strategy.

For checking what is described above, the feasible solutions, achieved by that strategy, are improved at first using the HLS as in a deterministic approach called SI1-Like 14, which extends the SI1-Like 13 approach used in Section 5.12. Soon as in deterministic approaches called SI1-Like 15 and SI1-Like 16, the insertion procedure and the HLS are used not just only to improve the feasible solutions but also the infeasible ones. The difference between the approaches SI1-Like 15 and SI1-Like 16 is that the first one does not use the steps h and i in Figure 5.2.

According to the experimental methodology in Section 5.2 on the problem group PG100 in Section 2.2, the latest three SI1-Like approaches are tested as in Tables G.9, G.10 and G.11 and the following findings are explored.

In SI1-Like 14 and as in Table G.9, improving the feasible solutions alone using the HLS, after applying the 'eject and insert' strategy, has led into getting performance and results on PG100 that are still worse, on average by 1.71% for NV and 2.86% for TD, compared with those of the approach SI1-Like 09.

But when the infeasible solutions gained out of that strategy are improved as well as the feasible solutions (using the insertion procedure and the HLS as in SI1-Like 15 and SI1-Like 16), this thing in itself has made the performance and the results, as in Tables G.10 and G.11 especially on the problem sets R1, RC1 and R2, a lot better in comparison to those of SI1-Like 09 and this is on average, with -0.65% to -5.88% for NV and -0.30% to -1.96% for TD, and as never been before in any of the SI1-Like approaches mentioned previously.

For instance for the first time, the SI1-Like 06 approach is now defeated on average, by SI1-Like 15 and SI1-Like 16, on R1 and RC1 with -0.49% for NV and -0.66% for TD. On the problem sets C1, C2 and RC2, SI1-Like 16 has managed

on average to match SI1-Like 09 in terms of NV and TD whereas SI1-Like 15 has deteriorated the TD results by 0.63% for C1 and 0.11% for RC2.

On the problem group PG100, the SI1-Like approach now, as consequence to improving the infeasible and the feasible solutions out of the 'eject and insert' strategy, is nearer on average, by 0.61% and 3.03% for NV and TD respectively instead of 1.28% and 3.28%, to the algorithms in Table 4.3 and a lot enhanced, by −2.52% for NV and −7.64% for TD as a substitute to −1.87% and −7.41%, in relation to the deterministic algorithms in Table 5.1. Despite the fact that the percentages of deviations as described previously are better on average, however the SI1-Like performance on PG100 is still worse on average by 5.07% for NV and 3.27% for TD than those of the deterministic approaches RVNSa [5] and RVNSb [5] and those of the algorithms LS [28] and LS+TA [28].

As a result of the improvement described in the previous paragraphs on PG100, the performance and results of the SI1-Like 15 and 16 approaches as in Tables G.10 and G.11 will be either better or closer on average, by −1.45% for NV and −1.86% for TD, during 300 seconds in relation to the average case performances of the DACS systems using the pheromone ants. And this enhancement is echoed on their best and worst case scenarios as well. For that on PG100 and in terms of performance and results, such SI1-Like approaches are better on average than DACS 01 by −5.19% for NV and −17.65% for TD and DACS 02 by −3.47% and −1.35%.

Also for instance, inserting SI1-Like 16, as a component in the DACS 03, DACS+HLS and DACS+HLS+2-Opt systems that use the pheromone ants, is going to lead on the problem set R2 into improving the performance of such systems in terms of NV and this is on average by −3.03% to −7.07% during the first 300 seconds of CPU time. Moreover in another example on the problem set R1 especially during the first 100 second of CPU time and by −1.63% in terms of NV, the addition of SI1-Like 15 into the system DACS 03 with its pheromone ants is expected to make the performance of DACS 03 get better on average.

Consequently, hybridizing the 'eject and insert' strategy with the insertion procedure and the hybrid local search HLS has made the performance and results of the SI1-Like approach much enhanced and not doing that merge would mean things are going to be worse. In order to do more fiddling with an SI1-Like approach as in Section 5.14, SI1-Like 16 is chosen instead of SI1-Like 15 and this is because it

is better on average by -0.87% for NV and -0.07% for TD on the PG100 problem group.

5.14 What if waiting time functions are included?

Since a vehicle or a customer waiting time might be reflected in the solution of a VRPTW problem instance, this section checks what the waiting time functions (whether of the vehicles or the customers) in Equation 5.3, 5.4 or 5.5 can do on the performance and the results of the approach SI1-Like 16 in Section 5.13 if they are injected instead of the servicing time functions in Equation 2.25.

In this section, four different types of SI1-Like approaches, as enumerated below, are tested as in Tables G.12, G.13, G.14 and G.15 with the waiting time functions, mentioned earlier, on the problem group PG100 in Section 2.2 according to the experimental methodology described in Section 5.2 and these approaches are different from each other as follows.

S1- The SI1-Like 17 deterministic approach uses the vehicle waiting time functions in Equation 5.3.

S2- The SI1-Like 18 deterministic approach uses the customer waiting time functions in Equation 5.4.

S3- The SI1-Like 19 deterministic approach uses a combination of the vehicle and the customer waiting time functions as in Equation 5.5.

S4- The SI1-Like 20 deterministic approach uses Equation 5.5 as in SI1-Like 19 but it panelizes the vehicle waiting time terms by multiplying them with the value -1.

In comparison to the performance and the results of SI1-Like 16 on average, including the waiting time functions in each of the four SI1-Like approaches, mentioned previously, has led on at most two problem sets into enhancing the performance and the results in terms of NV and then TD in a way that it was not before in any of the SI1-Like approaches. For example, using the vehicle waiting times alone as in SI1-Like 17 has resulted into gaining on average TD values that are better on C1 and R2 by -0.32% and -3.91% respectively.

Another instance, the interesting thing about SI1-Like 18 with its usage of just the customer waiting time functions is its ability to bring, on the problem set RC1, a very good quality NV result equal to 12.50, which is improved by -1.96% on average in contrast to that of SI1-Like 16. Also on RC2, it is able to bring a better TD result on average by -0.11. Furthermore despite the deterioration of the performance of SI1-Like 18 on R2 (if compared to the performance of SI1-Like 16) and on average in terms of NV by 3.13%, however such approach is able, for the first time, to get a very good TD result equal to 995.31 with this -5.14% as a percentage of deviation.

Moreover in association with SI1-Like 16 and when the waiting time functions of the vehicles and the customers are combined as in SI1-Like 19, the performance and results in terms of the total of traveled distances get improved respectively on average with -0.63% and -0.04% on the problem sets RC2 and R1. Also as in SI1-Like 20, the time the vehicle waiting times are panelized with the value -1, the TD results on the problem sets R1 and RC1 become, on average with -0.22% and -2.10%, equal to 1258.53 and 1392.36 in that order.

Of course once it comes to the degree of improvement in relation to the performances of other VRPTW algorithms in the literature in Tables 4.3 and 5.1, the inclusion of the waiting time functions, in each of the SI1-Like approaches mentioned above, has resulted in making the performance relatively either better or closer on average, on up to at most two problem sets, and in discovering quality solutions in the search space that are not there in any of the SI1-Like approaches mentioned in previous sections. For example with SI1-Like 17 on R2, it is the best SI1-like so far with its usage of the vehicle waiting times alone to approach, on average by 6.56% for NV and 2.05% for TD, the LS [28] and LS+TA [28] algorithms.

On RC1, SI1-Like 18 is the only one for now, after using the customer waiting times, to overcome on average the algorithm AD [14] by matching its NV value at first and then by gaining a better TD value with this -7.07%. Also, it is the only one that has the ability to defeat on R2 the TD result of the algorithm AKRed [60] by -0.01% on average. Furthermore with the usage of the vehicle and the customer waiting times together, SI1-Like 19 is the most excellent approach to overpower on average the algorithm CL1 [77] on RC2 by this -1.78% for TD knowing in mind that it is still better also by -0.15% for NV. Moreover after panelizing the vehicle waiting time terms in Equation 5.5 by -1, SI1-Like 20 with its TD result 1392.36 on

RC1 is able now on average to take-over by −0.14% the TD result 1394.26 of the deterministic algorithm RVNSa [5], which is however by 10.87% better in terms of NV than that SI1-Like approach mentioned earlier.

For each of the four SI1-like approaches using the waiting time functions, this enhancement of the performance, on at most two problem sets, is reflected also on the relationship between such new approaches and the average, best and worst case scenarios of the DACS systems that use the pheromone ants. For example during the first 300 seconds of CPU time and on R2, the performance of SI1-Like 17 with its usage of just the vehicle waiting times, as indicated from the NV and TD results of 2.91 and 1008.11 respectively, is expected to enhance, on average by −6.21% and −3.58%, the performances of the systems (DACS 01, DACS 02, DACS 03, DACS+HLS and DACS+HLS+2-Opt) even further and in a way more than that −6.21% and 0.34% of SI1-Like 16, which has NV and TD equal to 2.91 and 1049.18 correspondingly.

Another example if SI1-Like 18, using the customer waiting times alone, is embedded as a component inside the system DACS 03, the NV result of RC1 on average is expected to improve by −0.89% as a substitute to the 1.09% of deterioration during the first 100 seconds of CPU time. Also on the problem set RC2, inserting SI1-Like 19, which uses the vehicle and customer waiting times together, inside DACS 03 as a component will result, during the first 300 seconds of CPU time, in improving the performance of such system by −2.41% for NV and 3.16% for TD instead of −2.41% and 3.82%. What's more during 300 seconds, if DACS 03 and SI1-Like 20, with its penalty factor to the vehicle waiting times, are merged together, then this thing will lead on R1 into enhancing on average the DACS 03 performance from percentages of deviations equal to −0.99% for NV and 1.48% for TD into these of −0.99% and 1.25%.

Regardless of the improvement by each of the SI1-Like approaches that use the waiting time functions on some problem set cases, the negative side is that such new approaches, on the problem group PG100, are worse on average in terms of NV and by 0.38% than that of SI1-Like 16. Another negative side is that such approaches cannot dominate each other in terms of performance and results. Each one of them is doing well at some problem set cases while it is doing badly on others. Therefore, those negative sides can be avoided if the good parts of such SI1-like approaches

are merged together in a way that would make the performance more robust and reliable than that of SI1-Like 16.

5.15 A Summary of Chapter 5

In this chapter, many variants of deterministic approaches called Solomon Insertion1-Like, which its routing builder in Section 5.3.2 has some similarities to the insertion heuristic I1 of Solomon [1], are studied. Such approaches do not use any random component and there are many reasons, behind the study and investigation to them, as indicated at the start of this chapter. The aim to transfer here is to say that more fiddling with the components of a deterministic approach, as described in detail in Section 5.3, could lead at some stage into having an approach with signs of the ability to overcome on average in terms of performance that of a system using the pheromone ants.

Consequently, the start of these experimentations has been in Section 5.4 by inspecting what the maximization function can do on its own, as in SI1-Like 01 without the components mentioned in A1 to A6 in Section 5.4, and on the problem group PG100 in Section 2.2 according to the experimental methodology in Section 5.2. This SI1-Like 01 approach runs on each problem set for an averaged amount of CPU time in seconds that is less than or equal to 235.09 and this is the case of experimentation with all SI1-Like approaches tested later.

Using the maximization function alone on PG100, as in SI1-Like 01, makes the performance and results of the SI1-Like approach a lot worse, on average by 6.38% for NV and 11.31% for TD, than those of the deterministic VRPTW algorithms in Table 5.1. And in relation to the algorithms in Table 4.3 that has a lot of non-deterministic approaches, such approach is significantly inferior, with 9.86% for NV and 24.46% for TD, on average. As soon as it comes to DACS systems that use the pheromone ants, such SI1-Like approach is very poor during 100 seconds, on average by 6.46% for NV and 16.91% for TD, as well. For instance in terms of performance, the systems DACS 02, DACS 03, DACS+HLS and DACS+HLS+2-Opt are superior to such approach, on average with 7.19% for NV and 21.35% for TD, but however interestingly such approach is able on the problem sets R1 and RC1 to outperform DACS 01 and therefore to get on average these percentages -3.00% and -4.68% of

deviations for NV and TD respectively.

Later as in SI1-Like 02 of Section 5.5, a seeding strategy called No Seeding is injected in addition to the farthest in distance and the earliest deadline seeding strategies that are used by SI1-Like 01. Of course, adding the 'No Seeding' strategy has led into improving the performance and results significantly, on average by -2.49% for NV and -0.76% for TD, on all the six problem sets of PG100. As a result, this 'No Seeding' strategy has made the performance a lot nearer to those of the deterministic and non-deterministic approaches and those of the average, best and worst case scenarios of the DACS systems that use the pheromone ants. For instance during 100 seconds, the SI1-Like performance has the ability on average now, by -2.40% for NV and -3.48% for TD, to beat DACS 01 on the problem sets R1, C1 and RC1. As well, DACS 02 is defeated now on the problem sets C2 and RC2 in terms of NV with -1.72% on average.

Then, once the values of the parameters a_1 and a_2 are varied between 0 and 1 as in Section 5.6, this thing has driven the performance and results, as in SI1-Like 03, towards a further significant improvement on all the six problem sets of PG100 and on average by -2.97% for NV and -2.59% for TD. In comparison to the average, best and worst case scenarios of the DACS systems that use the pheromone ants and the other deterministic and non-deterministic approaches, the SI1-Like performance is much better now than those of the previous SI1-Like approaches. On PG100, it succeeds after 100 seconds only in outperforming DACS 01 on average by -2.22% and -4.19% for NV and TD correspondingly. Furthermore during 100 seconds, it conquers DACS 02 on average, by -3.95% for NV, on the problem sets R1, RC1, R2, C2 and RC2.

Thereafter in Section 5.7, the SI1-Like 04 approach is introduced with a removing heuristic of under-constrained tours that has the responsibility to capture such tours, to remove them and to re-insert their visited customers into the other tours via an insertion procedure and a hybrid local search HLS. As a result, the existence of the removing heuristic of under-constrained tours within SI1-Like 04 has improved, on average with -2.42% for NV and -7.49% for TD, the performance and results in a significant way on all the six problem sets of PG100. Also, it has resulted in getting the SI1-like performance near to those of the VRPTW algorithms (deterministic or non-deterministic) of the literature. Especially, it has made the performances of

deterministic approaches like I1 [1], I1-AD [14], PR1 [11] and TP [13] very much so behind.

During the first 300 seconds and in terms of the performance, the SI1-Like 04 approach with the removing heuristic described earlier is a lot closer to the average, best and worst case scenarios of the DACS systems that use the pheromone ants. For that, the SI1-Like performance is now able to beat, on average, by -1.84% for NV, that of the system DACS 03 on the problem sets R2 and RC2 and to further enhance the NV results, by -3.40% instead of -0.51%, over those of DACS 02 on the problem sets R1, RC1, R2, C2 and RC2. In relation to the performance of the system DACS 01, the same SI1-Like enhancement, described above, is achieved but on average by -4.57% for NV and -11.39% for TD and on the whole problem group PG100.

Next as in SI1-Like 05, when the order, in which the visited customers of under-constrained tours are re-inserted according to how much constrained they are as in Section 5.8, is tried on PG100, the number of vehicles on average is found to be similar to that of SI1-Like 04. But on average also, the SI1-Like 05 performance, on the problem sets C1, R2, C2 and RC2 of PG100, have deteriorated the TD results by 1.35%. Despite the deterioration described earlier, however improving the TD results, on some particular problem instances like R104, R108, RC104 and RC205, on average by -1.91% has encouraged the author to do more fiddling with the SI1-Like 05 approach.

Right afterwards in Section 5.9, the hybrid local search HLS is modified to work with the feasible solutions as well as the infeasible ones as in SI1-Like 06 and not to stop once all the visited customers of under-constrained tours are re-inserted into the other tours. Consequently, the performance and distance results have improved on average significantly by -6.85% on all the six problem sets of PG100 and have become a lot closer to those of the other state-of-the-art deterministic and non-deterministic approaches mentioned in the literature. The reduction of the number of vehicles has not changed on average from that of SI1-Like 05 and SI1-Like 04.

This improvement in terms of distance is reflected also on the relationship between the SI1-Like 06 approach and the average, best and worst case scenarios of the DACS systems using the pheromone ants. For example in addition to overcoming DACS 03 on the problem sets R2 and RC2 by -1.84% for NV, the SI1-Like

performance, during 300 seconds, is closing on average a lot to that of DACS 03 by 6.76% for TD instead of this 19.00%.

In relation to the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt and in terms of performance, SI1-Like 06 is much nearer, during 300 seconds, to such systems on average by 3.95% and 1.13% for TD and NV respectively as a substitute to these of 11.75% and 1.13% on the whole problem group PG100. Also, it enhances on PG100 the TD results on average from -11.02% to -17.10% when it comes to DACS 01 and this is on the top of beating such system by -4.57% for NV. Moreover on the problem sets R1, C1, RC1 and C2, it overcomes DACS 02 on average by -3.28% for TD and -4.07% for NV during 100 seconds. And DACS 02 is conquered on average by the SI1-Like approach, during 300 seconds on the problem sets R2 and RC2, with -2.84% for TD and -5.41% for NV.

Afterwards in SI1-Like 07, the order of the tours in HLS is changed to become in an ascending order according to the size of each tour as in Section 5.10. As a result on average, no change in terms of the number of vehicles is achieved on PG100. Also, the performance and distance results, on average, have deteriorated by 0.35% for TD in comparison to those of SI1-Like 06 on the four problem sets R1, RC1, R2 and C2 of PG100 but they have improved as well on the problem sets C1 and RC2 by -0.64% for TD, which has encouraged the author to do more experimentation with SI1-Like 07.

Of course then in Section 5.11, the addition of the inversion operator of type 1 and then type 2 for segments, as in SI1-Like 08 and SI1-Like 09 respectively, has not changed anything on average in terms of NV. But however, it has improved on average the performance and TD results, by -0.46%, on all the sex problem sets of PG100 in a way better than those of SI1-Like 07. Also by -0.83% and in comparison to SI1-Like 06, the TD results have got better on average only on the problem sets C1, R2, C2 and RC2.

But on the problem sets R1 and RC1, the SI1-Like approach with such inversion moves is beaten by SI1-Like 06, with 0.33% for TD, on average. Given that SI1-Like 07 is worse also on average by 0.44% for TD than SI1-Like 06 on R1 and RC1, the inversion moves are considered to be successful only in enhancing the TD results on four out of six problem sets, which are C1, R2, C2 and RC2. Moreover since the approach using two inversion types on PG100 is deviated (from SI1-Like 07 and

SI1-Like 06) on average by −0.45% for TD in a way that is better than this −0.26% of another approach that uses only one inversion type, SI1-Like 09 is chosen over SI1-Like 08 for more experimental work and comparison purposes.

Afterwards, the "eject and insert" strategy in Section 5.12 is tested on its own, without the HLS and the insertion procedure, in four approaches (SI1-Like 10, 11, 12 and 13) in which each approach uses a particular region - sector, track, major time interval or the whole graph. Each type of the regions mentioned earlier is used for ejecting least under-constrained customers already visited from their locations in their tours to other locations in other tours and this is in order to hopefully have the chance to insert some of the already unvisited customers removed from an under-constrained tour.

If these four SI1-Like approaches mentioned previously are compared on PG100 with the SI1-Like 09 deterministic approach, it will be recognized that their performances and results have deteriorated on average, in a range between 1.71% to 2.27% for NV and 16.35% to 17.08% for TD. But by 1.71%, SI1-Like 13 with its usage of the whole graph as a region is discovered on average to be having the least deteriorated percentage of deviation in terms of NV results and the only approach to improve, on the problem instance R101, the NV result by −5.00%.

Therefore, this discovery, mentioned earlier in itself, has motivated the author to merge the insertion procedure and the hybrid local search HLS of SI1-Like 09 with the "eject and insert" strategy of SI1-Like 13 so as to improve the feasible and the infeasible solutions out of that strategy. As in Section 5.13, the time the feasible solutions out of that strategy, as in SI1-Like 14, are enhanced using the HLS, the performance and the results show on PG100 that they are still worse, on average by 1.71% for NV and 2.86% for TD, in contrast to those of SI1-Like 09.

But as in SI1-Like 15 that does not use the steps h and i in Figure 5.2 and later in SI1-Like 16 that uses such steps, once the infeasible as well as the feasible solutions out of the "eject and insert" strategy are improved, using the insertion procedure and the HLS, things will be different. So, what is described earlier will result into gaining quality performance and results that are better, on average with −0.65% to −5.88% for NV and −0.30% to −1.96% for TD, than those of SI1-Like 09, on the problem sets R1, RC1 and R2 and in a way that has not been produced before. On average with −0.49% and −0.66% for NV and TD respectively on R1 and RC1, the

approach SI1-Like 06 is now defeated, by approaches as SI1-Like 15 and SI1-Like 16, for the first time. SI1-Like 16 has managed also on average to bring the same performance and results on the problem sets C1, C2 and RC2 as those of SI1-Like 09 whereas SI1-Like 15 is not able to do so.

For that, the SI1-Like performance is now much closer to those of the average, best and worst case scenarios of the DACS systems that use the pheromone ants and those of the deterministic and non-deterministic approaches. For instance during 300 seconds of the latest SI1-Like performance on PG100, DACS 01 and DACS 02 are now beaten even more on average by -3.47% to -5.19% for NV and -1.35% to -17.65% for TD.

Also during the first 100 second of CPU time, the performance of the system DACS 03 is expected on average, by -1.63% in terms of NV, to improve on R1 if SI1-Like 15 is inserted as a component inside DACS 03. Also, the NV and TD results of R2 computed by SI1-Like 16 suggest that if such SI1-Like approach is added as a component to DACS systems like DACS 03, DACS+HLS and DACS+HLS+2-Opt, then that addition is going to lead into improvements in terms of NV, on average by -3.03% to -7.07%, during the first 300 seconds of CPU time. On the PG100 problem group and in terms of performance, SI1-Like 16 is better, on average by -0.87% for NV and -0.07% for TD, than SI1-Like 15 and for that only SI1-Like 16 is selected for more fiddling with the waiting time functions as mentioned below.

Finally in Section 5.14 and as in the approaches SI1-Like 17, 18, 19 and 20, the waiting time functions as in Equation 5.3, 5.4 or 5.5 are introduced to be instead of the servicing time functions in Equation 2.25 with the intention of seeing the effects of that on the performance and results of SI1-Like 16. Therefore, including such waiting time functions has resulted in each of the four SI1-Like approaches into improving mainly the traveled distances on at most two problem sets of the problem group PG100 in a way that ranges on average between -0.04% to -3.91%. In addition, SI1-Like 18 with its customer waiting time functions has managed on average to improve the NV result by -1.96% on RC1.

Despite the improvement by each of the four SI1-Like approaches mentioned so far on at most two problem sets of the problem group PG100, however one of the negative sides concluded, in terms of performance and results, is that such latest SI1-Like approaches cannot dominate each other and each one of them is doing good

on some problem sets and bad on others. Also, SI1-Like 16 is better, on average in terms of NV and by 0.38%, than such latest SI1-Like approaches, which use the waiting time functions, on the problem group PG100. Nonetheless, the performances and results of such SI1-Like approaches mentioned above suggest that it would be better for future investigations to merge their good parts in order to have an SI1-Like approach that is more robust and reliable.

Chapter 6

Conclusions and Future Research

The purpose of the study in this thesis is to investigate many different variants of multiple ant colony and deterministic approaches, like DACS and SI1-Like algorithms, which try to solve the vehicle routing problem with time windows VRPTW and to discover research ideas of our own or mentioned in existing methods and heuristics and to see their effects on the decision making of such approaches. Also, another goal is to explore whether the usage and the update of the pheromone trails, locally and globally, is as advantageous as it is sometimes claimed in the literature and whether the usage of local searches and elements of a deterministic nature can replace such components at some stage successfully.

Another aim in this thesis is the interest in studying the scalability of such approaches on VRPTW problem groups that has problem instances with 100, 200 and 400 customers and to improve such approaches by making them have the ability to bring better performances and results during the amounts of CPU time allocated in a fast and more reliable way.

For that, one of the things learnt in this research is that the systematic investigation done; each time looking at the weaknesses of a particular algorithm and trying to find some way to fix those weaknesses, has led to algorithms that obtain good performance and therefore this is recommended as a strategy. Furthermore, another thing discovered is that multiple ant colony systems, like DACS 01 in Section 4.5.1, are better with the usage of local searches than without using them. But however such systems are supposed also to bring very good performance in comparison to those of other VRPTW algorithms because of using and updating the pheromone trails in a local and global way. Of course, the work in this thesis has created some

doubt about that.

Consequently as indicated from the performances of the systems DACS 02, DACS 03, DACS+HLS and DACS+HLS+2-Opt in Sections 4.6.1, 4.7.1, 4.7.3 and 4.7.4, the push-forward and push-backward strategy PFPBS and the local search with new ingredients (such as the usage of three or more move operators together, the candidate lists...etc) seem to be the most important parts to get right rather than the way in which the pheromone trails are used and updated locally and globally. In addition, when the ants of a multiple ant colony system are building their solutions in a way that is more deterministic as in Section 4.7.8, then they are able, as explored, to get results that are as good as the results could be gained by the pheromone ants. Therefore, the pheromone ants are not necessary always to bring good results.

Furthermore in Chapter 5, using a deterministic approach that captures and removes under-constrained tours could lead at some stage into improving, on some problem set cases, the average, best and worst-case performances of DACS systems that prefer using those pheromone ants. In this chapter, more conclusions about the DACS and SI1-Like approaches are made as in Section 6.1 and the future work of such approaches is ended up with and talked about as in Section 6.2.

6.1 Conclusions on DACS and SI1-Like algorithms

Definitely in addition to tackling the VRPTW multiple-objective problem and using the pheromone trails, once DACS 01 as a system is used with the XCHNG local search as in Section 4.5.1, it is possible on average to bring better performance and results on the problem group PG100 than those of a system without that local search. But, the performance and results of DACS 01 with XCHNG is not enough on that particular problem group since such system is supposed as it is indicated from the literature to show on average some sort of competitiveness, during the amount of CPU time allocated for it to run, with other well-known VRPTW algorithms.

For that, a number of components are investigated in Sections 4.5.2, 4.5.3, 4.5.4, 4.5.5 and 4.5.6 such as using different pheromone trail initializations, reinitializing the pheromone trails, reconfiguring the cycles, threading the colonies used and changing the kind of the move operator individually applied in the local search used. But all of these components mentioned earlier are not able to derive on aver-

age the search of a DACS system towards the competitiveness in performance and in computing the very good quality solutions expected. As a result, this thing has triggered many doubts about the kind of local search should be used and whether or not it is the main reason behind the poor performance of earlier DACS systems.

Then after some investigation as in Sections 4.6.1 and 4.6.2 on the problem groups PG100, PG200 and PG400, it has become clear that the inclusion of a triple move local search plays a major role, on average, in the significantly good performance and the quality results produced by the DACS 02 system. The average, best and worst case scenarios of the performance in a system that has such triple move local search are better, on average, than the scenarios of a system without that local search. However, the inclusion of such local search is not enough on average as it should be to make the DACS system be competitive or at least comparable, especially in terms of elapsed CPU time, with the other VRPTW approaches in the literature. Afterwards in Sections 4.6.3 4.6.4, 4.6.5 and 4.6.6, the usage of parallel ants, a new initialisation technique, candidate lists in a time-oriented or distance-oriented fashion, a new way of pheromone updating and new local search moves near the duplicated depots did not provide any help and did not make any push on average for the performance and the results of a DACS system to be competitive and quality either.

However, the time the push-forward and push-backward strategy PFPBS in Section 4.7.1 is added as in the DACS 03 system and tested on the problem groups PG100, PG200 and PG400, the dramatic and very significant improvement wanted, on average, will be recognized especially during the mount of CPU time allocated as never been before in any of the DACS systems previously mentioned. A DACS system with the PFPBS strategy is able on the three problem groups to improve, on average, the performance and results in the average, best and worst case scenarios in comparison to those of another DACS system without that strategy. Of course, the inclusion of the PFPBS strategy has made the DACS system have the ability to get, during the mount of CPU time allocated, performance and results that are comparable on average with those obtained in the literature by other VRPTW algorithms. Later in order to make the DACS system have the competitive factor, different types of double, triple and quadruple ant colony systems, in which each system has two, three or four colonies seeking different objectives or goals, are tried

in Section 4.7.2 but they did not seem on average to work efficiently.

Consequently, a hybrid local search HLS as in Section 4.7.3 is invented and added to the DACS system to test it on the problem groups PG100, PG200 and PG400. As a resultant of that addition, the DACS system with HLS has significantly improved, on average, the performance and the results in terms of the number of vehicles, during the mount of CPU time allocated, on the four problem sets of R1, RC1, R2 and RC2 of PG100 and on all the six problem sets of PG200 and PG400. It can be said that the new DACS system with HLS has improved, on average in terms of the number of vehicles, the average, best and worst case performances on such problem groups if compared with those of a DACS system without the HLS. But that improvement in terms of the number of vehicles has come at the expense of deteriorating on average the performance and results in terms of the total of traveled distances on all six problem sets in the three problem groups. Of course, the improvement in terms of the number of vehicles has made on average the DACS system with HLS have the sort of competitiveness wanted in computing the results when such results are compared with the results computed in the literature by other VRPTW algorithms.

Thereafter in Section 4.7.4, the addition of a variant of a 2-Opt move to the DACS system with HLS has adjusted, during the mount of CPU time allocated, the performance and results in terms of the total of traveled distances by improving them significantly on average after testing the new DACS system on the problem groups PG100, PG200 and PG400. Of course, adding the 2-Opt variant has not changed, on average, the significant reduction in terms of the number of vehicles on the three problem groups. Also on average, it has to be noted that the 2-Opt variant has improved, on the three problem groups, the average, best and worst case performances of the DACS system with HLS. In relation to the VRPTW algorithms of the literature, the performance of DACS+HLS+2-Opt is closer on average also.

Now after discovering the insignificant usage of the Savings ants in DACS systems as in Section 4.7.6, it can be said that if the usage and the update of the pheromone trails locally and globally are switched off as in Section 4.7.7, the DACS systems will perform significantly bad on average in comparison with DACS systems that use and update, locally and globally, the pheromone trails. But however according to what is described above in previous paragraphs, switching off the local search

in such DACS systems will lead on average into EVEN FAR WORSE performance and results than those of the systems that switch off the usage and the local and global update of the pheromone trails. Subsequently, an interesting discovery in Section 4.7.8 is explored in a way that suggests if there is a DACS system that uses more deterministic ants without any pheromone update and usage locally and globally, then that system is going to behave on average, in terms of performance and NV results, better than a DACS system that uses the pheromone ants.

Of course, that interesting discovery has been the trigger to explore a number of SI1-Like deterministic approaches in order possibly to merge such SI1-Like approaches with DACS systems at future work if they have proved to have the ability of improving things better. Therefore, the start to the research in deterministic approaches is the significant improvement gained of the SI1-Like performance and results on average, in comparison to those of an SI1-Like approach using the maximization function alone as in Section 5.4, on all six problem sets of the problem group PG100, after adding a "No Seeding" strategy and varying the parametric values of a_1 and a_2 as in Sections 5.5 and 5.6. On PG100, the significantly improved SI1-Like approaches get closer on average in terms of performance and results to those of the deterministic and non-deterministic VRPTW algorithms of the literature and those of the average, best and worst case scenarios of the DACS systems using the pheromone ants. Consequently during 100 seconds of CPU time, the SI1-Like performance is now better on average than that of DACS 02 in terms of the number of vehicles on the problem sets R1, RC1, R2, C2 and RC2 and that of DACS 01 on PG100 in terms of the number of vehicles and the total of traveled distances.

Then, the time a heuristic in Section 5.7 is designed and later added in particular to capture under-constrained tours, remove them, and re-insert their customers into other tours using the insertion procedure and the hybrid local search HLS, things have started to move significantly forward and to get improved on average in terms of performance and results on all the six problem sets of the problem group PG100. In relation to the performances of the other deterministic and non-deterministic VRPTW algorithms in the literature, the SI1-Like performance and results have become nearer on average. The SI1-Like performance and results get also nearer, on average, when they are compared with those of the average, best and worst case scenarios of the DACS systems using the pheromone ants. For that in addition to

beating even further the performances of DACS 01 on PG100 and DACS 02 on R1, RC1, R2, C2 and RC2, the SI1-Like performance, during 300 seconds of CPU time, overcomes now on average that of DACS 03 in terms of the number of vehicles on the problem sets R2 and RC2. Later in another SI1-Like approach as in Section 5.8, the change of the ordering of the customers to be inserted into the other tours has led on average into deteriorating the performance and results slightly on the problem sets C1, R2, C2 and RC2 of PG100 and into improving them on the problem sets R1 and RC1 to some extent.

Nonetheless, slight modifications to HLS in Section 5.9, such as the ability to improve on the feasible as well as the infeasible solutions and not to stop once all customers of an under-constrained tour are re-inserted into other tours, within SI1-Like have improved further on average the performance and the results significantly in terms of the total of traveled distances on all the six problem sets of the problem group PG100. In the latest SI1-Like performance and results, the quality of the number of vehicles on average has not changed and it is the same as always. The slight changes have made SI1-Like get closer on average in performance and results to those of the well-known deterministic and non-deterministic VRPTW approaches in the literature. In comparison to the performances and results of the average, best and worst case scenarios of the DACS systems using the pheromone ants, the SI1-Like performance and results get nearer on average too. As a result during 300 seconds of CPU time, the SI1-Like performance is defeating on average in a very great way those of DACS 01 and DACS 02 on PG100 and that of DACS 03 on R2 and RC2 and it is closing very much so on average to that of DACS 03 on R1, C1, RC1 and C2 and those of DACS+HLS and DACS+HLS+2-Opt on PG100.

Later in Section 5.10, the change of the ordering of the tours used within HLS has deteriorated on average the SI1-Like performance and the results slightly on the problem sets of R1, RC1, R2 and C2 of PG100. But on the problem sets C1 and RC2, the SI1-Like performance and results have got modestly better on average. Thereafter in Section 5.11, the inclusion of the two inversion moves of type 1 and 2 for segments within SI1-Like has led on average into improving slightly the performance and the results even further on the problem sets of C1, R2, C2 and RC2 only as never been before in any of the SI1-Like approaches described earlier. On the problem sets R1 and RC1, the SI1-Like performance and the results have

worsened on average a little bit.

Next, an "eject and insert" strategy is tried on its own so as to eject firstly as in Section 5.12 under-constrained customers from their tours in a region (i.e. sector, track, major time intervals or graph) into other tours and then to insert secondly the customers of a removed under-constrained tour. Of course on PG100, the usage of an "eject and insert" strategy on its own has not made any significant improvement on average and the SI1-Like performance and results have become very much so worse. Also, improving the new feasible solutions only out of that strategy using the hybrid local search HLS would lead, on PG100, to improvements on average but they are still bad compared to those of previous SI1-Like approaches.

But however after merging the insertion procedure and the hybrid local search HLS with the "eject and insert" strategy as in Section 5.13 to enhance the infeasible and the feasible solutions out of that strategy, the performance and the results have improved on average on the three problem sets of R1, RC1 and R2 of PG100 as never been in any of the SI1-Like approaches and with gaining the same quality of performance and results on C1, C2 and RC2 as before.

Of course, the most recent improvement in terms of performance and results has its own reflection on the relationship between the latest SI1-Like approach and the average, best and worst-case scenarios of the DACS systems using the pheromone ants and the other deterministic and non-deterministic VRPTW algorithms of the literature. For that on average, it is for the first time for an SI1-Like performance, during 300 seconds of CPU time, to beat on R2 those of DACS+HLS and DACS+HLS+2-Opt in addition to beating that of DACS 03 on R2 and RC2 and those of DACS 02 and DACS 01 on PG100.

According to what is described above, if such latest SI1-Like approach is merged or hybridized inside the DACS systems, then it would be guaranteed on average to improve the average, best and worst-case performances of DACS 01 and DACS 02 on PG100 and to enhance those of the systems DACS 03, DACS+HLS and DACS+HLS+2-Opt on one or two problem sets of PG100. For more information about merging an SI1-Like approach and a DACS system, check Section 6.2.

Finally, different combinations of waiting time functions in Section 5.14 are tried and some quality performance and results are recognized, on at most two problem sets of PG100 and as never been before, from each of the SI1-Like approaches tested

with such waiting time functions. However, one downfall thing is that each of the SI1-Like approaches, tested with different combinations of waiting time functions, cannot dominate each other in terms of performance and computing results because each one of them is good at tackling some problem set cases in which the other SI1-Like approaches cannot do any better on them. In order to avoid that downfall, the SI1-Like approaches that are good at tackling different problem set cases could be merged together to form an SI1-Like approach that is more robust and reliable.

6.2 Future work on DACS and SI1-Like algorithms

After recognizing that there are signs of the ability of improving the average, best and worst-case performances of DACS systems (like DACS 01 and DACS 02 on the problem group PG100 and DACS 03, DACS+HLS and DACS+HLS+2-Opt on one or two problem sets of PG100) between 100 to 300 seconds of CPU time if such DACS systems are merged or hybridized with SI1-Like approaches, one of the things will be worked at in the future is to make that merge or hybridization happen. But before doing that, improving the abilities of SI1-Like approaches even more will be thought about first by exploring route elimination techniques that take into consideration the amount of CPU time that is going to be elapsed in that process of route elimination. Since multiple ant colony systems have the ability to find very good quality solutions in a very short amount of CPU time, then the route elimination techniques have to be developed in a way that regards that matter seriously.

Therefore in order to do just what is described above, the removing heuristic of under-constrained tours in Section 5.3.3 and the "eject and insert" strategy in Section 5.3.5 could be re-designed in a way in which more sophisticated heuristics and strategies could take into consideration the removal of not just one under-constrained tour in a solution, as it is the case in the most recent SI1-Like approaches, but two or more under-constrained tours. For that also, ejection techniques that would locate two or more under-constrained customers at a time instead of one, in order to remove them from their locations and re-insert them feasibly into other locations in other tours, are one of the things that are in need to be explored, investigated and studied carefully. As a result, locating the under-constrained customers of a tour or more at an area in the graph, in which such customers share certain spatial

and temporal features, could be looked at in such ejection techniques. Such shared spatial and temporal features between the under-constrained customers could be as being located in the same sector and/or track and/or having the same major time intervals. Such thing is in order to possibly to try to re-locate under-constrained customers either in the same route they are in or to another route in which other customers with the same spatial and temporal features also exist. The insertion techniques might work, as above, but with considering constrained customers first.

As indicated from the literature review in Section 2.6, it is known that the best-accept strategy is used, by the hybrid local search HLS applied within an SI1-Like approach, in a way in which it chooses and accepts, at each step of HLS, the best of three or more neighboring solutions, created from three or move operators. It is known also that the first-accept strategy, which could accelerate the performance of that SI1-Like approach during the amount of CPU time allocated, is not used and so such strategy is not taken into consideration. Therefore, the first-accept strategy, according to what is mentioned above, is one of the techniques might be used to speed up the performance of an SI1-Like approach. Also, the effects of changing the objective function of HLS could be investigated in order to escape local minima faced during the search and to seek for quality solutions. Of course, the escape of local minima might lead into increasing the amount of CPU time elapsed during the search but such thing has to be avoided in a way in which the local minima escapes needed are done without any extra costs in CPU time terms. Therefore, the issue of escaping local minima must be studied and investigated carefully. Now, once it is possible to make an SI1-Like approach that is fast enough in terms of CPU time and able to find the quality solutions wanted at the same time, the merge of the DACS systems with the SI1-Like approaches could be done soon later.

The purpose of what is described above is to prove that with deterministic approaches quality solutions can be gained in a way as good as those of the multiple ant colony approaches that use the pheromone ants. Of course, the issue of using deterministic approaches has started from Section 4.7.8 when it has suggested that the pheromone component of ants is nothing but a perturbation force and ants that are more deterministic could be used instead. Therefore as a continuation of what is suggested in Section 4.7.8, the DACS systems that use the deterministic ants rather than the pheromone ants will be studied and investigated further. One of the

ways, could be followed to enhance the DACS systems that use the deterministic ants, is to encourage the insertion procedure of the ants with infeasible solutions to use ejection techniques that allow under-constrained customers to be removed from their tours and signed as unvisited in order to allow some of the unvisited customers that are more constrained in some sense to be inserted instead.

In the insertion procedure of ants, it could be tried also removing two or more under-constrained customers, which have the same spatial and temporal features, from the same tour they are in or from different tours and signing them as unvisited in order to let other customers who are constrained and not visited yet to be inserted. Of course, the spatial and temporal features of two or more under-constrained customers could be as having the same sector, track and/or major time intervals. Later, the re-insertion of such under-constrained customers that are removed could be easier. Also, when inserting the constrained customers not visited yet, the locations where to insert them exactly in the tours and near to which group of visited customers could be specified. Maybe, it is preferred to insert them near the customers who are located in the same sector and/or track spatially and/or the same major time intervals temporally. Also when deterministic ants construct feasible solutions, the idea of capturing and removing under-constrained tours, used in SI1-Like approaches, could be applied in order to increase the quality of the performance and the results of the DACS system involved.

Once quality performance and results are gained from the DACS systems with deterministic ants in a way as good as the performance and results of DACS systems with pheromone ants, then that in itself would prove what is said above and in Section 4.7.8 about the pheromone component and its perturbation nature. Also, another thing to investigate and study is why the saving ants in Section 4.7.6 are not able to find good solutions and how to convert them from ants that depend on the pheromone component into ants that are more deterministic in some sense. Furthermore, there is an interest in testing our future work not just on the problem groups with 100, 200 and 400 customers but also on problem groups with 800 and 1000 customers. In addition, there are aims to explore, study and investigate what are the sorts of features that make certain problem instances harder and harder. Consequently, the start point might be where the time windows of a problem instance like R101 of the problem group PG100 are varied from the tightest to widest widths.

Appendix A

Best results obtained

In this section, Tables A.1, A.2 and A.3 present a number of solutions equal to 12 solutions and such solutions represent the best results that our DACS systems, especially DACS+HLS and DACS+HLS+2-Opt, have managed to bring on some of the 56 problem instances related to the six problem sets R1, C1, RC1, R2, C2 and RC2 of the problem group PG100 in Section 2.2 of Solomon [1]. In each of the Tables A.1, A.2 and A.3, four solutions are mentioned for four different problem instances and therefore each of the four solutions is a solution for a particular problem instance, which its name mentioned underneath the PNo field. Now, each of the four solutions is with a number of vehicles and a total of traveled distances underneath the fields NV and TD respectively.

The tours of each of the four solutions in Table A.1, A.2 or A.3 is mentioned underneath the field Tours and the graph of such tours has an index between A.1 to A.12 (inclusive) and this index is mentioned underneath the field Figure. Now in the figures indexed between A.1 to A.12, the demands of the visited customers are reflected by the black circles and each black circle describes how much big or small the demand of a customer is. The bigger the black circle the bigger the demand and the smaller the black circle the smaller the demand. Also, the time windows of the customers are reflected by the white rectangles and each white rectangle indicates how much wide or tight the time window of a customer is. The wider the white rectangle is the wider the time window and the tighter the white rectangle is the tighter the time window.

Finally, the tours in the figures indexed between A.1 to A.12 are the colored lines that go by passing the customers in order either to pick up or to deliver items.

If the servicing process is a pick up process, then the colored lines of the tours get thicker and thicker. However if the servicing process is a delivery process, then the colored lines of the tours become thinner and thinner.

Table A.1: The best solutions, for some of the problem instances in the problem sets R1 and C1 of the problem group PG100, as computed by DACS+HLS and DACS+HLS+2-Opt.

DACS system	PNo.	NV	TD	Figure	Tours
DACS+HLS	R112	9.00	1024.99	A.1	0 69 30 70 51 9 81 33 50 3 77 68 80 0 61 16 86 17 45 8 46 36 47 48 0 95 92 98 85 91 44 38 14 42 100 37 93 0 59 99 6 94 87 2 57 15 43 97 13 58 0 27 52 31 88 7 82 18 83 84 5 60 96 89 0 12 29 24 55 25 67 23 56 39 4 0 21 72 75 41 22 74 73 40 53 26 54 0 62 11 19 49 64 63 90 10 32 0 28 76 79 78 34 35 71 65 66 20 1
DACS+HLS+2-Opt	R111	10.00	1175.77	A.2	0 33 81 65 66 71 35 34 50 0 52 31 88 62 11 63 10 90 32 20 70 0 39 23 67 22 41 75 74 72 2 0 7 19 64 49 36 46 47 48 82 0 92 98 14 44 38 43 57 97 13 96 0 83 18 45 8 84 17 86 85 100 37 0 59 91 16 61 99 87 6 94 93 5 60 89 0 95 42 15 53 40 73 56 4 21 58 0 28 12 76 79 29 24 55 25 54 26 0 27 1 69 30 51 9 78 3 68 80 77
DACS+HLS	C102	10.00	828.94	A.3	0 57 55 54 53 56 58 60 59 0 67 65 63 62 74 72 61 64 68 66 69 0 90 87 86 83 82 84 85 88 89 91 0 81 78 76 71 70 73 77 79 80 0 20 24 25 27 29 30 28 26 23 22 21 0 98 96 95 94 92 93 97 100 99 0 5 3 7 8 10 11 9 6 4 2 1 75 0 32 33 31 35 37 38 39 36 34 0 13 17 18 19 15 16 14 12 0 43 42 41 40 44 46 45 48 51 50 52 49 47
DACS+HLS+2-Opt	C104	10.00	824.78	A.4	0 90 87 86 83 82 84 85 88 89 91 0 20 24 25 27 29 30 28 26 23 22 21 0 43 42 41 40 44 46 45 48 51 50 52 49 47 0 98 96 95 94 92 93 97 100 99 0 67 65 62 74 72 61 64 68 66 69 0 13 17 18 19 15 16 14 12 10 0 81 78 76 71 70 73 77 79 80 63 0 5 3 7 8 11 9 6 4 2 1 75 0 57 55 54 53 56 58 60 59 0 32 33 31 35 37 38 39 36 34

Table A.2: The best solutions, for some of the problem instances in the problem sets RC1 and R2 of the problem group PG100, as computed by DACS+HLS and DACS+HLS+2-Opt.

DACS system	PNo.	NV	TD	Figure	Tours
DACS+HLS	RC101	14.00	1732.39	A.5	0 14 47 12 73 79 46 55 0 52 99 57 86 74 24 0 65 83 23 21 19 18 89 0 59 75 87 97 58 77 0 82 11 15 16 9 10 13 17 0 92 95 62 67 71 94 96 54 68 0 27 29 31 30 34 26 32 93 0 42 44 61 81 43 70 0 69 98 88 53 78 60 0 28 33 85 50 91 80 0 63 76 51 22 49 20 48 25 0 64 90 84 56 66 0 72 39 36 40 38 41 37 35 0 5 45 2 7 6 8 3 1 4 100
DACS+HLS+2-Opt	RC105	14.00	1549.14	A.6	0 31 29 27 30 28 26 32 34 50 91 0 19 23 18 22 49 20 77 0 33 63 85 84 56 66 0 88 79 73 60 0 39 36 44 38 40 37 35 43 70 0 65 52 86 87 59 97 75 58 0 42 61 81 41 72 54 0 12 14 15 16 47 78 55 68 0 51 76 89 48 21 25 24 0 82 11 9 10 13 17 0 2 45 3 5 8 6 7 46 4 1 100 0 69 90 53 98 0 92 95 62 67 71 94 93 96 80 0 64 83 99 57 74
DACS+HLS	R204	2.00	849.34	A.7	0 27 52 18 82 7 88 31 69 10 62 11 19 48 83 60 5 44 38 14 99 87 84 8 45 46 47 36 49 64 63 90 32 20 66 65 71 35 34 78 3 77 29 24 25 26 40 58 0 94 95 92 98 37 42 43 15 57 41 22 75 56 23 67 39 12 76 28 53 79 33 81 9 51 30 70 1 50 68 80 54 55 4 72 74 73 21 2 13 97 100 91 16 86 17 61 85 93 59 96 6 89
DACS+HLS+2-Opt	R211	2.00	914.69	A.8	0 21 2 95 92 59 5 45 83 99 98 85 61 16 86 38 44 14 42 87 57 15 41 22 75 23 67 39 56 72 73 40 26 53 94 6 84 17 46 48 60 96 13 97 37 43 100 91 93 89 0 52 27 28 12 29 76 69 31 88 62 11 19 7 82 18 8 47 36 49 64 63 90 30 51 71 65 9 81 33 3 79 78 34 35 66 20 32 10 70 1 50 77 68 80 24 54 55 25 4 74 58

Table A.3: The best solutions, for some of the problem instances in the problem sets C2 and RC2 of the problem group PG100, as computed by DACS+HLS and DACS+HLS+2-Opt.

DACS system	PNo.	NV	TD	Figure	Tours
DACS+HLS	C204	3.00	590.60	A.9	0 67 63 62 74 72 61 64 66 69 68 65 49 55 54 53 56 58 60 59 57 40 44 46 41 42 45 51 50 52 47 43 48 0 93 5 75 2 1 99 100 97 92 94 95 98 7 3 4 89 91 88 84 86 83 82 85 76 71 70 73 80 79 81 78 77 96 87 90 0 20 22 24 27 30 29 6 32 33 31 35 37 38 39 36 34 28 26 23 18 19 16 14 12 15 17 13 25 9 11 10 8 21
DACS+HLS+2-Opt	C203	3.00	591.17	A.10	0 93 5 75 2 1 99 100 97 92 94 95 98 7 3 4 89 91 88 84 86 83 82 85 76 71 70 73 80 79 81 78 77 96 87 90 0 20 22 24 27 30 29 6 32 33 31 35 37 38 39 36 34 28 26 23 18 19 16 14 12 15 17 13 25 9 11 10 8 21 0 67 63 62 74 72 61 64 66 69 68 65 49 55 54 53 56 58 60 59 57 40 44 46 45 51 50 52 47 42 41 43 48
DACS+HLS	RC202	3.00	1368.23	A.11	0 45 5 3 1 42 39 36 44 69 88 73 16 99 53 78 79 8 6 46 2 55 68 54 43 35 37 72 96 93 94 80 0 91 92 95 85 63 33 28 26 27 29 31 30 62 67 71 61 41 38 40 81 90 84 49 20 66 56 50 34 32 89 48 21 24 25 77 75 58 52 0 65 82 98 12 14 47 15 11 83 64 23 19 51 76 18 22 57 86 87 9 10 97 59 74 13 17 7 4 60 100 70
DACS+HLS+2-Opt	RC203	3.00	1073.02	A.12	0 45 5 3 1 42 39 36 43 44 61 69 88 98 99 87 9 53 78 79 8 6 7 60 73 14 12 10 13 16 47 17 86 74 59 97 75 58 77 25 57 0 91 92 95 85 63 33 32 34 31 29 27 26 28 30 62 50 67 71 72 41 38 40 35 37 54 81 96 93 94 80 0 90 65 82 15 11 52 83 64 24 23 21 48 18 76 89 19 49 22 20 51 84 56 66 68 55 2 46 4 100 70

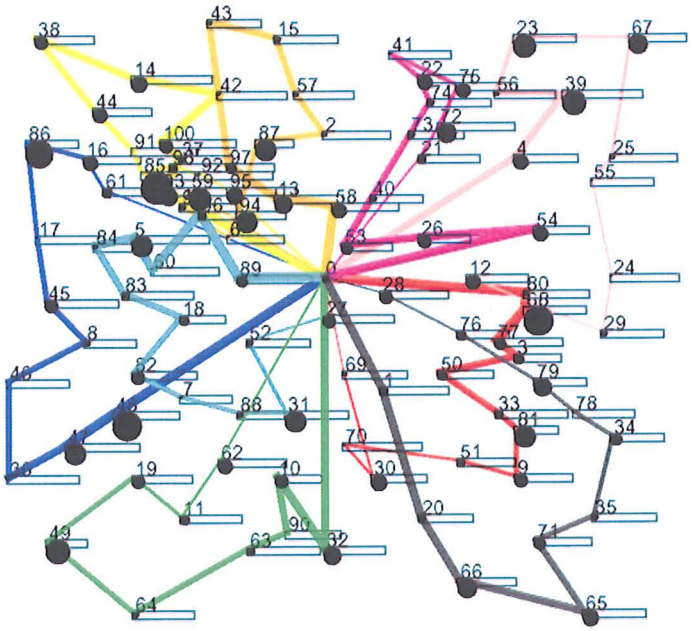


Figure A.1: The best solution of the problem instance R112 of the problem group PG100 as computed by DACS+HLS.

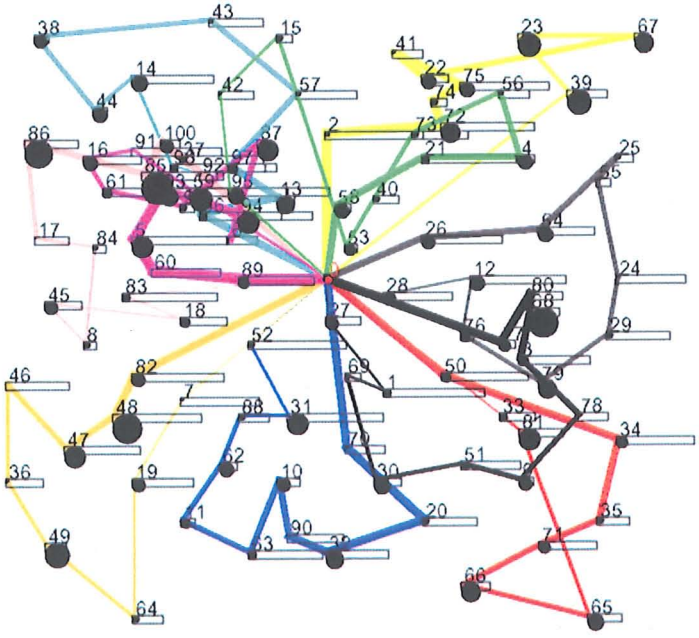


Figure A.2: The best solution of the problem instance R111 of the problem group PG100 as computed by DACS+HLS+2-Opt.

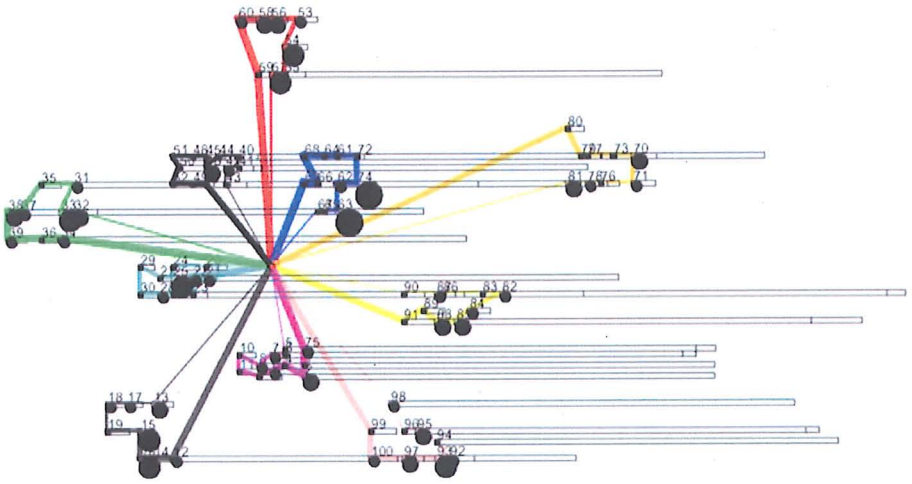


Figure A.3: The best solution of the problem instance C102 of the problem group PG100 as computed by DACS+HLS.

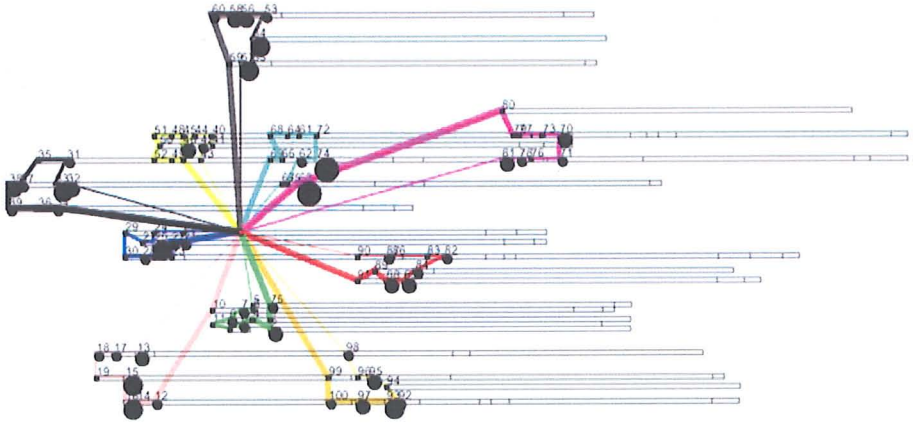


Figure A.4: The best solution of the problem instance C104 of the problem group PG100 as computed by DACS+HLS+2-Opt.

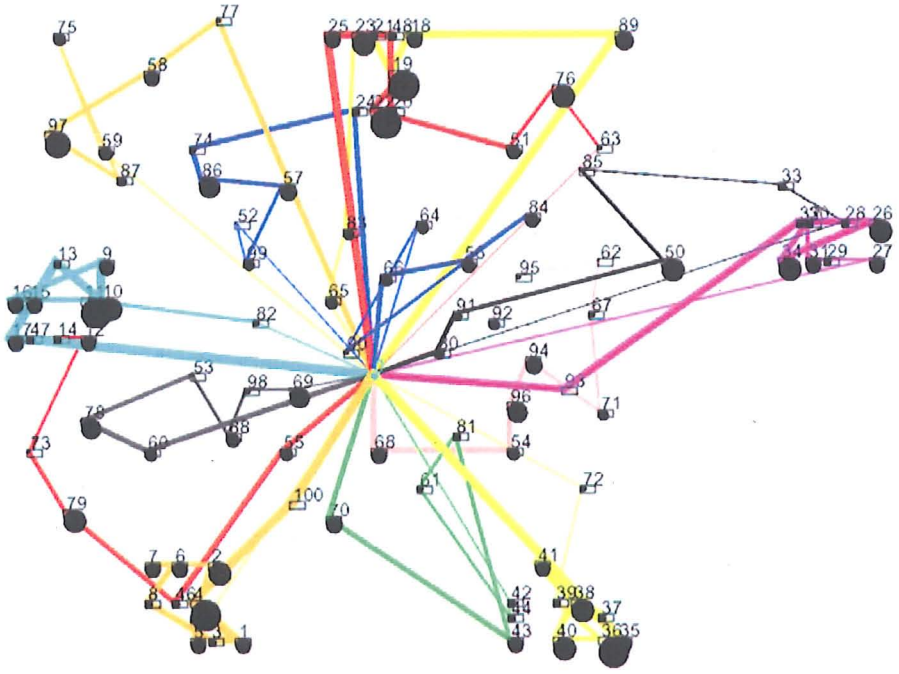


Figure A.5: The best solution of the problem instance RC101 of the problem group PG100 as computed by DACS+HLS.

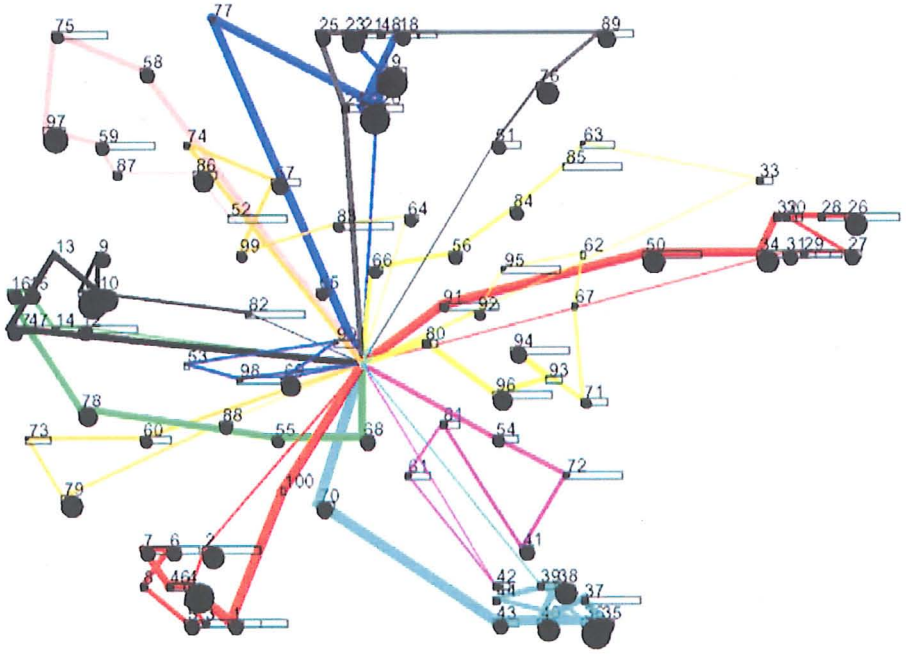


Figure A.6: The best solution of the problem instance RC105 of the problem group PG100 as computed by DACS+HLS+2-Opt.

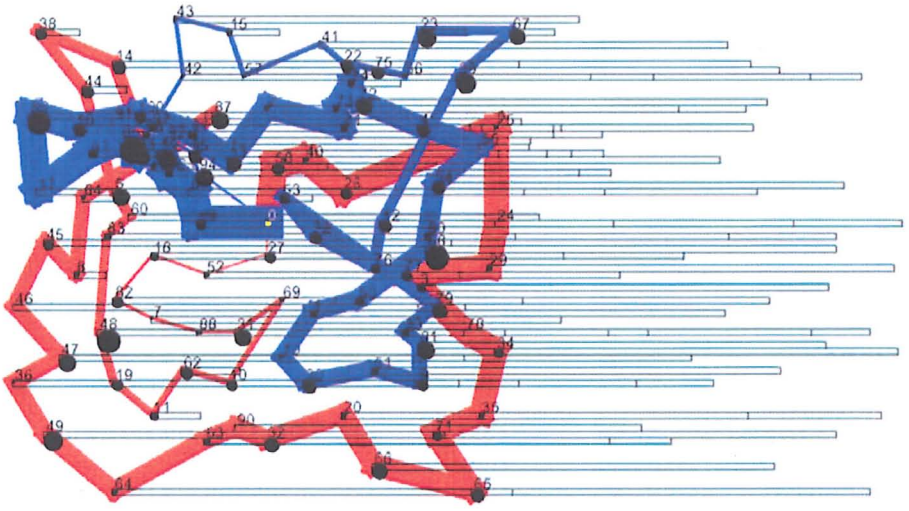


Figure A.7: The best solution of the problem instance R204 of the problem group PG100 as computed by DACS+HLS.

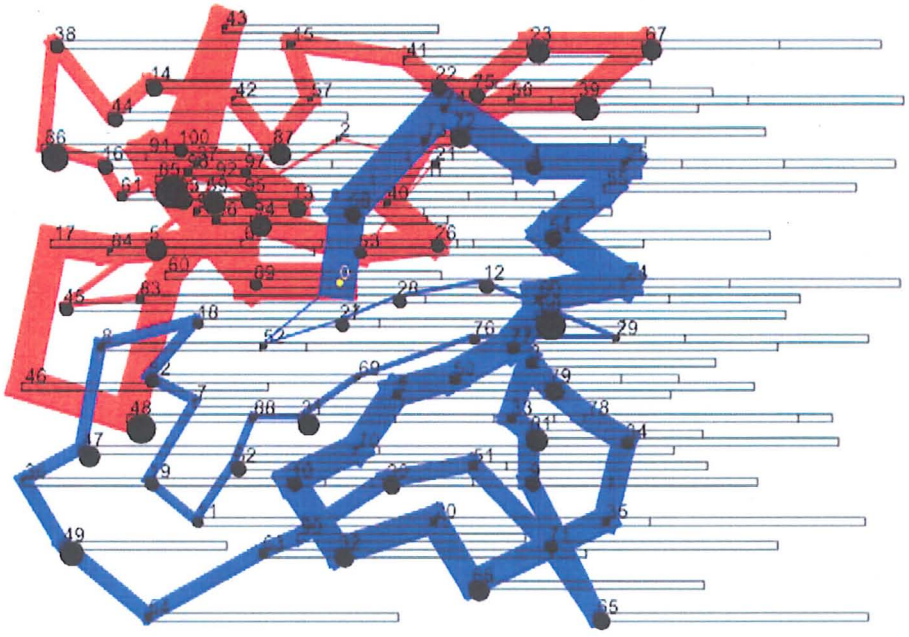


Figure A.8: The best solution of the problem instance R211 of the problem group PG100 as computed by DACS+HLS+2-Opt.

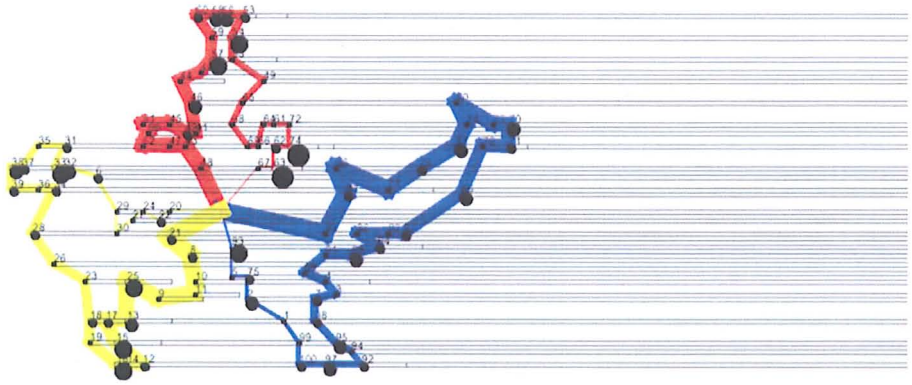


Figure A.9: The best solution of the problem instance C204 of the problem group PG100 as computed by DACS+HLS.

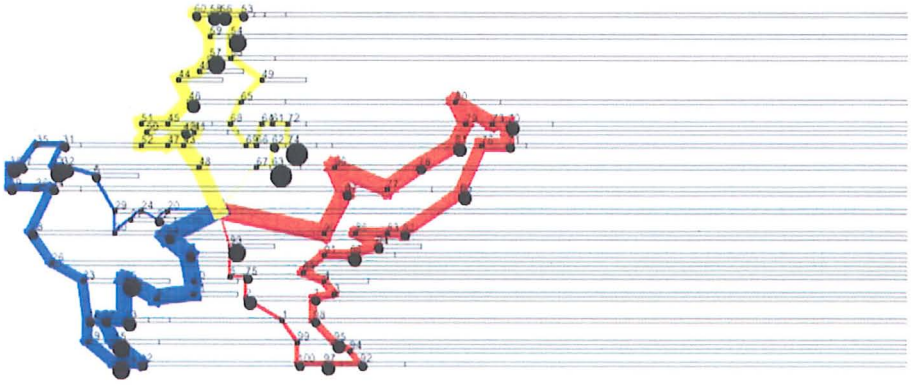


Figure A.10: The best solution of the problem instance C203 of the problem group PG100 as computed by DACS+HLS+2-Opt.

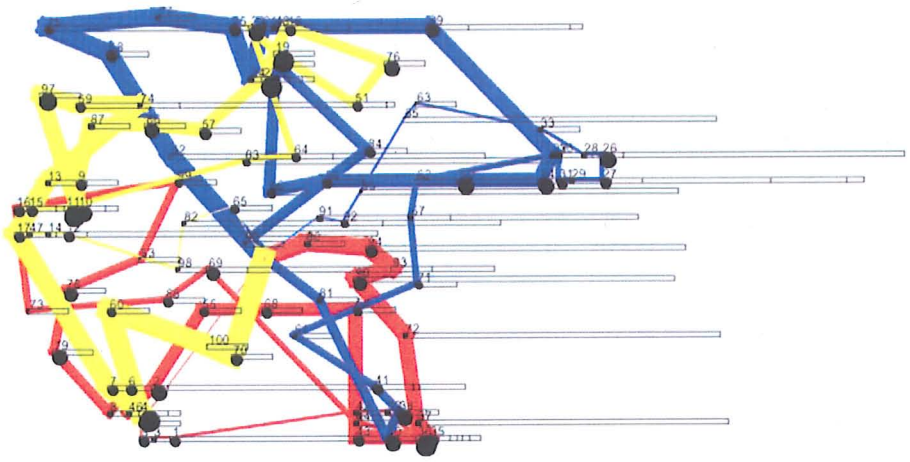


Figure A.11: The best solution of the problem instance RC202 of the problem group PG100 as computed by DACS+HLS.

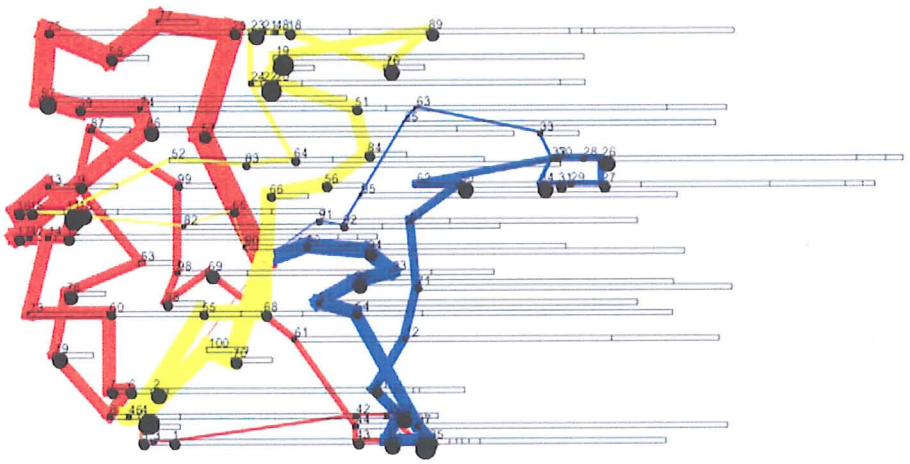


Figure A.12: The best solution of the problem instance RC203 of the problem group PG100 as computed by DACS+HLS+2-Opt.

Appendix B

Tables of results related to the system DACS 01

This section mentions in Tables B.1 to B.8 information about the experiments done to the system DACS 01. All tables, mentioned in A1 to A4, have the results, either on the problem group PG100 in Section 2.2 or the problem set R1 of that group, after three runs of 300 to 400 or 600 to 800 seconds. But, this is with the exception of Table B.7, which has its results after nine runs. Another exception is that the results of Table B.4 are gained after using the author's problem set R1-200.

A1- Tables B.1 and B.2 represent the best and worst case performances of the system DACS 01 as in Section 4.5.1, which uses the XCHNG local search.

A2- Tables B.3 to B.5 represent, on two different problem sets, the average case performances of mainly two DACS systems, which either initialize as in Section 4.5.2 the pheromone trails with V-1 and V in the VMIN and DMIN colonies respectively or initialize and reinitialize such trails with $1/(n.J_{\Psi}^{gb})$ as in Section 4.5.3.

A3- Tables B.6 and B.7 represent the average case performances of two DACS systems, which use the reconfigured cycles in Section 4.5.4 and the two threaded-colonies (of VMIN and DMIN) as in Section 4.5.5 respectively.

A4- Table B.8 represents the average case performances of six DACS systems, which use a local search each with one operator of the following moves in Section 4.5.6 - OrOpt1, OrOpt2, Relocate, 2-Opt*, Swap and XCHNG.

Table B.1: Comparison between the best case performances of DACS 01 and other VRPTW algorithms, after three runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	300 - 400	20.00	1922.25	10.00	850.02	16.00	2009.01	4.00	1744.30	3.00	586.33	4.00	1783.90
Freq.		1	1	3	1	1	1	3	1	3	1	2	1
% to SA+LNS [29]	1800	5.26	16.44	0.00	2.54	14.29	18.36	0.00	35.46	0.00	-0.88	0.00	21.68
02 - Best	300 - 400	18.00	1868.77	10.00	959.54	14.00	1801.63	4.00	1513.33	3.00	716.45	4.00	1684.41
Freq.		1	1	3	1	2	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	5.88	25.75	0.00	15.75	16.67	15.88	33.33	22.33	0.00	21.11	33.33	21.41
03 - Best	300 - 400	15.00	1605.63	10.00	989.86	12.00	1526.55	3.00	1373.59	3.00	760.07	3.00	1510.05
Freq.		3	1	3	1	3	1	3	1	3	1	2	1
% to SA+LNS [29]	1800	7.14	32.30	0.00	19.54	9.09	20.99	0.00	41.93	0.00	28.57	0.00	37.61
04 - Best	300 - 400	12.00	1306.83	10.00	1128.73	12.00	1412.43	3.00	1103.16	3.00	804.98	3.00	1093.47
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	20.00	33.18	0.00	36.85	20.00	24.39	50.00	32.29	0.00	36.30	0.00	29.98
05 - Best	300 - 400	15.00	1680.21	10.00	852.95	16.00	1760.98	3.00	1428.22	3.00	586.33	4.00	1884.67
Freq.		2	1	3	2	3	1	3	1	3	1	2	1
% to SA+LNS [29]	1800	7.14	21.13	0.00	2.90	23.08	7.65	0.00	37.75	0.00	-0.43	0.00	42.49
06 - Best	300 - 400	14.00	1580.47	10.00	884.22	14.00	1681.46	3.00	1283.71	3.00	640.18	3.00	1618.82
Freq.		2	1	3	1	3	1	3	1	3	1	1	1
% to SA+LNS [29]	1800	16.67	25.64	0.00	6.67	16.67	22.18	0.00	34.24	0.00	8.78	0.00	36.35
07 - Best	300 - 400	12.00	1330.21	10.00	971.87	13.00	1602.06	3.00	1225.70	3.00	631.56	4.00	1561.00
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	20.00	19.32	0.00	17.24	18.18	30.15	50.00	36.02	0.00	7.35	33.33	42.72
08 - Best	300 - 400	11.00	1173.60	10.00	921.24	12.00	1472.72	3.00	1024.46	3.00	652.51	3.00	1249.78
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	22.22	21.38	0.00	11.13	20.00	29.21	50.00	38.93	0.00	10.91	0.00	42.73
09 - Best	300 - 400	13.00	1592.90	10.00	996.27	-	-	3.00	1260.58	-	-	-	-
Freq.		1	1	3	1	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	18.18	33.03	0.00	20.19	-	-	0.00	33.58	-	-	-	-
10 - Best	300 - 400	12.00	1342.31	-	-	-	-	3.00	1376.11	-	-	-	-
Freq.		2	1	-	-	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	20.00	19.14	-	-	-	-	0.00	42.16	-	-	-	-
11 - Best	300 - 400	13.00	1388.08	-	-	-	-	3.00	1186.08	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	30.00	26.56	-	-	-	-	50.00	29.80	-	-	-	-
12 - Best	300 - 400	11.00	1214.79	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to SA+LNS [29]	1800	10.00	25.65	-	-	-	-	-	-	-	-	-	-
DACS 01 - AVGs	300 - 400	13.83	1500.51	10.00	950.52	13.63	1658.35	3.18	1319.93	3.00	672.30	3.50	1548.26
% to SA+LNS [29]	1800	13.67	24.64	0.00	14.74	17.15	20.26	16.55	34.64	0.00	13.98	7.69	33.60
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table B.2: Comparison between the worst case performances of DACS 01 and other VRPTW algorithms, after three runs, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	300 - 400	21.00	1885.27	10.00	855.07	17.00	1997.17	4.00	1765.67	3.00	589.98	5.00	2008.60
Freq.		2	1	3	1	2	1	3	1	3	1	1	1
% to SA+LNS [29]	1800	10.53	14.20	0.00	3.15	13.33	22.73	0.00	35.49	0.00	-0.27	25.00	32.23
02 - Worst	300 - 400	19.00	1882.45	10.00	994.18	15.00	1826.80	4.00	1570.58	3.00	798.84	4.00	1735.73
Freq.		2	1	3	1	1	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	11.76	26.67	0.00	19.93	25.00	17.50	0.00	34.70	0.00	13.47	0.00	33.39
03 - Worst	300 - 400	15.00	1618.80	10.00	1141.33	12.00	1668.25	3.00	1433.89	3.00	846.79	4.00	1553.00
Freq.		3	1	3	1	3	1	3	1	3	1	1	1
% to SA+LNS [29]	1800	7.14	32.92	0.00	37.83	9.09	30.48	0.00	39.64	0.00	12.43	33.33	37.95
04 - Worst	300 - 400	12.00	1332.30	10.00	1185.04	12.00	1525.61	3.00	1152.72	3.00	851.27	3.00	1191.47
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	20.00	34.60	0.00	43.68	20.00	31.97	0.00	44.32	0.00	26.65	0.00	37.59
05 - Worst	300 - 400	16.00	1704.37	10.00	854.55	16.00	1926.43	3.00	1466.62	3.00	638.42	5.00	1839.62
Freq.		1	1	3	1	3	1	3	1	3	1	1	1
% to SA+LNS [29]	1800	14.29	19.51	0.00	3.09	14.29	23.19	0.00	38.13	0.00	8.41	25.00	31.79
06 - Worst	300 - 400	15.00	1579.31	10.00	952.04	14.00	1739.23	3.00	1371.72	3.00	651.88	4.00	1652.05
Freq.		1	1	3	1	3	1	3	1	3	1	2	1
% to SA+LNS [29]	1800	25.00	22.22	0.00	14.85	16.67	25.34	0.00	34.71	0.00	-4.97	33.33	33.28
07 - Worst	300 - 400	12.00	1423.57	10.00	1006.87	13.00	1644.68	3.00	1311.04	3.00	674.86	4.00	1667.84
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	9.09	32.78	0.00	21.46	18.18	33.47	0.00	51.29	0.00	-1.59	33.33	47.55
08 - Worst	300 - 400	11.00	1303.03	10.00	989.99	12.00	1475.39	3.00	1070.94	3.00	662.41	3.00	1288.31
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to SA+LNS [29]	1800	10.00	35.54	0.00	19.43	20.00	23.62	50.00	38.49	0.00	12.59	0.00	40.84
09 - Worst	300 - 400	14.00	1506.18	10.00	1051.26	-	-	3.00	1344.93	-	-	-	-
Freq.		2	1	3	1	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	16.67	29.15	0.00	26.82	-	-	0.00	37.22	-	-	-	-
10 - Worst	300 - 400	13.00	1446.66	-	-	-	-	3.00	1474.88	-	-	-	-
Freq.		1	1	-	-	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	18.18	29.83	-	-	-	-	0.00	46.52	-	-	-	-
11 - Worst	300 - 400	13.00	1478.30	-	-	-	-	3.00	1195.06	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to SA+LNS [29]	1800	18.18	39.03	-	-	-	-	0.00	47.62	-	-	-	-
12 - Worst	300 - 400	11.00	1277.02	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to SA+LNS [29]	1800	10.00	29.42	-	-	-	-	-	-	-	-	-	-
DACS 01 - AVGs	300 - 400	14.33	1536.44	10.00	1003.37	13.88	1725.45	3.18	1379.82	3.00	714.31	4.00	1617.08
% to SA+LNS [29]	1800	13.94	27.80	0.00	21.12	16.79	25.56	2.97	40.20	0.00	8.44	18.34	36.28
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table B.3: Five different DACS algorithms that are tested for three runs each on the problem set R1-100 of the problem group PG100 and are compared with the system DACS 01.

	Initial. Way	V-1 & V		V-1 & V		V-1 & V		V-1 & V		V-1 & V	
	Evap. Param. ρ	0.0		0.1		0.5		0.8		1.0	
PNo. (R1 set)	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	300 - 400	20.67	2052.92	20.00	2022.65	19.67	2079.44	19.33	1962.47	19.67	2052.34
SDs		0.58	42.14	0.00	56.79	0.58	67.64	0.58	46.21	0.58	46.17
% to DACS 01	300 - 400	0.00	8.36	-3.23	6.76	-4.84	9.76	-6.45	3.58	-4.84	8.33
02 - AVGs	300 - 400	19.00	1907.44	18.00	2018.83	18.00	1951.69	18.00	1992.71	18.00	2039.98
SDs		0.00	64.90	0.00	108.33	1.00	27.89	0.00	39.07	0.00	114.80
% to DACS 01	300 - 400	1.79	1.64	-3.57	7.58	-3.57	4.00	-3.57	6.18	-3.57	8.70
03 - AVGs	300 - 400	14.33	1678.57	14.67	1679.87	15.67	1810.72	15.67	1727.04	15.67	1746.38
SDs		0.58	29.94	0.58	19.93	0.58	95.62	0.58	59.99	0.58	67.83
% to DACS 01	300 - 400	-4.44	4.12	-2.22	4.20	4.44	12.32	4.44	7.13	4.44	8.32
04 - AVGs	300 - 400	11.00	1334.84	12.00	1422.39	11.33	1391.18	11.67	1424.94	11.67	1440.09
SDs		0.00	20.97	0.00	75.30	0.58	93.10	0.58	97.32	0.58	88.19
% to DACS 01	300 - 400	-8.33	1.27	0.00	7.91	-5.56	5.54	-2.78	8.11	-2.78	9.25
05 - AVGs	300 - 400	16.00	1786.09	16.00	1758.26	16.00	1794.50	16.00	1817.51	16.33	1775.89
SDs		0.00	23.76	0.00	18.55	0.00	27.08	0.00	59.29	0.58	38.60
% to DACS 01	300 - 400	4.35	3.27	4.35	1.66	4.35	3.76	4.35	5.09	6.52	2.68
06 - AVGs	300 - 400	13.33	1647.83	14.33	1663.74	14.00	1698.19	14.67	1728.96	13.67	1690.30
SDs		0.58	44.46	0.58	40.77	0.00	75.62	0.58	124.02	0.58	90.07
% to DACS 01	300 - 400	-6.98	1.23	0.00	2.21	-2.33	4.33	2.33	6.22	-4.65	3.84
07 - AVGs	300 - 400	12.00	1460.00	12.00	1499.04	12.67	1575.58	12.67	1555.49	12.33	1540.36
SDs		0.00	8.67	0.00	9.89	0.58	48.30	0.58	57.60	0.58	84.26
% to DACS 01	300 - 400	0.00	5.12	0.00	7.93	5.56	13.45	5.56	12.00	2.78	10.91
08 - AVGs	300 - 400	11.00	1313.79	11.00	1377.20	11.33	1382.66	11.67	1417.20	11.00	1406.67
SDs		0.00	23.72	0.00	75.64	0.58	8.96	0.58	80.25	0.00	24.96
% to DACS 01	300 - 400	0.00	5.77	0.00	10.88	3.03	11.32	6.06	14.10	0.00	13.25
09 - AVGs	300 - 400	13.00	1596.19	14.00	1617.08	13.67	1644.72	13.67	1658.15	14.33	1696.65
SDs		0.00	60.68	0.00	53.66	0.58	86.41	0.58	86.54	0.58	43.93
% to DACS 01	300 - 400	-4.88	4.48	2.44	5.85	0.00	7.65	0.00	8.53	4.88	11.05
10 - AVGs	300 - 400	12.00	1486.61	12.00	1547.34	13.00	1613.11	13.00	1599.67	12.67	1589.29
SDs		0.00	42.77	0.00	30.61	0.00	23.30	0.00	59.36	0.58	72.77
% to DACS 01	300 - 400	-2.70	7.20	-2.70	11.58	5.41	16.32	5.41	15.36	2.70	14.61
11 - AVGs	300 - 400	12.00	1493.39	12.00	1524.96	12.67	1626.06	13.00	1564.43	12.33	1531.08
SDs		0.00	17.59	0.00	20.27	0.58	44.40	0.00	35.55	0.58	49.83
% to DACS 01	300 - 400	-7.69	4.86	-7.69	7.08	-2.56	14.18	0.00	9.85	-5.13	7.51
12 - AVGs	300 - 400	10.67	1262.67	11.00	1361.11	11.33	1385.18	11.33	1400.09	12.00	1476.02
SDs		0.58	36.58	0.00	31.41	0.58	23.79	0.58	78.30	0.00	46.45
% to DACS 01	300 - 400	-3.03	1.39	0.00	9.29	3.03	11.22	3.03	12.42	9.09	18.52
AVGs	300 - 400	13.75	1585.03	13.92	1624.37	14.11	1662.75	14.22	1654.06	14.14	1665.42
SDs		0.14	3.78	0.00	18.33	0.05	18.63	0.19	6.51	0.05	4.18
% to DACS 01	300 - 400	-2.37	4.09	-1.18	6.67	0.20	9.19	0.99	8.62	0.39	9.36
DACS 01 - AVGs	300 - 400	14.08	1522.81								
SDs as in Table 4.2		0.08	12.57								

Table B.4: Five different DACS algorithms that are tested for three runs each on the problem set R1-200 created by the author and are compared with the system DACS 01.

	Initial. Way	V-1 & V		V-1 & V		V-1 & V		V-1 & V		V-1 & V	
	Evap. Param. ρ	0.0		0.1		0.5		0.8		1.0	
PNo. (R1 set)	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	600 - 800	34.00	3268.17	34.33	3260.90	32.00	3075.57	32.33	3166.41	32.67	3360.55
SDs		0.00	42.65	0.58	21.26	0.00	128.95	0.58	90.52	0.58	159.96
% to DACS 01	600 - 800	0.00	9.34	0.98	9.10	-5.88	2.90	-4.90	5.94	-3.92	12.43
02 - AVGs	600 - 800	21.67	2354.50	22.67	2304.85	22.67	2463.80	22.67	2478.42	23.00	2468.76
SDs		0.58	107.05	0.58	74.04	0.58	77.11	0.58	75.14	0.00	42.21
% to DACS 01	600 - 800	-2.99	13.19	1.49	10.80	1.49	18.45	1.49	19.15	2.99	18.68
03 - AVGs	600 - 800	25.33	2591.26	25.67	2563.75	26.00	2626.60	25.67	2588.87	26.67	2721.31
SDs		0.58	24.04	0.58	23.85	0.00	65.21	0.58	71.75	0.58	61.64
% to DACS 01	600 - 800	1.33	4.28	2.67	3.17	4.00	5.70	2.67	4.18	6.67	9.51
04 - AVGs	600 - 800	19.00	1992.64	19.00	1982.18	19.00	2001.65	20.00	2118.48	20.00	2152.78
SDs		0.00	25.98	0.00	51.08	0.00	22.65	0.00	45.65	1.00	114.50
% to DACS 01	600 - 800	1.79	8.04	1.79	7.47	1.79	8.52	7.14	14.86	7.14	16.72
05 - AVGs	600 - 800	20.00	2081.04	20.67	2159.80	21.00	2292.81	20.67	2240.87	20.67	2172.16
SDs		0.00	37.01	0.58	22.11	0.00	89.63	0.58	14.81	0.58	28.98
% to DACS 01	600 - 800	-4.76	-3.28	-1.59	0.39	0.00	6.57	-1.59	4.15	-1.59	0.96
06 - AVGs	600 - 800	19.00	1958.72	19.00	2009.83	19.67	2064.39	19.67	2104.77	20.00	2183.41
SDs		0.00	16.12	0.00	21.41	0.58	41.90	0.58	74.54	0.00	82.01
% to DACS 01	600 - 800	-3.39	7.24	-3.39	10.04	0.00	13.03	0.00	15.24	1.69	19.54
AVGs	600 - 800	23.17	2374.39	23.56	2380.22	23.39	2420.80	23.50	2449.64	23.83	2509.83
SDs		0.00	15.81	0.19	24.28	0.10	41.56	0.29	40.24	0.17	31.55
% to DACS 01	600 - 800	-1.18	6.50	0.47	6.76	-0.24	8.59	0.24	9.88	1.66	12.58

Problem set R1	Time(secs.)	NV	TD
DACS 01 - AVGs	600-800	23.44	2229.40
SDs		0.25	40.38
% to DACS 01	600-800	0.00	0.00

	PNo. (R1 set)	01		02		03		04		05		06	
DACS 01	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
AVGs	600-800	34.00	2988.94	22.33	2080.11	25.00	2484.94	18.67	1844.43	21.00	2151.51	19.67	1826.47
SDs		0.00	19.26	0.58	122.83	0.00	116.86	0.58	30.89	0.00	21.72	0.58	45.11
% to DACS 01	600-800	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table B.5: Five different DACS algorithms that are tested for three runs each on the problem set R1 of the problem group PG100 and are compared with the system DACS 01.

	Initial. Way	$1/(n.J_{\Psi}^{gb})$		=		=		=		=	
	Evap. Param. ρ	0.0		0.1		0.5		0.8		1.0	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	300 - 400	20.00	2012.99	20.00	2022.10	20.00	2000.23	19.67	1931.05	20.33	1905.36
SDs		0.00	47.55	0.00	113.41	0.00	68.74	0.58	28.82	0.58	39.04
% to DACS 01	300 - 400	-3.23	6.25	-3.23	6.73	-3.23	5.58	-4.84	1.92	-1.61	0.57
02 - AVGs	300 - 400	19.00	1846.65	18.33	1774.59	18.67	1801.31	19.00	1778.42	18.33	1852.67
SDs		0.00	35.53	0.58	34.92	0.58	70.10	0.00	4.71	0.58	92.80
% to DACS 01	300 - 400	1.79	-1.60	-1.79	-5.44	0.00	-4.01	1.79	-5.23	-1.79	-1.28
03 - AVGs	300 - 400	14.00	1664.75	14.00	1586.37	14.00	1547.91	14.00	1575.41	13.67	1666.66
SDs		0.00	100.83	0.00	66.92	0.00	44.50	0.00	46.90	0.58	95.67
% to DACS 01	300 - 400	-6.67	3.26	-6.67	-1.60	-6.67	-3.99	-6.67	-2.28	-8.89	3.38
04 - AVGs	300 - 400	11.00	1328.44	11.00	1219.13	11.00	1266.44	11.33	1315.07	11.33	1319.06
SDs		0.00	39.83	0.00	66.54	0.00	38.93	0.58	14.84	0.58	43.00
% to DACS 01	300 - 400	-8.33	0.78	-8.33	-7.51	-8.33	-3.92	-5.56	-0.23	-5.56	0.07
05 - AVGs	300 - 400	15.33	1778.27	15.67	1724.67	15.67	1712.89	15.67	1713.41	15.33	1721.84
SDs		0.58	72.69	0.58	27.56	0.58	36.01	0.58	49.39	0.58	33.32
% to DACS 01	300 - 400	0.00	2.82	2.17	-0.28	2.17	-0.96	2.17	-0.93	0.00	-0.44
06 - AVGs	300 - 400	13.67	1644.00	13.33	1593.41	13.33	1569.67	13.00	1623.22	13.33	1602.76
SDs		0.58	26.47	0.58	32.05	0.58	61.98	0.00	89.47	0.58	37.46
% to DACS 01	300 - 400	-4.65	1.00	-6.98	-2.11	-6.98	-3.57	-9.30	-0.28	-6.98	-1.54
07 - AVGs	300 - 400	12.00	1445.91	11.67	1386.03	12.00	1395.46	12.00	1452.19	12.00	1443.52
SDs		0.00	14.91	0.58	71.88	0.00	25.84	0.00	17.10	0.00	19.74
% to DACS 01	300 - 400	0.00	4.11	-2.78	-0.20	0.00	0.48	0.00	4.56	0.00	3.94
08 - AVGs	300 - 400	11.00	1318.59	10.67	1221.16	11.00	1279.80	11.00	1266.26	11.00	1296.94
SDs		0.00	24.47	0.58	56.07	0.00	32.56	0.00	8.30	0.00	14.31
% to DACS 01	300 - 400	0.00	6.16	-3.03	-1.68	0.00	3.04	0.00	1.95	0.00	4.42
09 - AVGs	300 - 400	13.00	1528.33	13.00	1571.96	13.00	1508.05	13.33	1550.30	13.00	1578.89
SDs		0.00	14.78	0.00	95.21	0.00	40.28	0.58	27.50	0.00	67.42
% to DACS 01	300 - 400	-4.88	0.04	-4.88	2.89	-4.88	-1.29	-2.44	1.47	-4.88	3.35
10 - AVGs	300 - 400	12.00	1503.46	11.67	1431.03	12.00	1469.67	12.00	1496.04	12.00	1467.99
SDs		0.00	21.14	0.58	16.49	0.00	73.47	0.00	39.10	0.00	9.61
% to DACS 01	300 - 400	-2.70	8.42	-5.41	3.19	-2.70	5.98	-2.70	7.88	-2.70	5.86
11 - AVGs	300 - 400	12.00	1488.83	12.00	1443.26	12.00	1458.51	12.00	1459.42	12.00	1509.00
SDs		0.00	14.17	0.00	25.55	0.00	28.96	0.00	68.73	0.00	31.05
% to DACS 01	300 - 400	-7.69	4.54	-7.69	1.34	-7.69	2.41	-7.69	2.48	-7.69	5.96
12 - AVGs	300 - 400	11.00	1295.58	10.67	1284.32	11.00	1300.95	11.00	1301.66	11.00	1308.20
SDs		0.00	9.90	0.58	37.75	0.00	5.56	0.00	23.98	0.00	55.39
% to DACS 01	300 - 400	0.00	4.03	-3.03	3.12	0.00	4.46	0.00	4.52	0.00	5.04
AVGs	300 - 400	13.67	1571.32	13.50	1521.50	13.64	1525.91	13.67	1538.54	13.61	1556.07
SDs		0.00	8.18	0.17	10.57	0.10	14.57	0.00	5.62	0.05	10.92
% to DACS 01	300 - 400	-2.96	3.19	-4.14	-0.09	-3.16	0.20	-2.96	1.03	-3.35	2.18
DACS 01 - AVGs	300 - 400	14.08	1522.81								
SDs as in Table 4.2		0.08	12.57								

Table B.6: A DACS system with reconfigured cycles, which is tested for three runs on the problem set R1 of the problem group PG100 and compared with the system DACS 01.

PNo.	Time(secs.)	NV	TD	NV	TD
01 - AVGs	300 - 400	20.00	1941.16	20.00	2022.10
SDs		0.00	61.43	0.00	113.41
% to DACS 01	300 - 400	0.00	-4.00	0.00	0.00
02 - AVGs	300 - 400	18.33	1849.77	18.33	1774.59
SDs		0.58	185.68	0.58	34.92
% to DACS 01	300 - 400	0.00	4.24	0.00	0.00
03 - AVGs	300 - 400	14.00	1533.23	14.00	1586.37
SDs		0.00	34.64	0.00	66.92
% to DACS 01	300 - 400	0.00	-3.35	0.00	0.00
04 - AVGs	300 - 400	11.33	1267.68	11.00	1219.13
SDs		0.58	32.99	0.00	66.54
% to DACS 01	300 - 400	3.03	3.98	0.00	0.00
05 - AVGs	300 - 400	15.00	1715.29	15.67	1724.67
SDs		0.00	95.45	0.58	27.56
% to DACS 01	300 - 400	-4.26	-0.54	0.00	0.00
06 - AVGs	300 - 400	13.33	1566.34	13.33	1593.41
SDs		0.58	23.85	0.58	32.06
% to DACS 01	300 - 400	0.00	-1.70	0.00	0.00
07 - AVGs	300 - 400	11.67	1401.70	11.67	1386.03
SDs		0.58	64.63	0.58	71.88
% to DACS 01	300 - 400	0.00	1.13	0.00	0.00
08 - AVGs	300 - 400	10.33	1220.75	10.67	1221.16
SDs		0.58	56.20	0.58	56.07
% to DACS 01	300 - 400	-3.12	-0.03	0.00	0.00
09 - AVGs	300 - 400	12.67	1513.45	13.00	1571.96
SDs		0.58	63.86	0.00	95.21
% to DACS 01	300 - 400	-2.56	-3.72	0.00	0.00
10 - AVGs	300 - 400	12.00	1397.47	11.67	1431.03
SDs		0.00	15.06	0.58	16.49
% to DACS 01	300 - 400	2.86	-2.35	0.00	0.00
11 - AVGs	300 - 400	12.00	1434.86	12.00	1443.26
SDs		0.00	29.22	0.00	25.55
% to DACS 01	300 - 400	0.00	-0.58	0.00	0.00
12 - AVGs	300 - 400	11.00	1228.15	10.67	1284.32
SDs		0.00	48.50	0.58	37.75
% to DACS 01	300 - 400	3.13	-4.37	0.00	0.00
AVGs	300 - 400	13.47	1505.82	13.50	1521.50
SDs		0.05	9.93	0.17	10.57
% to DACS 01	300 - 400	-0.21	-1.03	0.00	0.00

Table B.7: A Threaded-DACS system, which is tested for nine runs on the problem set R1 of the problem group PG100 and compared with the system DACS 01.

		R1		R1	
PNo.	Time(secs.)	NV	TD	NV	TD
01 - AVGs	300 - 400	19.89	1947.06	19.56	1936.73
SDs		0.60	51.98	0.53	33.87
% to DACS 01	300 - 400	1.70	0.53	0.00	0.00
02 - AVGs	300 - 400	18.44	1767.32	18.33	1777.14
SDs		0.53	83.21	0.50	69.01
% to DACS 01	300 - 400	0.61	-0.55	0.00	0.00
03 - AVGs	300 - 400	13.56	1492.43	13.78	1554.11
SDs		0.53	28.61	0.44	58.11
% to DACS 01	300 - 400	-1.61	-3.97	0.00	0.00
04 - AVGs	300 - 400	11.00	1255.03	11.00	1250.88
SDs		0.00	27.05	0.00	25.65
% to DACS 01	300 - 400	0.00	0.33	0.00	0.00
05 - AVGs	300 - 400	15.00	1688.78	15.11	1679.84
SDs		0.00	53.71	0.33	35.12
% to DACS 01	300 - 400	-0.74	0.53	0.00	0.00
06 - AVGs	300 - 400	13.00	1530.16	13.11	1577.46
SDs		0.00	35.65	0.33	24.32
% to DACS 01	300 - 400	-0.85	-3.00	0.00	0.00
07 - AVGs	300 - 400	11.67	1398.98	12.00	1396.94
SDs		0.50	31.93	0.00	32.33
% to DACS 01	300 - 400	-2.78	0.15	0.00	0.00
08 - AVGs	300 - 400	10.89	1224.37	10.78	1246.46
SDs		0.33	23.73	0.44	26.48
% to DACS 01	300 - 400	1.03	-1.77	0.00	0.00
09 - AVGs	300 - 400	12.67	1452.50	12.67	1481.91
SDs		0.50	39.56	0.50	56.03
% to DACS 01	300 - 400	0.00	-1.98	0.00	0.00
10 - AVGs	300 - 400	11.89	1384.42	12.00	1420.32
SDs		0.33	42.23	0.00	29.59
% to DACS 01	300 - 400	-0.93	-2.53	0.00	0.00
11 - AVGs	300 - 400	11.89	1414.32	11.89	1404.99
SDs		0.33	32.16	0.33	28.69
% to DACS 01	300 - 400	0.00	0.66	0.00	0.00
12 - AVGs	300 - 400	10.89	1244.17	11.00	1235.11
SDs		0.33	32.79	0.00	21.16
% to DACS 01	300 - 400	-1.01	0.73	0.00	0.00
AVGs	300 - 400	13.40	1483.30	13.44	1496.62
SDs		0.15	13.67	0.09	13.81
% to DACS 01	300 - 400	-0.28	-0.90	0.00	0.00

Table B.8: Five different DACS algorithms, which use different move operators, are tested for three runs each on the problem set R1 of the problem group PG100 and compared with the system DACS 01 that uses the XCHNG local search. The move operators used here are OrOpt1 (O1), OrOpt2 (O2), Relocate (REL), 2-Opt* (2O*), Swap (SW) and XCHNG (XCH).

	Move Op.	O1		O2		REL		2O*		SW		XCH	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	300 - 400	20.33	1951.89	20.00	1931.37	20.67	1939.07	20.00	1913.78	21.00	1920.32	20.00	1901.50
SDs		0.58	47.06	0.00	39.64	0.58	19.81	0.00	63.10	0.00	24.34	0.00	24.85
% to DACS 01	300 - 400	1.67	2.65	0.00	1.57	3.33	1.98	0.00	0.65	5.00	0.99	0.00	0.00
02 - AVGs	300 - 400	18.67	1798.88	18.67	1761.29	19.00	1793.49	19.00	1696.97	19.33	1773.99	18.67	1767.43
SDs		0.58	72.09	0.58	20.51	1.00	87.93	0.00	12.42	0.58	30.57	0.58	48.87
% to DACS 01	300 - 400	0.00	1.78	0.00	-0.35	1.79	1.47	1.79	-3.99	3.57	0.37	0.00	0.00
03 - AVGs	300 - 400	15.00	1517.49	14.33	1494.67	14.33	1552.35	14.00	1501.49	14.00	1519.02	14.00	1519.46
SDs		0.00	19.57	0.58	51.31	0.58	41.10	0.00	34.26	0.00	67.88	0.00	37.56
% to DACS 01	300 - 400	7.14	-0.13	2.38	-1.63	2.38	2.16	0.00	-1.18	0.00	-0.03	0.00	0.00
04 - AVGs	300 - 400	11.00	1291.24	11.33	1290.03	11.67	1272.61	11.33	1250.41	11.33	1278.00	11.33	1309.32
SDs		0.00	53.99	0.58	34.96	0.58	30.86	0.58	38.11	0.58	8.37	0.58	45.58
% to DACS 01	300 - 400	-2.94	-1.38	0.00	-1.47	2.94	-2.80	0.00	-4.50	0.00	-2.39	0.00	0.00
05 - AVGs	300 - 400	15.33	1747.27	15.33	1679.10	15.67	1704.19	15.33	1658.63	15.33	1701.72	15.67	1690.69
SDs		0.58	37.19	0.58	52.31	0.58	48.60	0.58	22.78	0.58	34.32	0.58	22.65
% to DACS 01	300 - 400	-2.13	3.35	-2.13	-0.69	0.00	0.80	-2.13	-1.90	-2.13	0.65	0.00	0.00
06 - AVGs	300 - 400	13.33	1567.50	13.33	1595.76	13.67	1537.47	13.33	1530.95	13.33	1560.10	13.00	1526.36
SDs		0.58	5.42	0.58	38.00	0.58	38.39	0.58	51.82	0.58	17.43	0.00	54.08
% to DACS 01	300 - 400	2.56	2.70	2.56	4.55	5.13	0.73	2.56	0.30	2.56	2.21	0.00	0.00
07 - AVGs	300 - 400	12.00	1384.16	12.00	1382.95	12.00	1388.86	12.00	1388.67	12.00	1420.61	12.00	1356.24
SDs		0.00	48.36	0.00	7.44	0.00	23.51	0.00	18.37	0.00	18.43	0.00	17.70
% to DACS 01	300 - 400	0.00	2.06	0.00	1.97	0.00	2.41	0.00	2.39	0.00	4.75	0.00	0.00
08 - AVGs	300 - 400	11.00	1257.12	11.00	1210.53	11.00	1238.31	11.00	1224.82	11.00	1244.47	11.00	1278.89
SDs		0.00	43.83	0.00	16.70	0.00	22.17	0.00	31.97	0.00	36.89	0.00	29.98
% to DACS 01	300 - 400	0.00	-1.70	0.00	-5.35	0.00	-3.17	0.00	-4.23	0.00	-2.69	0.00	0.00
09 - AVGs	300 - 400	13.00	1482.71	13.00	1518.82	13.00	1513.29	13.00	1499.16	13.00	1489.53	13.00	1507.05
SDs		0.00	43.97	0.00	35.52	0.00	17.49	0.00	33.61	0.00	30.58	0.00	21.15
% to DACS 01	300 - 400	0.00	-1.62	0.00	0.78	0.00	0.41	0.00	-0.52	0.00	-1.16	0.00	0.00
10 - AVGs	300 - 400	12.00	1396.91	12.00	1412.29	12.00	1404.08	12.00	1388.02	12.00	1417.04	12.00	1423.57
SDs		0.00	36.82	0.00	40.96	0.00	29.33	0.00	16.30	0.00	8.99	0.00	14.98
% to DACS 01	300 - 400	0.00	-1.87	0.00	-0.79	0.00	-1.37	0.00	-2.50	0.00	-0.46	0.00	0.00
11 - AVGs	300 - 400	12.00	1393.74	12.00	1431.10	12.00	1407.39	12.00	1386.29	11.67	1392.25	12.00	1413.74
SDs		0.00	18.18	0.00	41.05	0.00	23.53	0.00	26.43	0.58	37.79	0.00	8.53
% to DACS 01	300 - 400	0.00	-1.42	0.00	1.23	0.00	-0.45	0.00	-1.94	-2.78	-1.52	0.00	0.00
12 - AVGs	300 - 400	11.00	1229.05	11.00	1271.77	11.00	1246.74	11.00	1261.07	11.00	1279.78	11.00	1274.26
SDs		0.00	26.33	0.00	47.23	0.00	36.68	0.00	40.05	0.00	9.29	0.00	9.25
% to DACS 01	300 - 400	0.00	-3.55	0.00	-0.20	0.00	-2.16	0.00	-1.03	0.00	0.43	0.00	0.00
AVGs	300 - 400	13.72	1501.50	13.67	1498.31	13.83	1499.82	13.67	1475.02	13.75	1499.74	13.64	1497.38
SDs		0.05	20.53	0.08	13.35	0.14	12.52	0.08	6.58	0.08	2.98	0.13	8.56
% to DACS 01	300 - 400	0.61	0.28	0.20	0.06	1.43	0.16	0.20	-1.49	0.81	0.16	0.00	0.00

Appendix C

Tables of results related to the system DACS 02

This section mentions in Tables C.1 to C.16 information about the experiments done to the system DACS 02. All tables, mentioned in B1 to B5, have the results, either on the problem groups PG100, PG200 and PG400 in Section 2.2 or the problem set R1 of PG100, after three runs of 300, 400, 1800, 2400 or 4800 seconds.

B1- Tables C.1 to C.10 represent, on the problem groups PG100, P200 and PG400, the best and worst case performances of the system DACS 02 as in Sections 4.6.1 and 4.6.2, which uses the local search of triple moves (M1 to M3).

B2- Table C.11 represents the average case performances of two DACS systems, which use the parallel and the sequential ants as in Section 4.6.3.

B3- Tables C.12 to C.14 represent the average case performances of two DACS systems, which use two different initialization techniques as in Section 4.6.4 - either the nearest neighborhood heuristic NN or the SI1-Like 01 insertion heuristic.

B4- Table C.15 represents the average case performances of two DACS systems, which use different types of candidate lists as in Section 4.6.5 - either distance or time oriented.

B5- Table C.16 represent the average case performances of three DACS systems, which use or not a different way of pheromone updating locally and globally and local search moves near the depots as in Section 4.6.6.

Table C.1: Comparison between the best case performances of DACS 02 and other VRPTW algorithms, after three runs of 300 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	300	19.00	1658.03	10.00	828.94	15.00	1684.73	4.00	1377.37	3.00	591.56	4.00	1565.96
Freq.		1	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.00	-13.75	0.00	-2.48	-6.25	-16.14	0.00	-21.04	0.00	0.89	0.00	-12.22
02 - Best	300	18.00	1503.47	10.00	828.94	14.00	1518.03	4.00	1219.92	3.00	591.56	4.00	1391.15
Freq.		3	1	3	1	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-19.55	0.00	-13.61	0.00	-15.74	0.00	-19.39	0.00	-17.43	0.00	-17.41
03 - Best	300	14.00	1248.17	10.00	887.96	12.00	1350.44	3.00	1105.39	3.00	591.17	3.00	1342.13
Freq.		3	1	3	1	3	1	3	1	3	1	1	1
% to DACS 01	300 - 400	-6.67	-22.26	0.00	-10.29	0.00	-11.54	0.00	-19.53	0.00	-22.22	0.00	-11.12
04 - Best	300	10.00	1037.92	10.00	865.09	11.00	1198.09	3.00	858.03	3.00	923.28	3.00	942.73
Freq.		2	1	3	1	3	1	3	1	1	1	3	1
% to DACS 01	300 - 400	-16.67	-20.58	0.00	-23.36	-8.33	-15.18	0.00	-22.22	0.00	14.70	0.00	-13.79
05 - Best	300	14.00	1438.62	10.00	828.94	15.00	1594.39	3.00	1137.77	3.00	588.88	4.00	1445.04
Freq.		1	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.67	-14.38	0.00	-2.81	-6.25	-9.46	0.00	-20.34	0.00	0.44	0.00	-23.33
06 - Best	300	13.00	1306.93	10.00	828.94	12.00	1410.89	3.00	1046.67	3.00	588.49	3.00	1278.66
Freq.		3	1	3	2	1	1	3	1	3	3	1	1
% to DACS 01	300 - 400	-7.14	-17.31	0.00	-6.25	-14.29	-16.09	0.00	-18.47	0.00	-8.07	0.00	-21.01
07 - Best	300	11.00	1136.82	10.00	828.94	12.00	1286.85	3.00	954.02	3.00	588.29	4.00	1174.79
Freq.		1	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-8.33	-14.54	0.00	-14.71	-7.69	-19.68	0.00	-22.17	0.00	-6.85	0.00	-24.74
08 - Best	300	10.00	1026.55	10.00	828.94	11.00	1162.01	3.00	810.76	3.00	588.32	3.00	970.17
Freq.		1	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-9.09	-12.53	0.00	-10.02	-8.33	-21.10	0.00	-20.86	0.00	-9.84	0.00	-22.37
09 - Best	300	12.00	1188.09	10.00	828.94	-	-	3.00	1034.54	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-25.41	0.00	-16.80	-	-	0.00	-17.93	-	-	-	-
10 - Best	300	12.00	1124.82	-	-	-	-	3.00	1047.79	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	0.00	-16.20	-	-	-	-	0.00	-23.86	-	-	-	-
11 - Best	300	12.00	1132.83	-	-	-	-	3.00	859.57	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-18.39	-	-	-	-	0.00	-27.53	-	-	-	-
12 - Best	300	11.00	1000.20	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-17.66	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	300	13.00	1233.54	10.00	839.51	12.75	1400.68	3.18	1041.08	3.00	631.44	3.50	1263.83
% to DACS 01	300 - 400	-6.02	-17.79	0.00	-11.68	-6.42	-15.54	0.00	-21.13	0.00	-6.08	0.00	-18.37
% to SA+LNS [29]	1800	6.82	2.47	0.00	1.34	9.63	1.57	16.55	6.20	0.00	7.05	7.69	9.05
DACS 01 - AVGs	300 - 400	13.83	1500.51	10.00	950.52	13.63	1658.35	3.18	1319.93	3.00	672.30	3.50	1548.26
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table C.2: Comparison between the worst case performances of DACS 02 and other VRPTW algorithms, after three runs of 300 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	300	20.00	1671.06	10.00	828.94	16.00	1709.48	4.00	1512.67	3.00	591.56	4.00	1744.26
Freq.		2	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-4.76	-11.36	0.00	-3.06	-5.88	-14.40	0.00	-15.29	0.00	0.27	-20.00	-13.16
02 - Worst	300	18.00	1530.42	10.00	986.80	14.00	1550.70	4.00	1276.75	3.00	591.56	4.00	1419.95
Freq.		3	1	3	1	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.26	-18.70	0.00	-0.74	-6.67	-15.11	0.00	-18.71	0.00	-25.95	0.00	-18.19
03 - Worst	300	14.00	1291.81	10.00	950.94	12.00	1384.85	3.00	1142.12	3.00	607.89	4.00	1229.80
Freq.		3	1	3	1	3	1	3	1	3	1	2	1
% to DACS 01	300 - 400	-6.67	-20.20	0.00	-16.68	0.00	-16.99	0.00	-20.35	0.00	-28.21	0.00	-20.81
04 - Worst	300	11.00	1052.03	10.00	929.27	11.00	1212.07	3.00	869.97	4.00	675.27	3.00	987.26
Freq.		1	1	3	1	3	1	3	1	2	1	3	1
% to DACS 01	300 - 400	-8.33	-21.04	0.00	-21.58	-8.33	-20.55	0.00	-24.53	33.33	-20.67	0.00	-17.14
05 - Worst	300	15.00	1471.87	10.00	828.94	15.00	1609.47	3.00	1167.18	3.00	588.88	4.00	1547.14
Freq.		2	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.25	-13.64	0.00	-3.00	-6.25	-16.45	0.00	-20.42	0.00	-7.76	-20.00	-15.90
06 - Worst	300	13.00	1310.34	10.00	872.30	13.00	1458.76	3.00	1077.82	3.00	588.49	4.00	1245.34
Freq.		3	1	3	1	2	1	3	1	3	3	2	1
% to DACS 01	300 - 400	-13.33	-17.03	0.00	-8.38	-7.14	-16.13	0.00	-21.43	0.00	-9.72	0.00	-24.62
07 - Worst	300	12.00	1159.05	10.00	828.94	12.00	1317.62	3.00	973.89	3.00	588.29	4.00	1245.55
Freq.		2	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-18.58	0.00	-17.67	-7.69	-19.89	0.00	-25.72	0.00	-12.83	0.00	-25.32
08 - Worst	300	11.00	1006.01	10.00	828.94	11.00	1203.02	3.00	822.22	3.00	588.32	3.00	990.02
Freq.		2	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-22.79	0.00	-16.27	-8.33	-18.46	0.00	-23.22	0.00	-11.18	0.00	-23.15
09 - Worst	300	12.00	1277.55	10.00	828.94	-	-	3.00	1104.06	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-14.29	-15.18	0.00	-21.15	-	-	0.00	-17.91	-	-	-	-
10 - Worst	300	12.00	1166.78	-	-	-	-	3.00	1069.07	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-19.35	-	-	-	-	0.00	-27.51	-	-	-	-
11 - Worst	300	12.00	1154.86	-	-	-	-	3.00	933.57	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-21.88	-	-	-	-	0.00	-21.88	-	-	-	-
12 - Worst	300	11.00	1024.35	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-19.79	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	300	13.42	1259.68	10.00	876.00	13.00	1430.75	3.18	1086.30	3.13	602.53	3.75	1301.17
% to DACS 01	300 - 400	-6.40	-18.01	0.00	-12.69	-6.31	-17.08	0.00	-21.27	4.17	-15.65	-6.25	-19.54
% to SA+LNS [29]	1800	6.65	4.78	0.00	5.75	9.43	4.11	2.97	10.38	4.17	-8.53	10.95	9.66
DACS 01 - AVGs	300 - 400	14.33	1536.44	10.00	1003.37	13.88	1725.45	3.18	1379.82	3.00	714.31	4.00	1617.08
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table C.3: Comparison between the best case performances of DACS 02 and other VRPTW algorithms, after three runs of 400 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	400	19.00	1679.57	10.00	828.94	15.00	1684.81	4.00	1331.23	3.00	591.56	4.00	1518.90
Freq.		1	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.00	-12.62	0.00	-2.48	-6.25	-16.14	0.00	-23.68	0.00	0.89	0.00	-14.85
02 - Best	400	18.00	1489.85	10.00	828.94	14.00	1508.05	4.00	1210.94	3.00	591.56	4.00	1345.69
Freq.		3	1	3	2	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-20.28	0.00	-13.61	0.00	-16.30	0.00	-19.98	0.00	-17.43	0.00	-20.11
03 - Best	400	14.00	1255.12	10.00	888.19	12.00	1348.98	3.00	1037.77	3.00	591.17	3.00	1236.10
Freq.		3	1	3	1	3	1	3	1	3	3	1	1
% to DACS 01	300 - 400	-6.67	-21.83	0.00	-10.27	0.00	-11.63	0.00	-24.45	0.00	-22.22	0.00	-18.14
04 - Best	400	10.00	1031.91	10.00	834.47	11.00	1209.87	3.00	863.36	3.00	604.66	3.00	909.97
Freq.		2	1	3	1	3	1	3	1	1	1	3	1
% to DACS 01	300 - 400	-16.67	-21.04	0.00	-26.07	-8.33	-14.34	0.00	-21.74	0.00	-24.89	0.00	-16.78
05 - Best	400	15.00	1402.51	10.00	828.94	15.00	1607.76	3.00	1114.67	3.00	588.88	4.00	1430.12
Freq.		3	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-16.53	0.00	-2.81	-6.25	-8.70	0.00	-21.95	0.00	0.44	0.00	-24.12
06 - Best	400	13.00	1297.13	10.00	828.94	12.00	1426.92	3.00	1057.20	3.00	588.49	4.00	1249.92
Freq.		3	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-7.14	-17.93	0.00	-6.25	-14.29	-15.14	0.00	-17.64	0.00	-8.07	33.33	-22.79
07 - Best	400	11.00	1104.42	10.00	828.94	11.00	1305.79	3.00	918.81	3.00	588.29	4.00	1142.56
Freq.		1	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-8.33	-16.97	0.00	-14.71	-15.38	-18.49	0.00	-25.04	0.00	-6.85	0.00	-26.81
08 - Best	400	10.00	1056.81	10.00	828.94	11.00	1141.03	3.00	788.26	3.00	588.32	3.00	904.13
Freq.		2	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-9.09	-9.95	0.00	-10.02	-8.33	-22.52	0.00	-23.06	0.00	-9.84	0.00	-27.66
09 - Best	400	13.00	1208.87	10.00	828.94	-	-	3.00	993.66	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	0.00	-24.11	0.00	-16.80	-	-	0.00	-21.17	-	-	-	-
10 - Best	400	11.00	1205.23	-	-	-	-	3.00	1028.13	-	-	-	-
Freq.		1	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-8.33	-10.21	-	-	-	-	0.00	-25.29	-	-	-	-
11 - Best	400	12.00	1122.48	-	-	-	-	3.00	884.14	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-19.13	-	-	-	-	0.00	-25.46	-	-	-	-
12 - Best	400	11.00	990.37	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-18.47	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	400	13.08	1237.02	10.00	836.14	12.63	1404.15	3.18	1020.74	3.00	591.62	3.63	1217.17
% to DACS 01	300 - 400	-5.42	-17.56	0.00	-12.03	-7.34	-15.33	0.00	-22.67	0.00	-12.00	3.57	-21.38
% to SA+LNS [29]	1800	7.50	2.76	0.00	0.94	8.56	1.82	16.55	4.12	0.00	0.30	11.54	5.03
DACS 01 - AVGs	300 - 400	13.83	1500.51	10.00	950.52	13.63	1658.35	3.18	1319.93	3.00	672.30	3.50	1548.26
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table C.4: Comparison between the worst case performances of DACS 02 and other VRPTW algorithms, after three runs of 400 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	400	20.00	1677.74	10.00	828.94	16.00	1738.81	4.00	1351.95	3.00	591.56	4.00	1744.55
Freq.		2	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-4.76	-11.01	0.00	-3.06	-5.88	-12.94	0.00	-24.29	0.00	0.27	-20.00	-13.15
02 - Worst	400	18.00	1518.20	10.00	871.66	14.00	1517.20	4.00	1277.27	3.00	591.56	4.00	1435.03
Freq.		3	1	3	1	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.26	-19.35	0.00	-12.32	-6.67	-16.95	0.00	-18.68	0.00	-25.95	0.00	-17.32
03 - Worst	400	14.00	1310.53	10.00	944.06	12.00	1393.53	3.00	1120.77	3.00	591.17	4.00	1200.70
Freq.		3	1	3	1	3	1	3	1	3	3	2	1
% to DACS 01	300 - 400	-6.67	-19.04	0.00	-17.28	0.00	-16.47	0.00	-21.84	0.00	-30.19	0.00	-22.68
04 - Worst	400	11.00	1060.49	10.00	891.72	11.00	1318.20	3.00	878.55	4.00	707.19	3.00	951.58
Freq.		1	1	3	1	3	1	3	1	2	1	3	1
% to DACS 01	300 - 400	-8.33	-20.40	0.00	-24.75	-8.33	-13.60	0.00	-23.78	33.33	-16.93	0.00	-20.13
05 - Worst	400	15.00	1481.71	10.00	828.94	15.00	1629.19	3.00	1142.88	3.00	588.88	4.00	1479.35
Freq.		3	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.25	-13.06	0.00	-3.00	-6.25	-15.43	0.00	-22.07	0.00	-7.76	-20.00	-19.58
06 - Worst	400	13.00	1332.88	10.00	828.94	13.00	1428.70	3.00	1077.87	3.00	588.49	4.00	1264.05
Freq.		3	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-13.33	-15.60	0.00	-12.93	-7.14	-17.85	0.00	-21.42	0.00	-9.72	0.00	-23.49
07 - Worst	400	12.00	1133.62	10.00	828.94	13.00	1296.57	3.00	956.84	3.00	588.29	4.00	1180.24
Freq.		2	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-20.37	0.00	-17.67	0.00	-21.17	0.00	-27.02	0.00	-12.83	0.00	-29.24
08 - Worst	400	11.00	995.38	10.00	828.94	11.00	1256.41	3.00	811.12	3.00	588.32	3.00	1001.79
Freq.		1	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	0.00	-23.61	0.00	-16.27	-8.33	-14.84	0.00	-24.26	0.00	-11.18	0.00	-22.24
09 - Worst	400	13.00	1237.84	10.00	828.94	-	-	3.00	1013.15	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.14	-17.82	0.00	-21.15	-	-	0.00	-24.67	-	-	-	-
10 - Worst	400	12.00	1188.20	-	-	-	-	3.00	1085.37	-	-	-	-
Freq.		2	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-17.87	-	-	-	-	0.00	-26.41	-	-	-	-
11 - Worst	400	12.00	1132.81	-	-	-	-	3.00	901.20	-	-	-	-
Freq.		3	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-23.37	-	-	-	-	0.00	-24.59	-	-	-	-
12 - Worst	400	11.00	1007.32	-	-	-	-	-	-	-	-	-	-
Freq.		3	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-21.12	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	400	13.50	1256.39	10.00	853.45	13.13	1447.33	3.18	1056.09	3.13	604.43	3.75	1282.16
% to DACS 01	300 - 400	-5.81	-18.23	0.00	-14.94	-5.41	-16.12	0.00	-23.46	4.17	-15.38	-6.25	-20.71
% to SA+LNS [29]	1800	7.31	4.50	0.00	3.03	10.48	5.32	2.97	7.31	4.17	-8.24	10.95	8.06
DACS 01 - AVGs	300 - 400	14.33	1536.44	10.00	1003.37	13.88	1725.45	3.18	1379.82	3.00	714.31	4.00	1617.08
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	656.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table C.5: Comparison between the best case performances of DACS 02 and other VRPTW algorithms, after three runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	1800	19.00	1690.49	10.00	828.94	15.00	1699.43	4.00	1271.71	3.00	591.56	4.00	1487.19
Freq.		2	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.00	-12.06	0.00	-2.48	-6.25	-15.41	0.00	-27.09	0.00	0.89	0.00	-16.63
02 - Best	1800	17.00	1498.89	10.00	828.94	13.00	1520.62	4.00	1105.41	3.00	591.56	4.00	1195.04
Freq.		2	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.56	-19.79	0.00	-13.61	-7.14	-15.60	0.00	-26.95	0.00	-17.43	0.00	-29.05
03 - Best	1800	14.00	1228.85	10.00	839.27	11.00	1366.70	3.00	999.66	3.00	591.17	3.00	1116.18
Freq.		3	1	3	1	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.67	-23.47	0.00	-15.21	-8.33	-10.47	0.00	-27.22	0.00	-22.22	0.00	-26.08
04 - Best	1800	11.00	1011.24	10.00	824.78	11.00	1175.64	3.00	813.18	3.00	593.93	3.00	863.02
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 01	300 - 400	-8.33	-22.62	0.00	-26.93	-8.33	-16.76	0.00	-26.29	0.00	-26.22	0.00	-21.07
05 - Best	1800	14.00	1460.61	10.00	828.94	14.00	1629.75	3.00	1043.21	3.00	588.88	4.00	1343.83
Freq.		1	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.67	-13.07	0.00	-2.81	-12.50	-7.45	0.00	-26.96	0.00	0.44	0.00	-28.70
06 - Best	1800	13.00	1294.24	10.00	828.94	12.00	1535.51	3.00	970.58	3.00	588.49	3.00	1227.91
Freq.		3	1	3	3	1	1	3	1	3	3	2	1
% to DACS 01	300 - 400	-7.14	-18.11	0.00	-6.25	-14.29	-8.68	0.00	-24.39	0.00	-8.07	0.00	-24.15
07 - Best	1800	11.00	1102.32	10.00	828.94	12.00	1245.95	3.00	903.96	3.00	588.29	4.00	1041.52
Freq.		2	1	3	3	3	1	3	1	3	2	3	1
% to DACS 01	300 - 400	-8.33	-17.13	0.00	-14.71	-7.69	-22.23	0.00	-26.25	0.00	-6.85	0.00	-33.28
08 - Best	1800	10.00	961.19	10.00	828.94	11.00	1149.96	3.00	738.85	3.00	588.32	3.00	849.96
Freq.		3	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-9.09	-18.10	0.00	-10.02	-8.33	-21.92	0.00	-27.88	0.00	-9.84	0.00	-31.99
09 - Best	1800	12.00	1175.85	10.00	828.94	-	-	3.00	955.87	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-26.18	0.00	-16.80	-	-	0.00	-24.17	-	-	-	-
10 - Best	1800	11.00	1120.88	-	-	-	-	3.00	996.22	-	-	-	-
Freq.		2	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-8.33	-16.50	-	-	-	-	0.00	-27.61	-	-	-	-
11 - Best	1800	11.00	1130.62	-	-	-	-	3.00	842.84	-	-	-	-
Freq.		1	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-15.38	-18.55	-	-	-	-	0.00	-28.94	-	-	-	-
12 - Best	1800	10.00	975.60	-	-	-	-	-	-	-	-	-	-
Freq.		2	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	-9.09	-19.69	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	1800	12.75	1220.90	10.00	829.63	12.38	1415.45	3.18	967.41	3.00	590.28	3.50	1140.58
% to DACS 01	300 - 400	-7.83	-18.63	0.00	-12.72	-9.17	-14.65	0.00	-26.71	0.00	-12.20	0.00	-26.33
% to SA+LNS [29]	1800	4.77	1.42	0.00	0.15	6.41	2.64	16.55	-1.32	0.00	0.07	7.69	-1.58
DACS 01 - AVGs	300 - 400	13.83	1500.51	10.00	950.52	13.63	1658.35	3.18	1319.93	3.00	672.30	3.50	1548.26
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table C.6: Comparison between the worst case performances of DACS 02 and other VRPTW algorithms, after three runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	1800	20.00	1661.79	10.00	828.94	16.00	1676.81	4.00	1317.01	3.00	591.56	4.00	1516.34
Freq.		1	1	3	3	1	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-4.76	-11.85	0.00	-3.06	-5.88	-16.04	0.00	-26.25	0.00	0.27	-20.00	-24.51
02 - Worst	1800	18.00	1484.40	10.00	828.94	14.00	1526.45	4.00	1145.55	3.00	591.56	4.00	1245.39
Freq.		1	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-5.26	-21.15	0.00	-16.62	-6.67	-16.44	0.00	-27.06	0.00	-25.95	0.00	-28.25
03 - Worst	1800	14.00	1241.41	10.00	897.01	12.00	1338.21	3.00	1019.21	3.00	591.17	3.00	1143.63
Freq.		3	1	3	1	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.67	-23.31	0.00	-21.41	0.00	-19.78	0.00	-28.92	0.00	-30.19	-25.00	-26.36
04 - Worst	1800	11.00	1025.91	10.00	865.31	11.00	1192.40	3.00	836.05	3.00	613.46	3.00	870.24
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 01	300 - 400	-8.33	-23.00	0.00	-26.98	-8.33	-21.84	0.00	-27.47	0.00	-27.94	0.00	-26.96
05 - Worst	1800	15.00	1428.25	10.00	828.94	15.00	1596.03	3.00	1082.64	3.00	588.88	4.00	1386.64
Freq.		2	1	3	3	2	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-6.25	-16.20	0.00	-3.00	-6.25	-17.15	0.00	-26.18	0.00	-7.76	-20.00	-24.62
06 - Worst	1800	13.00	1335.37	10.00	828.94	13.00	1427.48	3.00	1012.19	3.00	586.49	4.00	1137.48
Freq.		3	1	3	3	2	1	3	1	3	3	1	1
% to DACS 01	300 - 400	-13.33	-15.45	0.00	-12.93	-7.14	-17.92	0.00	-26.21	0.00	-9.72	0.00	-31.15
07 - Worst	1800	12.00	1134.79	10.00	828.94	12.00	1298.72	3.00	932.41	3.00	591.73	4.00	1097.31
Freq.		1	1	3	3	3	1	3	1	3	1	3	1
% to DACS 01	300 - 400	0.00	-20.29	0.00	-17.67	-7.69	-21.03	0.00	-28.88	0.00	-12.32	0.00	-34.21
08 - Worst	1800	10.00	989.65	10.00	828.94	11.00	1179.62	3.00	786.69	3.00	588.32	3.00	939.50
Freq.		3	1	3	3	3	1	3	1	3	3	3	1
% to DACS 01	300 - 400	-9.09	-24.05	0.00	-16.27	-8.33	-20.05	0.00	-26.54	0.00	-11.18	0.00	-27.08
09 - Worst	1800	12.00	1255.75	10.00	828.94	-	-	3.00	1006.63	-	-	-	-
Freq.		3	1	3	3	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-14.29	-16.63	0.00	-21.15	-	-	0.00	-25.15	-	-	-	-
10 - Worst	1800	12.00	1107.82	-	-	-	-	3.00	1016.56	-	-	-	-
Freq.		1	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-23.42	-	-	-	-	0.00	-31.07	-	-	-	-
11 - Worst	1800	12.00	1115.89	-	-	-	-	3.00	865.33	-	-	-	-
Freq.		2	1	-	-	-	-	3	1	-	-	-	-
% to DACS 01	300 - 400	-7.69	-24.52	-	-	-	-	0.00	-27.59	-	-	-	-
12 - Worst	1800	11.00	996.26	-	-	-	-	-	-	-	-	-	-
Freq.		1	1	-	-	-	-	-	-	-	-	-	-
% to DACS 01	300 - 400	0.00	-21.99	-	-	-	-	-	-	-	-	-	-
DACS 02 - AVGs	1800	13.33	1231.44	10.00	840.54	13.00	1404.47	3.18	1001.84	3.00	593.15	3.63	1167.07
% to DACS 01	300 - 400	-6.98	-19.65	0.00	-16.23	-6.31	-18.60	0.00	-27.39	0.00	-16.96	-9.38	-27.83
% to SA+LNS [29]	1800	5.99	2.43	0.00	1.47	9.43	2.20	2.97	1.80	0.00	-9.95	7.25	-1.64
DACS 01 - AVGs	300 - 400	14.33	1536.44	10.00	1003.37	13.88	1725.45	3.18	1379.82	3.00	714.31	4.00	1617.08
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table C.7: Comparison between the best case performances of DACS 02 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	2400	20.00	5679.12	20.00	2704.57	19.00	3920.21	5.00	4249.72	6.00	1955.91	7.00	3380.33
Freq.		2	1	3	2	3	1	3	1	3	2	3	1
% to RVNSc [5]	720 - 1680	5.26	13.03	0.00	0.00	5.56	-6.04	25.00	-12.46	0.00	1.27	16.67	6.71
02 - Best	2400	19.00	4361.07	19.00	3538.24	19.00	3584.89	5.00	3666.87	6.00	1907.66	5.00	3192.60
Freq.		3	1	2	1	3	1	3	1	3	1	1	1
% to RVNSc [5]	720 - 1680	5.56	2.99	5.56	12.66	5.56	-2.45	25.00	-4.10	0.00	2.39	0.00	11.12
03 - Best	2400	19.00	3816.81	19.00	3322.59	18.00	4284.60	4.00	3200.98	6.00	2130.87	5.00	2663.51
Freq.		3	1	3	1	3	1	3	1	1	1	3	1
% to RVNSc [5]	720 - 1680	5.56	7.44	5.56	17.28	0.00	30.45	0.00	5.86	0.00	17.82	25.00	-2.91
04 - Best	2400	18.00	3819.34	19.00	2923.96	18.00	3642.80	4.00	2292.07	7.00	1920.36	4.00	2495.23
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	20.19	5.56	8.45	0.00	19.66	0.00	13.42	16.67	9.44	0.00	18.21
05 - Best	2400	19.00	4489.63	20.00	2738.12	19.00	3704.27	4.00	3690.59	6.00	1895.45	5.00	3116.82
Freq.		3	1	3	1	3	1	3	1	3	1	2	1
% to RVNSc [5]	720 - 1680	5.56	0.57	0.00	1.33	5.56	-4.70	0.00	5.88	0.00	0.86	25.00	-7.78
06 - Best	2400	18.00	4442.06	20.00	2701.04	19.00	3761.96	4.00	3258.70	6.00	2018.27	5.00	2989.89
Freq.		1	1	2	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	16.14	0.00	0.00	5.56	1.52	0.00	8.29	0.00	8.66	0.00	12.14
07 - Best	2400	18.00	4189.96	20.00	2748.45	19.00	3647.88	4.00	2733.22	6.00	1931.46	4.00	3078.18
Freq.		3	1	3	1	3	1	3	1	2	1	1	1
% to RVNSc [5]	720 - 1680	0.00	25.80	0.00	1.76	5.56	3.98	0.00	7.86	0.00	4.40	0.00	13.84
08 - Best	2400	18.00	3566.34	20.00	2791.54	19.00	3471.89	4.00	2240.23	6.00	1903.59	4.00	2638.16
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	15.58	5.26	-0.30	5.56	3.89	0.00	20.68	0.00	4.44	0.00	11.22
09 - Best	2400	19.00	4348.12	19.00	2865.75	18.00	4413.33	4.00	3364.11	6.00	1936.10	4.00	2460.49
Freq.		3	1	3	1	1	1	3	1	1	1	3	1
% to RVNSc [5]	720 - 1680	5.56	7.99	5.56	3.26	0.00	35.22	0.00	7.29	0.00	4.76	0.00	10.26
10 - Best	2400	18.00	4106.62	18.00	3314.37	18.00	4050.59	4.00	2991.04	6.00	2029.11	4.00	2317.56
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	17.55	0.00	21.20	0.00	26.65	0.00	10.40	0.00	12.18	0.00	13.89
DACS 02 - AVGs	2400	18.60	4281.91	19.40	2964.86	18.60	3848.24	4.20	3168.75	6.10	1962.88	4.70	2833.28
% to RVNSc [5]	720 - 1680	2.76	12.05	2.65	6.70	3.33	9.70	5.00	4.05	1.67	6.54	6.82	7.80
% to ES4C [2]	2400	2.20	15.57	2.65	6.57	3.33	8.25	5.00	3.72	1.67	6.33	9.30	5.92
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table C.8: Comparison between the worst case performances of DACS 02 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	2400	21.00	5572.44	20.00	2732.73	19.00	4086.35	5.00	4443.65	6.00	1988.85	7.00	3496.20
Freq.		1	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	10.53	10.90	0.00	1.04	5.56	-2.06	25.00	-8.46	0.00	2.97	16.67	10.37
02 - Worst	2400	19.00	4684.42	20.00	3838.74	19.00	3907.62	5.00	3929.54	6.00	1918.32	6.00	3074.83
Freq.		3	1	1	1	3	1	3	1	3	1	2	1
% to RVNSc [5]	720 - 1680	5.56	10.63	11.11	22.23	5.56	6.34	25.00	2.77	0.00	2.96	20.00	7.02
03 - Worst	2400	19.00	4014.04	19.00	4011.31	18.00	4497.62	4.00	3420.10	7.00	1940.26	5.00	2874.29
Freq.		3	1	3	1	3	1	3	1	2	1	3	1
% to RVNSc [5]	720 - 1680	5.56	12.99	5.56	41.59	0.00	36.95	0.00	13.11	16.67	7.28	25.00	4.77
04 - Worst	2400	18.00	3861.59	19.00	3261.25	18.00	4001.67	4.00	2398.44	7.00	1961.67	4.00	2537.73
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	21.52	5.56	20.96	0.00	31.45	0.00	18.68	16.67	11.79	0.00	20.22
05 - Worst	2400	19.00	5088.24	20.00	2749.76	19.00	3896.25	4.00	3730.97	6.00	2027.65	6.00	3044.14
Freq.		3	1	3	1	3	1	3	1	3	1	1	1
% to RVNSc [5]	720 - 1680	5.56	13.98	0.00	1.77	5.56	0.24	0.00	7.04	0.00	7.89	50.00	-9.93
06 - Worst	2400	19.00	4031.01	21.00	3041.21	19.00	3965.21	4.00	3272.04	6.00	2217.54	5.00	3093.48
Freq.		2	1	1	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	5.56	5.39	5.00	12.59	5.56	7.01	0.00	8.73	0.00	19.39	0.00	16.02
07 - Worst	2400	18.00	4588.39	20.00	2995.17	19.00	3778.17	4.00	2798.41	7.00	1873.60	5.00	2971.20
Freq.		3	1	3	1	3	1	3	1	1	1	2	1
% to RVNSc [5]	720 - 1680	0.00	37.77	0.00	10.89	5.56	7.69	0.00	10.43	16.67	1.27	25.00	9.88
08 - Worst	2400	18.00	3613.45	20.00	2925.02	19.00	3517.68	4.00	2406.31	6.00	1966.64	4.00	2848.06
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	17.10	5.26	4.47	5.56	5.26	0.00	29.63	0.00	7.90	0.00	20.07
09 - Worst	2400	19.00	4449.73	19.00	3115.46	19.00	3543.77	4.00	3636.52	7.00	2114.63	4.00	2694.37
Freq.		3	1	3	1	2	1	3	1	2	1	3	1
% to RVNSc [5]	720 - 1680	5.56	10.51	5.56	12.26	5.56	8.57	0.00	15.98	16.67	14.42	0.00	20.74
10 - Worst	2400	18.00	4311.58	19.00	3009.01	18.00	4559.79	4.00	3106.56	6.00	2211.96	4.00	2363.46
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	720 - 1680	0.00	23.42	5.56	10.04	0.00	42.57	0.00	14.66	0.00	22.29	0.00	16.14
DACS 02 - AVGs	2400	18.80	4421.49	19.70	3167.97	18.70	3975.43	4.20	3314.25	6.40	2022.11	5.00	2899.78
% to RVNSc [5]	720 - 1680	3.87	15.70	4.23	14.00	3.89	13.32	5.00	8.83	6.67	9.75	13.64	10.33
% to ES4C [2]	2400	3.30	19.34	4.23	13.87	3.89	11.83	5.00	8.49	6.67	9.54	16.28	8.40
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table C.9: Comparison between the best case performances of DACS 02 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	4800	40.00	12758.74	40.00	7452.57	38.00	10261.83	9.00	10338.43	12.00	4280.52	12.00	8217.11
Freq.		1	1	3	1	1	1	2	1	3	1	1	1
% to RVNSc [5]	3900-7980	5.26	15.11	0.00	4.20	2.70	13.54	12.50	6.21	0.00	3.99	9.09	8.82
02 - Best	4800	38.00	11970.91	40.00	14907.02	38.00	10068.77	9.00	9353.09	12.00	4793.98	11.00	7918.58
Freq.		3	1	1	1	3	1	3	1	1	1	1	1
% to RVNSc [5]	3900-7980	5.56	18.94	8.11	96.15	5.56	11.62	12.50	17.04	0.00	20.27	10.00	24.56
03 - Best	4800	37.00	10468.32	38.00	11728.70	37.00	9468.83	8.00	7378.74	13.00	5067.06	9.00	6402.18
Freq.		1	1	3	1	3	1	3	1	3	1	2	1
% to RVNSc [5]	3900-7980	2.78	20.92	5.56	56.28	2.78	14.76	0.00	17.92	8.33	29.14	12.50	17.96
04 - Best	4800	37.00	8878.24	37.00	9439.35	36.00	11233.36	8.00	5926.27	13.00	4772.62	8.00	4971.57
Freq.		3	1	3	1	1	1	3	1	3	1	3	1
% to RVNSc [5]	3900-7980	2.78	15.62	2.78	29.35	0.00	45.00	0.00	29.06	8.33	24.36	0.00	31.88
05 - Best	4800	37.00	11693.51	40.00	7636.20	38.00	9734.30	8.00	8585.95	12.00	4426.17	12.00	7348.30
Freq.		1	1	2	1	3	1	3	1	1	1	3	1
% to RVNSc [5]	3900-7980	2.78	14.16	0.00	6.77	5.56	5.86	0.00	15.07	0.00	12.17	33.33	12.04
06 - Best	4800	37.00	10998.78	41.00	8755.12	37.00	10347.59	8.00	7759.71	13.00	5120.56	10.00	7482.91
Freq.		2	1	1	1	1	1	3	1	3	1	1	1
% to RVNSc [5]	3900-7980	2.78	17.26	2.50	22.39	2.78	15.30	0.00	18.76	8.33	31.80	11.11	24.06
07 - Best	4800	37.00	10163.97	40.00	8034.08	38.00	10169.39	8.00	6766.67	13.00	5121.69	10.00	6579.00
Freq.		3	1	1	1	3	1	3	1	2	1	3	1
% to RVNSc [5]	3900-7980	2.78	22.48	0.00	12.37	5.56	15.54	0.00	24.35	8.33	30.14	25.00	9.28
08 - Best	4800	36.00	10806.72	41.00	8438.18	37.00	9817.21	8.00	5399.37	12.00	4721.75	8.00	5994.38
Freq.		3	1	2	1	3	1	3	1	3	1	1	1
% to RVNSc [5]	3900-7980	0.00	41.80	7.89	14.33	2.78	14.56	0.00	24.52	0.00	22.77	0.00	16.50
09 - Best	4800	37.00	11272.49	39.00	8695.06	37.00	9921.60	8.00	7562.24	13.00	5125.68	8.00	5774.18
Freq.		2	1	3	1	3	1	3	1	2	1	3	1
% to RVNSc [5]	3900-7980	2.78	17.06	5.41	19.37	2.78	17.10	0.00	9.77	8.33	29.46	0.00	18.96
10 - Best	4800	37.00	10315.59	38.00	9335.90	37.00	9463.05	8.00	7292.23	12.00	4496.61	8.00	5504.05
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	3900-7980	2.78	16.06	5.56	23.77	2.78	15.17	0.00	16.83	0.00	18.59	0.00	19.20
DACS 02 - AVGs	4800	37.30	10932.73	39.40	9442.22	37.30	10048.59	8.20	7636.27	12.50	4792.68	9.60	6619.23
% to RVNSc [5]	3900-7980	3.04	19.42	3.68	28.96	3.32	16.45	2.50	16.62	4.17	22.18	10.34	17.50
% to ES4C [2]	4800	2.75	22.50	3.68	24.50	3.32	14.67	2.50	17.44	4.17	21.80	11.63	19.96
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table C.10: Comparison between the worst case performances of DACS 02 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	4800	42.00	13556.08	40.00	7590.24	40.00	9987.67	10.00	10648.29	12.00	4330.70	14.00	7883.97
Freq.		1	1	3	1	1	1	1	1	3	1	1	1
% to RVNSc [5]	3900-7980	10.53	22.30	0.00	6.13	8.11	10.50	25.00	9.39	0.00	5.21	27.27	4.41
02 - Worst	4800	38.00	13074.85	42.00	13091.75	38.00	10183.16	9.00	9715.24	13.00	5773.11	12.00	8360.20
Freq.		3	1	1	1	3	1	3	1	2	1	2	1
% to RVNSc [5]	3900-7980	5.56	29.91	13.51	72.26	5.56	12.89	12.50	21.58	8.33	44.83	20.00	31.51
03 - Worst	4800	38.00	10523.83	38.00	13266.88	37.00	10099.09	8.00	7895.78	13.00	6280.77	10.00	6342.79
Freq.		2	1	3	1	3	1	3	1	3	1	1	1
% to RVNSc [5]	3900-7980	5.56	21.56	5.56	76.78	2.78	22.40	0.00	26.18	8.33	60.07	25.00	16.87
04 - Worst	4800	37.00	9094.92	37.00	10101.36	37.00	9141.58	8.00	5972.67	13.00	5212.78	8.00	5154.40
Freq.		3	1	3	1	2	1	3	1	3	1	3	1
% to RVNSc [5]	3900-7980	2.78	18.44	2.78	38.43	2.78	18.00	0.00	30.07	8.33	35.83	0.00	36.73
05 - Worst	4800	38.00	12360.98	41.00	8316.07	38.00	10530.70	8.00	9025.34	13.00	4861.44	12.00	7411.69
Freq.		2	1	1	1	3	1	3	1	2	1	3	1
% to RVNSc [5]	3900-7980	5.56	20.68	2.50	16.28	5.56	14.52	0.00	20.96	8.33	23.20	33.33	13.01
06 - Worst	4800	38.00	11115.41	43.00	8798.84	39.00	10182.26	8.00	8128.77	13.00	5504.01	11.00	7157.99
Freq.		1	1	1	1	1	1	3	1	3	1	2	1
% to RVNSc [5]	3900-7980	5.56	18.51	7.50	23.00	8.33	13.46	0.00	24.41	8.33	41.67	22.22	18.68
07 - Worst	4800	37.00	10524.87	41.00	8890.62	38.00	10495.55	8.00	6926.74	14.00	5004.91	10.00	6866.46
Freq.		3	1	2	1	3	1	3	1	1	1	3	1
% to RVNSc [5]	3900-7980	2.78	26.83	2.50	24.35	5.56	19.24	0.00	27.29	16.67	27.17	25.00	14.06
08 - Worst	4800	36.00	11059.15	42.00	9173.21	37.00	10468.23	8.00	5856.81	12.00	5018.82	9.00	6077.79
Freq.		3	1	1	1	3	1	3	1	3	1	2	1
% to RVNSc [5]	3900-7980	0.00	45.11	10.53	24.29	2.78	22.16	0.00	35.06	0.00	30.50	12.50	18.12
09 - Worst	4800	38.00	11148.77	39.00	10894.81	37.00	10623.16	8.00	7784.55	14.00	5381.77	8.00	5853.09
Freq.		1	1	3	1	3	1	3	1	1	1	3	1
% to RVNSc [5]	3900-7980	5.56	15.77	5.41	49.57	2.78	25.38	0.00	13.00	16.67	35.93	0.00	20.58
10 - Worst	4800	37.00	10596.88	38.00	12003.98	37.00	10224.52	8.00	7438.54	12.00	5002.57	8.00	5572.88
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to RVNSc [5]	3900-7980	2.78	19.22	5.56	59.14	2.78	24.43	0.00	19.18	0.00	31.94	0.00	20.69
DACS 02 - AVGs	4800	37.90	11305.57	40.10	10212.78	37.80	10193.59	8.30	7939.27	12.90	5237.09	10.20	6668.15
% to RVNSc [5]	3900-7980	4.70	23.50	5.53	39.49	4.71	18.14	3.75	21.25	7.50	33.51	17.24	18.37
% to ES4C [2]	4800	4.41	26.67	5.53	34.66	4.71	16.33	3.75	22.11	7.50	33.09	18.60	20.84
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table C.11: Comparison between the average case performances of two different DACS systems that use sequential and parallel ants on the problem set R1 of the problem group PG100.

Search way of ants		Parallel		Sequential	
PNo. (R1 set)	Time(secs)	NV	TD	NV	TD
01 - AVGs	1800	19.00	1668.79	19.00	1674.41
SDs		0.00	21.83	0.00	17.13
% to Sequential DACS 02	1800	0.00	-0.34	0.00	0.00
02 - AVGs	1800	18.00	1502.33	18.00	1503.22
SDs		0.00	6.89	0.00	4.19
% to Sequential DACS 02	1800	0.00	-0.06	0.00	0.00
03 - AVGs	1800	13.67	1260.65	14.00	1245.87
SDs		0.58	33.94	0.00	12.11
% to Sequential DACS 02	1800	-2.38	1.19	0.00	0.00
04 - AVGs	1800	10.33	1020.51	10.33	1035.07
SDs		0.58	11.74	0.58	30.92
% to Sequential DACS 02	1800	0.00	-1.41	0.00	0.00
05 - AVGs	1800	14.67	1427.37	14.67	1432.42
SDs		0.58	16.64	0.58	27.29
% to Sequential DACS 02	1800	0.00	-0.35	0.00	0.00
06 - AVGs	1800	13.00	1318.01	12.33	1286.83
SDs		0.00	31.40	0.58	14.61
% to Sequential DACS 02	1800	5.41	2.42	0.00	0.00
07 - AVGs	1800	11.67	1113.74	11.00	1136.64
SDs		0.58	14.89	0.00	35.85
% to Sequential DACS 02	1800	6.06	-2.01	0.00	0.00
08 - AVGs	1800	10.00	996.51	10.00	989.01
SDs		0.00	24.89	0.00	20.82
% to Sequential DACS 02	1800	0.00	0.76	0.00	0.00
09 - AVGs	1800	12.33	1232.43	12.33	1248.75
SDs		0.58	32.70	0.58	21.76
% to Sequential DACS 02	1800	0.00	-1.31	0.00	0.00
10 - AVGs	1800	11.67	1194.95	12.00	1138.78
SDs		0.58	42.03	0.00	14.74
% to Sequential DACS 02	1800	-2.78	4.93	0.00	0.00
11 - AVGs	1800	12.00	1128.38	11.67	1151.24
SDs		0.00	36.58	0.58	71.14
% to Sequential DACS 02	1800	2.86	-1.99	0.00	0.00
12 - AVGs	1800	10.33	1040.86	10.33	1019.61
SDs		0.58	38.36	0.58	25.51
% to Sequential DACS 02	1800	0.00	2.08	0.00	0.00
AVGs	1800	13.06	1242.05	12.97	1238.49
SDs		0.13	10.30	0.10	12.42
% to Sequential DACS 02	1800	0.64	0.29	0.00	0.00

Table C.12: Comparison between the systems DACS 2.1 + SI1-Like 01 and DACS 2.1 + NN on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
	Time(secs)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
V2.1+SILK - AVGs	100	13.47	1295.24	10.00	882.87	13.33	1488.72	3.18	1045.10	3.13	614.19	3.75	1247.96
SDs		0.25	23.28	0.00	19.00	0.26	28.18	0.00	9.39	0.00	12.36	0.00	11.75
% to V2.1+NN	100	0.41	-1.78	0.00	-2.45	-0.93	-1.89	0.00	-0.64	1.35	0.87	1.12	-3.62
V2.1+SILK - AVGs	300	13.17	1263.39	10.00	854.42	12.75	1445.16	3.18	1002.90	3.00	596.28	3.58	1202.28
SDs		0.17	5.53	0.00	14.40	0.22	4.99	0.00	1.21	0.00	7.84	0.07	3.58
% to V2.1+NN	300	-1.25	-0.51	0.00	-2.92	-1.92	-0.94	1.94	-0.81	0.00	0.06	-1.15	1.40
V2.1+SILK - AVGs	400	13.31	1247.91	10.00	857.57	13.00	1435.63	3.15	994.39	3.00	598.07	3.58	1191.22
SDs		0.13	0.43	0.00	17.29	0.13	6.11	0.05	9.07	0.00	7.53	0.07	3.05
% to V2.1+NN	400	1.27	-2.81	0.00	-1.07	0.00	-2.29	-0.95	-0.51	0.00	1.18	0.00	1.08
V2.1+SILK - AVGs	600	13.17	1252.14	10.00	856.12	12.83	1423.90	3.12	979.46	3.00	591.31	3.50	1202.40
SDs		0.22	15.32	0.00	4.29	0.14	6.37	0.05	5.19	0.00	0.75	0.00	36.62
% to V2.1+NN	600	-0.21	-1.74	0.00	0.12	-0.65	-1.57	-0.96	-0.94	0.00	0.15	-3.45	2.65
V2.1+SILK - AVGs	1200	13.06	1240.21	10.00	849.48	12.75	1418.69	3.12	978.85	3.00	591.29	3.50	1166.56
SDs		0.24	2.32	0.00	9.67	0.13	4.94	0.10	6.35	0.00	0.77	0.13	10.69
% to V2.1+NN	1200	-0.63	-0.67	0.00	-1.71	-0.33	-1.30	1.98	0.31	0.00	0.03	1.20	-0.57
V2.1+SILK - AVGs	1800	13.03	1233.76	10.00	851.47	12.75	1407.95	3.18	965.32	3.00	592.38	3.50	1152.25
SDs		0.05	3.85	0.00	6.76	0.00	4.79	0.00	5.15	0.00	4.01	0.13	17.44
% to V2.1+NN	1800	0.00	-1.87	0.00	-0.03	0.99	-1.22	2.94	-0.58	0.00	-0.38	-2.33	0.24
V2.1+NN - AVGs	100	13.42	1318.68	10.00	905.05	13.46	1517.45	3.18	1051.87	3.08	608.91	3.71	1294.87
SDs		0.00	18.37	0.00	12.11	0.07	34.57	0.00	7.38	0.07	7.60	0.14	60.86
V2.1+NN - AVGs	300	13.33	1269.82	10.00	880.15	13.00	1458.89	3.12	1011.13	3.00	595.94	3.63	1185.68
SDs		0.00	5.22	0.00	23.42	0.22	20.60	0.05	9.30	0.00	4.53	0.13	12.44
V2.1+NN - AVGs	400	13.14	1283.97	10.00	866.85	13.00	1469.34	3.18	999.47	3.00	591.11	3.58	1178.47
SDs		0.13	14.52	0.00	4.54	0.33	13.83	0.00	7.18	0.00	1.38	0.07	6.07
V2.1+NN - AVGs	600	13.19	1274.35	10.00	855.12	12.92	1446.54	3.15	988.78	3.00	590.40	3.63	1171.39
SDs		0.05	27.75	0.00	9.26	0.07	26.48	0.05	15.66	0.00	0.21	0.00	8.81
V2.1+NN - AVGs	1200	13.14	1248.54	10.00	864.26	12.79	1437.36	3.06	975.86	3.00	591.14	3.46	1173.25
SDs		0.13	6.23	0.00	10.92	0.19	36.11	0.10	5.23	0.00	1.50	0.07	10.97
V2.1+NN - AVGs	1800	13.03	1257.25	10.00	851.70	12.63	1425.31	3.09	970.99	3.00	594.66	3.58	1149.45
SDs		0.10	28.33	0.00	14.05	0.25	17.64	0.09	10.26	0.00	7.30	0.07	8.56

Table C.13: Comparison between the average case performances of the algorithms DACS 2.1 + SI1-Like 01 and DACS 2.1 + NN, after three runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.67	1662.82	10.00	828.94	15.00	1694.65	4.00	1305.75	3.00	591.56	4.00	1497.85
SDs		0.58	14.24	0.00	0.00	0.00	2.41	0.00	28.81	0.00	0.00	0.00	24.32
% to V2.1+NN	1800	1.72	-1.73	0.00	0.00	-2.17	-1.53	0.00	-1.00	0.00	0.00	0.00	-1.08
02 - AVGs	1800	18.00	1493.45	10.00	886.25	14.00	1518.98	4.00	1121.34	3.00	591.56	4.00	1214.30
SDs		0.00	11.54	0.00	58.37	0.00	6.24	0.00	23.44	0.00	0.00	0.00	28.11
% to V2.1+NN	1800	1.89	-0.26	0.00	3.78	0.00	0.17	9.09	-4.51	0.00	0.00	0.00	2.11
03 - AVGs	1800	14.00	1250.36	10.00	936.29	12.00	1336.39	3.00	977.09	3.00	591.17	3.00	1123.39
SDs		0.00	12.50	0.00	89.07	0.00	20.19	0.00	13.89	0.00	0.00	0.00	22.34
% to V2.1+NN	1800	0.00	-1.64	0.00	3.78	2.86	-1.15	0.00	0.36	0.00	0.00	0.00	0.21
04 - AVGs	1800	10.67	1012.05	10.00	867.08	11.00	1210.82	3.00	785.48	3.00	610.74	3.00	860.08
SDs		0.58	22.81	0.00	12.95	0.00	18.44	0.00	19.48	0.00	32.05	0.00	22.11
% to V2.1+NN	1800	0.00	-1.47	0.00	2.49	0.00	0.88	0.00	-0.08	0.00	-2.90	0.00	0.57
05 - AVGs	1800	14.33	1402.63	10.00	828.94	15.00	1619.44	3.00	1072.20	3.00	588.88	4.00	1376.44
SDs		0.58	9.18	0.00	0.00	1.00	20.82	0.00	32.28	0.00	0.00	0.00	16.58
% to V2.1+NN	1800	-2.27	-1.92	0.00	0.00	2.27	1.57	0.00	-3.18	0.00	0.00	0.00	-2.25
06 - AVGs	1800	12.33	1306.16	10.00	828.94	12.67	1425.65	3.00	953.94	3.00	588.49	3.33	1205.44
SDs		0.58	14.86	0.00	0.00	0.58	6.62	0.00	11.05	0.00	0.00	0.58	77.40
% to V2.1+NN	1800	-5.13	0.61	0.00	0.00	5.56	-2.52	0.00	1.22	0.00	0.00	-16.67	4.84
07 - AVGs	1800	11.00	1165.51	10.00	828.94	11.33	1274.29	3.00	850.71	3.00	588.29	3.67	1058.22
SDs		0.00	38.65	0.00	0.00	0.58	44.17	0.00	17.69	0.00	0.00	0.58	48.61
% to V2.1+NN	1800	-2.94	-1.99	0.00	-2.72	0.00	-6.42	0.00	-0.92	0.00	0.00	0.00	-1.55
08 - AVGs	1800	10.00	993.61	10.00	828.94	11.00	1183.36	3.00	740.73	3.00	588.32	3.00	882.27
SDs		0.00	31.30	0.00	0.00	0.00	45.15	0.00	9.22	0.00	0.00	0.00	43.93
% to V2.1+NN	1800	0.00	-1.36	0.00	-7.21	0.00	-0.91	28.57	-2.93	0.00	0.00	0.00	-0.10
09 - AVGs	1800	12.33	1249.64	10.00	828.94	-	-	3.00	1005.33	-	-	-	-
SDs		0.58	22.40	0.00	0.00	-	-	0.00	34.57	-	-	-	-
% to V2.1+NN	1800	0.00	3.05	0.00	-0.22	-	-	0.00	2.90	-	-	-	-
10 - AVGs	1800	11.67	1131.34	-	-	-	-	3.00	978.39	-	-	-	-
SDs		0.58	30.90	-	-	-	-	0.00	18.35	-	-	-	-
% to V2.1+NN	1800	0.00	-13.84	-	-	-	-	0.00	0.72	-	-	-	-
11 - AVGs	1800	11.67	1123.22	-	-	-	-	3.00	827.51	-	-	-	-
SDs		0.58	36.40	-	-	-	-	0.00	20.59	-	-	-	-
% to V2.1+NN	1800	0.00	-0.93	-	-	-	-	0.00	2.43	-	-	-	-
12 - AVGs	1800	10.67	1014.31	-	-	-	-	-	-	-	-	-	-
SDs		0.58	38.47	-	-	-	-	-	-	-	-	-	-
% to V2.1+NN	1800	6.67	-0.08	-	-	-	-	-	-	-	-	-	-
V2.1+SILK -AVGs	1800	13.03	1233.76	10.00	851.47	12.75	1407.95	3.18	965.32	3.00	592.38	3.50	1152.25
SDs		0.05	3.85	0.00	6.76	0.00	4.79	0.00	5.15	0.00	4.01	0.13	17.44
% to V2.1+NN	1800	0.00	-1.87	0.00	-0.03	0.99	-1.22	2.94	-0.58	0.00	-0.38	-2.33	0.24

Table C.14: The average case performance of the algorithm DACS 2.1 + NN, after three runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.33	1692.07	10.00	828.94	15.33	1721.04	4.00	1318.94	3.00	591.56	4.00	1514.18
SDs		0.58	21.00	0.00	0.00	0.58	21.53	0.00	28.74	0.00	0.00	0.00	30.05
02 - AVGs	1800	17.67	1497.40	10.00	853.93	14.00	1516.35	3.67	1174.35	3.00	591.56	4.00	1189.24
SDs		0.58	10.43	0.00	43.29	0.00	13.08	0.58	74.19	0.00	0.00	0.00	17.03
03 - AVGs	1800	14.00	1271.19	10.00	902.22	11.67	1351.94	3.00	973.63	3.00	591.17	3.00	1121.04
SDs		0.00	24.01	0.00	20.95	0.58	32.00	0.00	19.01	0.00	0.00	0.00	6.13
04 - AVGs	1800	10.67	1027.13	10.00	846.02	11.00	1200.28	3.00	786.14	3.00	628.97	3.00	855.18
SDs		0.58	16.40	0.00	7.40	0.00	20.53	0.00	4.34	0.00	58.44	0.00	20.99
05 - AVGs	1800	14.67	1430.05	10.00	828.94	14.67	1594.41	3.00	1107.36	3.00	588.88	4.00	1408.10
SDs		0.58	23.44	0.00	0.00	0.58	29.26	0.00	19.91	0.00	0.00	0.00	5.82
06 - AVGs	1800	13.00	1298.22	10.00	828.94	12.00	1462.55	3.00	942.44	3.00	588.49	4.00	1149.78
SDs		0.00	18.29	0.00	0.00	0.00	63.84	0.00	20.39	0.00	0.00	0.00	16.42
07 - AVGs	1800	11.33	1189.12	10.00	852.11	11.33	1361.65	3.00	858.65	3.00	588.29	3.67	1074.92
SDs		0.58	78.46	0.00	20.07	0.58	61.91	0.00	16.58	0.00	0.00	0.58	49.07
08 - AVGs	1800	10.00	1007.27	10.00	893.40	11.00	1194.25	2.33	763.07	3.00	588.32	3.00	883.20
SDs		0.00	15.04	0.00	82.15	0.00	28.48	0.58	13.29	0.00	0.00	0.00	9.41
09 - AVGs	1800	12.33	1212.70	10.00	830.75	-	-	3.00	977.03	-	-	-	-
SDs		0.58	31.73	0.00	3.14	-	-	0.00	31.18	-	-	-	-
10 - AVGs	1800	11.67	1313.00	-	-	-	-	3.00	971.37	-	-	-	-
SDs		0.58	261.28	-	-	-	-	0.00	9.33	-	-	-	-
11 - AVGs	1800	11.67	1133.78	-	-	-	-	3.00	807.86	-	-	-	-
SDs		0.58	39.77	-	-	-	-	0.00	21.87	-	-	-	-
12 - AVGs	1800	10.00	1015.08	-	-	-	-	-	-	-	-	-	-
SDs		0.00	27.14	-	-	-	-	-	-	-	-	-	-
V2.1+NN - AVGs	1800	13.03	1257.25	10.00	851.70	12.63	1425.31	3.09	970.99	3.00	594.66	3.58	1149.45
SDs		0.10	28.33	0.00	14.05	0.25	17.64	0.09	10.26	0.00	7.30	0.07	8.56

Table C.15: Comparison between the average case performances of two different DACS systems, after three runs of 1800 seconds, on the problem set R1 of the problem group PG100.

Candidate lists		Distance Oriented		Time Oriented	
PNo. (R1 set)	Time(secs)	NV	TD	NV	TD
01 - AVGs	1800	19.00	1680.86	19.33	1671.68
SDs		0.00	24.40	0.58	8.86
% to Time Oriented DACS 2.1	1800	-1.72	0.55	0.00	0.00
02 - AVGs	1800	18.00	1474.49	17.67	1486.83
SDs		0.00	1.08	0.58	7.96
% to Time Oriented DACS 2.1	1800	1.89	-0.83	0.00	0.00
03 - AVGs	1800	14.00	1223.37	13.67	1260.59
SDs		0.00	7.21	0.58	69.01
% to Time Oriented DACS 2.1	1800	2.44	-4.47	0.00	0.00
04 - AVGs	1800	10.00	1002.05	10.00	1013.17
SDs		0.00	15.03	0.00	16.96
% to Time Oriented DACS 2.1	1800	0.00	-1.10	0.00	0.00
05 - AVGs	1800	14.33	1407.01	14.00	1437.32
SDs		0.58	6.28	0.00	28.17
% to Time Oriented DACS 2.1	1800	2.38	-2.11	0.00	0.00
06 - AVGs	1800	12.33	1316.37	12.67	1274.19
SDs		0.58	34.03	0.58	6.98
% to Time Oriented DACS 2.1	1800	-2.63	3.31	0.00	0.00
07 - AVGs	1800	11.00	1089.83	11.00	1089.97
SDs		0.00	15.00	0.00	9.28
% to Time Oriented DACS 2.1	1800	0.00	-0.01	0.00	0.00
08 - AVGs	1800	10.00	965.64	10.00	970.07
SDs		0.00	10.84	0.00	10.08
% to Time Oriented DACS 2.1	1800	0.00	-0.46	0.00	0.00
09 - AVGs	1800	12.67	1208.42	12.00	1218.77
SDs		0.58	46.53	0.00	22.85
% to Time Oriented DACS 2.1	1800	5.56	-0.85	0.00	0.00
10 - AVGs	1800	11.33	1133.74	11.67	1109.27
SDs		0.58	51.26	0.58	13.81
% to Time Oriented DACS 2.1	1800	-2.86	2.21	0.00	0.00
11 - AVGs	1800	11.00	1103.17	11.00	1127.13
SDs		0.00	14.78	0.00	35.08
% to Time Oriented DACS 2.1	1800	0.00	-2.13	0.00	0.00
12 - AVGs	1800	10.00	1002.34	10.00	984.30
SDs		0.00	30.49	0.00	11.39
% to Time Oriented DACS 2.1	1800	0.00	1.83	0.00	0.00
DACS 02 - AVGs	1800	12.81	1217.27	12.75	1221.94
SDs		0.10	12.83	0.08	5.85
% to Time Oriented DACS 2.1	1800	0.44	-0.38	0.00	0.00

Table C.16: Comparison between the average case performances of three different DACS systems, after three runs of 1800 seconds, on the problem set R1 of the problem group PG100.

Pheromone updating of the other colony		Yes		No		No	
Local search moves near depot		No		Yes		No	
PNo. (R1 set)	Time(secs.)	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.67	1679.50	19.67	1658.97	19.00	1674.41
SDs		0.58	9.75	0.58	5.62	0.00	17.13
% to Sequential DACS 02	1800	3.51	0.30	3.51	-0.92	0.00	0.00
02 - AVGs	1800	17.33	1508.51	18.33	1481.82	18.00	1503.22
SDs		0.58	4.93	0.58	7.00	0.00	4.19
% to Sequential DACS 02	1800	-3.70	0.35	1.85	-1.42	0.00	0.00
03 - AVGs	1800	14.00	1261.64	14.00	1229.43	14.00	1245.87
SDs		0.00	7.20	0.00	9.36	0.00	12.11
% to Sequential DACS 02	1800	0.00	1.27	0.00	-1.32	0.00	0.00
04 - AVGs	1800	10.67	1006.10	10.67	1029.70	10.33	1035.07
SDs		0.58	15.57	0.58	41.69	0.58	30.92
% to Sequential DACS 02	1800	3.23	-2.80	3.23	-0.52	0.00	0.00
05 - AVGs	1800	14.67	1438.51	14.33	1421.98	14.67	1432.42
SDs		0.58	28.56	0.58	3.35	0.58	27.29
% to Sequential DACS 02	1800	0.00	0.42	-2.27	-0.73	0.00	0.00
06 - AVGs	1800	12.67	1302.95	13.00	1280.51	12.33	1286.83
SDs		0.58	4.05	0.00	10.39	0.58	14.61
% to Sequential DACS 02	1800	2.70	1.25	5.41	-0.49	0.00	0.00
07 - AVGs	1800	11.33	1114.87	11.00	1095.41	11.00	1136.64
SDs		0.58	15.42	0.00	8.80	0.00	35.85
% to Sequential DACS 02	1800	3.03	-1.92	0.00	-3.63	0.00	0.00
08 - AVGs	1800	10.00	999.71	10.00	986.65	10.00	989.01
SDs		0.00	10.69	0.00	13.78	0.00	20.82
% to Sequential DACS 02	1800	0.00	1.08	0.00	-0.24	0.00	0.00
09 - AVGs	1800	12.00	1211.48	12.00	1198.11	12.33	1248.75
SDs		0.00	21.29	0.00	25.52	0.58	21.76
% to Sequential DACS 02	1800	-2.70	-2.98	-2.70	-4.06	0.00	0.00
10 - AVGs	1800	11.67	1154.67	12.00	1102.29	12.00	1138.78
SDs		0.58	17.09	0.00	7.14	0.00	14.74
% to Sequential DACS 02	1800	-2.78	1.40	0.00	-3.20	0.00	0.00
11 - AVGs	1800	11.00	1127.70	11.00	1089.29	11.67	1151.24
SDs		0.00	8.38	0.00	14.90	0.58	71.14
% to Sequential DACS 02	1800	-5.71	-2.05	-5.71	-5.38	0.00	0.00
12 - AVGs	1800	10.00	990.30	10.33	976.36	10.33	1019.61
SDs		0.00	16.00	0.58	12.78	0.58	25.51
% to Sequential DACS 02	1800	-3.23	-2.87	0.00	-4.24	0.00	0.00
DACS 01 - AVGs	1800	12.92	1232.99	13.03	1212.54	12.97	1238.49
SDs		0.08	3.48	0.05	6.35	0.10	12.42
% to Sequential DACS 02	1800	-0.43	-0.44	0.43	-2.09	0.00	0.00

Appendix D

Tables of results related to the system DACS 03

This section mentions in Tables D.1 to D.6 information about the experiments done to the system DACS 03, which uses the push-forward and push-backward strategy PFPBS as in Section 4.7.1. All tables, collectively, represent the best and worst case performances of DACS 03, on the problem groups PG100, PG200 and PG400 in Section 2.2, after thirty or three runs of 1800, 2400 or 4800 seconds.

Table D.1: Comparison between the best case performances of DACS 03 and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	1800	19.00	1656.88	10.00	828.94	14.00	1747.00	4.00	1257.39	3.00	591.56	4.00	1413.52
Freq.		30	1	30	30	1	1	30	1	30	30	30	1
% to DACS 02	1800	0.00	-1.99	0.00	0.00	-6.67	2.80	0.00	-1.13	0.00	0.00	0.00	-4.95
02 - Best	1800	17.00	1489.81	10.00	828.94	13.00	1485.70	3.00	1191.70	3.00	591.56	4.00	1161.29
Freq.		24	1	30	24	30	1	19	1	30	30	30	3
% to DACS 02	1800	0.00	-0.61	0.00	0.00	0.00	-2.30	-25.00	7.81	0.00	0.00	0.00	-2.82
03 - Best	1800	13.00	1297.67	10.00	828.06	11.00	1268.40	3.00	952.04	3.00	591.17	3.00	1076.25
Freq.		15	1	30	23	19	1	30	1	30	19	30	1
% to DACS 02	1800	-7.14	5.60	0.00	-1.34	0.00	-7.19	0.00	-4.76	0.00	0.00	0.00	-3.58
04 - Best	1800	10.00	991.05	10.00	824.78	10.00	1140.87	3.00	753.85	3.00	590.60	3.00	801.91
Freq.		25	1	30	28	14	1	30	1	30	2	30	1
% to DACS 02	1800	-9.09	-2.00	0.00	0.00	-9.09	-2.96	0.00	-7.30	0.00	-0.56	0.00	-7.08
05 - Best	1800	14.00	1407.85	10.00	828.94	14.00	1552.98	3.00	1029.16	3.00	588.88	4.00	1324.89
Freq.		29	1	30	29	18	1	30	1	30	30	30	1
% to DACS 02	1800	0.00	-3.61	0.00	0.00	0.00	-4.71	0.00	-1.35	0.00	0.00	0.00	-1.41
06 - Best	1800	12.00	1261.78	10.00	828.94	12.00	1399.49	3.00	913.91	3.00	588.49	3.00	1182.42
Freq.		24	1	30	30	29	1	30	1	30	30	27	1
% to DACS 02	1800	-7.69	-2.51	0.00	0.00	0.00	-8.86	0.00	-5.84	0.00	0.00	0.00	-3.70
07 - Best	1800	10.00	1131.90	10.00	828.94	11.00	1236.47	3.00	815.94	3.00	588.29	3.00	1088.20
Freq.		1	1	30	28	24	2	30	1	30	29	29	1
% to DACS 02	1800	-9.09	2.68	0.00	0.00	-8.33	-0.76	0.00	-9.74	0.00	0.00	-25.00	4.48
08 - Best	1800	10.00	942.78	10.00	828.94	10.00	1142.55	2.00	729.17	3.00	588.32	3.00	846.83
Freq.		30	1	30	26	5	1	28	1	30	30	30	1
% to DACS 02	1800	0.00	-1.92	0.00	0.00	-9.09	-0.64	-33.33	-1.31	0.00	0.00	0.00	-0.37
09 - Best	1800	12.00	1159.29	10.00	828.94	-	-	3.00	926.48	-	-	-	-
Freq.		30	1	30	30	-	-	30	1	-	-	-	-
% to DACS 02	1800	0.00	-1.41	0.00	0.00	-	-	0.00	-3.07	-	-	-	-
10 - Best	1800	10.00	1179.74	-	-	-	-	3.00	949.37	-	-	-	-
Freq.		1	1	-	-	-	-	30	1	-	-	-	-
% to DACS 02	1800	-9.09	5.25	-	-	-	-	0.00	-4.70	-	-	-	-
11 - Best	1800	11.00	1069.58	-	-	-	-	3.00	778.91	-	-	-	-
Freq.		29	1	-	-	-	-	30	1	-	-	-	-
% to DACS 02	1800	0.00	-5.40	-	-	-	-	0.00	-7.59	-	-	-	-
12 - Best	1800	10.00	959.60	-	-	-	-	-	-	-	-	-	-
Freq.		30	1	-	-	-	-	-	-	-	-	-	-
% to DACS 02	1800	0.00	-1.64	-	-	-	-	-	-	-	-	-	-
DACS 03 - AVGs	1800	12.33	1212.33	10.00	828.38	11.88	1371.68	3.00	936.17	3.00	589.86	3.38	1111.91
% to DACS 02	1800	-3.27	-0.70	0.00	-0.15	-4.04	-3.09	-5.71	-3.23	0.00	-0.07	-3.57	-2.51
% to SA+LNS [29]	1800	1.34	0.71	0.00	0.00	2.11	-0.53	9.89	-4.50	0.00	0.00	3.85	-4.06
DACS 02 - AVGs	1800	12.75	1220.90	10.00	829.63	12.38	1415.45	3.18	967.41	3.00	590.28	3.50	1140.58
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table D.2: Comparison between the worst case performances of DACS 03 and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	1800	19.00	1728.67	10.00	828.94	15.00	1778.75	4.00	1335.31	3.00	591.56	4.00	1568.59
Freq.		30	1	30	30	29	1	30	1	30	30	30	1
% to DACS 02	1800	-5.00	4.02	0.00	0.00	-6.25	6.08	0.00	1.39	0.00	0.00	0.00	3.45
02 - Worst	1800	18.00	1479.22	10.00	886.84	13.00	1569.26	4.00	1123.55	3.00	591.56	4.00	1272.76
Freq.		6	1	30	1	30	1	11	1	30	30	30	1
% to DACS 02	1800	0.00	-0.35	0.00	6.98	-7.14	2.80	0.00	-1.92	0.00	0.00	0.00	2.20
03 - Worst	1800	14.00	1237.92	10.00	859.37	12.00	1345.32	3.00	1003.94	3.00	600.54	3.00	1223.02
Freq.		15	1	30	1	11	1	30	1	30	4	30	1
% to DACS 02	1800	0.00	-0.28	0.00	-4.20	0.00	0.53	0.00	-1.50	0.00	1.58	0.00	6.94
04 - Worst	1800	11.00	1015.02	10.00	828.07	11.00	1178.03	3.00	792.81	3.00	680.09	3.00	858.91
Freq.		5	1	30	1	16	1	30	1	30	1	30	1
% to DACS 02	1800	0.00	-1.06	0.00	-4.30	0.00	-1.21	0.00	-5.17	0.00	10.86	0.00	-1.30
05 - Worst	1800	15.00	1452.85	10.00	862.37	15.00	1600.85	3.00	1131.53	3.00	588.88	4.00	1454.38
Freq.		1	1	30	1	12	1	30	1	30	30	30	1
% to DACS 02	1800	0.00	1.72	0.00	4.03	0.00	0.30	0.00	4.52	0.00	0.00	0.00	4.89
06 - Worst	1800	13.00	1297.96	10.00	828.94	13.00	1400.52	3.00	975.66	3.00	588.49	4.00	1119.30
Freq.		6	1	30	30	1	1	30	1	30	30	3	1
% to DACS 02	1800	0.00	-2.80	0.00	0.00	0.00	-1.89	0.00	-3.61	0.00	0.00	0.00	-1.60
07 - Worst	1800	11.00	1144.73	10.00	863.70	12.00	1274.69	3.00	865.07	3.00	591.73	4.00	1005.10
Freq.		29	1	30	2	6	1	30	1	30	1	1	1
% to DACS 02	1800	-8.33	0.88	0.00	4.19	0.00	-1.85	0.00	-7.22	0.00	0.00	0.00	-8.40
08 - Worst	1800	10.00	987.60	10.00	915.91	11.00	1172.76	3.00	733.77	3.00	588.32	3.00	947.50
Freq.		30	1	30	1	25	1	2	1	30	30	30	1
% to DACS 02	1800	0.00	-0.21	0.00	10.49	0.00	-0.58	0.00	-6.73	0.00	0.00	0.00	0.85
09 - Worst	1800	12.00	1241.94	10.00	828.94	-	-	3.00	989.84	-	-	-	-
Freq.		30	1	30	30	-	-	30	1	-	-	-	-
% to DACS 02	1800	0.00	-1.10	0.00	0.00	-	-	0.00	-1.67	-	-	-	-
10 - Worst	1800	12.00	1113.81	-	-	-	-	3.00	1031.71	-	-	-	-
Freq.		4	1	-	-	-	-	30	1	-	-	-	-
% to DACS 02	1800	0.00	0.54	-	-	-	-	0.00	1.49	-	-	-	-
11 - Worst	1800	12.00	1105.63	-	-	-	-	3.00	830.34	-	-	-	-
Freq.		1	1	-	-	-	-	30	1	-	-	-	-
% to DACS 02	1800	0.00	-0.92	-	-	-	-	0.00	-4.04	-	-	-	-
12 - Worst	1800	10.00	1033.13	-	-	-	-	-	-	-	-	-	-
Freq.		30	1	-	-	-	-	-	-	-	-	-	-
% to DACS 02	1800	-9.09	3.70	-	-	-	-	-	-	-	-	-	-
DACS 03 - AVGs	1800	13.08	1236.54	10.00	855.90	12.75	1415.02	3.18	983.05	3.00	602.65	3.63	1181.20
% to DACS 02	1800	-1.88	0.41	0.00	1.83	-1.92	0.75	0.00	-1.88	0.00	1.60	0.00	1.21
% to SA+LNS [29]	1800	4.00	2.85	0.00	3.32	7.32	2.97	2.97	-0.11	0.00	-8.51	7.25	-0.45
DACS 02 - AVGs	1800	13.33	1231.44	10.00	840.54	13.00	1404.47	3.18	1001.84	3.00	593.15	3.63	1167.07
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table D.3: Comparison between the best case performances of DACS 03 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	2400	20.00	5328.26	20.00	2704.57	19.00	3676.06	5.00	4218.91	6.00	1931.44	6.00	3227.82
Freq.		3	1	3	3	3	1	3	1	3	2	3	1
% to DACS 02	2400	0.00	-6.18	0.00	0.00	0.00	-6.23	0.00	-0.72	0.00	-1.25	-14.29	-4.51
02 - Best	2400	18.00	4593.97	18.00	3037.62	18.00	3696.25	4.00	3727.02	6.00	1863.16	5.00	2948.93
Freq.		2	1	1	1	3	1	1	1	3	1	3	1
% to DACS 02	2400	-5.26	5.34	-5.26	-14.15	-5.26	3.11	-20.00	1.64	0.00	-2.33	0.00	-7.63
03 - Best	2400	18.00	3786.85	18.00	2929.80	18.00	3603.68	4.00	3025.85	6.00	1838.66	4.00	2715.43
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	-5.26	-0.78	-5.26	-11.82	0.00	-15.89	0.00	-5.47	0.00	-13.71	-20.00	1.95
04 - Best	2400	18.00	3286.51	18.00	2809.67	18.00	3251.33	4.00	2117.85	6.00	1863.32	4.00	2204.79
Freq.		3	1	3	1	3	1	3	1	2	1	3	1
% to DACS 02	2400	0.00	-13.95	-5.26	-3.91	0.00	-10.75	0.00	-7.60	-14.29	-2.97	0.00	-11.64
05 - Best	2400	18.00	4769.84	20.00	2702.05	19.00	3491.33	4.00	3690.71	6.00	1883.55	5.00	2859.71
Freq.		2	1	3	2	3	1	3	1	3	2	3	1
% to DACS 02	2400	-5.26	6.24	0.00	-1.32	0.00	-5.75	0.00	0.00	0.00	-0.63	0.00	-8.25
06 - Best	2400	18.00	4022.23	20.00	2701.04	18.00	4244.07	4.00	3080.34	6.00	1871.24	5.00	2835.96
Freq.		3	1	3	1	2	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-9.45	0.00	0.00	-5.26	12.82	0.00	-5.47	0.00	-7.28	0.00	-5.15
07 - Best	2400	18.00	3495.89	20.00	2701.04	18.00	3969.75	4.00	2658.60	6.00	1901.87	4.00	3032.46
Freq.		3	1	3	2	3	1	3	1	3	1	1	1
% to DACS 02	2400	0.00	-16.57	0.00	-1.72	-5.26	8.82	0.00	-2.73	0.00	-1.53	0.00	-1.49
08 - Best	2400	18.00	3215.95	20.00	2695.81	18.00	3709.87	4.00	2044.06	6.00	1855.26	4.00	2518.26
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-9.82	0.00	-3.43	-5.26	6.85	0.00	-8.76	0.00	-2.54	0.00	-4.54
09 - Best	2400	18.00	4460.35	19.00	2708.67	18.00	3558.28	4.00	3203.21	6.00	1860.31	4.00	2373.23
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	-5.26	2.58	0.00	-5.48	0.00	-19.37	0.00	-4.78	0.00	-3.91	0.00	-3.55
10 - Best	2400	18.00	3637.88	18.00	2885.76	18.00	3466.72	4.00	2768.78	6.00	1838.46	4.00	2263.50
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-11.41	0.00	-12.93	0.00	-14.41	0.00	-7.43	0.00	-9.40	0.00	-2.33
DACS 03 - AVGs	2400	18.20	4059.77	19.10	2787.60	18.20	3666.73	4.10	3053.53	6.00	1870.73	4.50	2698.01
% to DACS 02	2400	-2.15	-5.19	-1.55	-5.98	-2.15	-4.72	-2.38	-3.64	-1.64	-4.69	-4.26	-4.77
% to RVNSc [5]	720-1680	0.55	6.24	1.06	0.32	1.11	4.52	2.50	0.27	0.00	1.54	2.27	2.65
% to ES4C [2]	2400	0.00	9.58	1.06	0.20	1.11	3.14	2.50	-0.05	0.00	1.34	4.65	0.86
DACS 02 - AVGs	2400	18.60	4281.91	19.40	2964.86	18.60	3848.24	4.20	3168.75	6.10	1962.88	4.70	2833.28
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table D.4: Comparison between the worst case performances of DACS 03 and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	2400	20.00	5516.17	20.00	2704.57	19.00	3866.24	5.00	4441.10	6.00	2014.82	6.00	3437.23
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS 02	2400	-4.76	-1.01	0.00	-1.03	0.00	-5.39	0.00	-0.06	0.00	1.31	-14.29	-1.69
02 - Worst	2400	19.00	4237.05	19.00	2945.46	18.00	3796.73	5.00	3547.76	6.00	1905.16	5.00	3034.49
Freq.		1	1	2	1	3	1	2	1	3	1	3	1
% to DACS 02	2400	0.00	-9.55	-5.00	-23.27	-5.26	-2.84	0.00	-9.72	0.00	-0.69	-16.67	-1.31
03 - Worst	2400	18.00	3806.12	18.00	3176.69	18.00	3709.80	4.00	3073.54	6.00	1870.22	4.00	2819.15
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	-5.26	-5.18	-5.26	-20.81	0.00	-17.52	0.00	-10.13	-14.29	-3.61	-20.00	-1.92
04 - Worst	2400	18.00	3482.33	18.00	3017.15	18.00	3449.93	4.00	2181.50	7.00	1898.38	4.00	2392.36
Freq.		3	1	3	1	3	1	3	1	1	1	3	1
% to DACS 02	2400	0.00	-9.82	-5.26	-7.48	0.00	-13.79	0.00	-9.05	0.00	-3.23	0.00	-5.73
05 - Worst	2400	19.00	4357.19	20.00	2713.94	19.00	3634.79	4.00	3910.87	6.00	1944.57	5.00	3036.49
Freq.		1	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-14.37	0.00	-1.30	0.00	-6.71	0.00	4.82	0.00	-4.10	-16.67	-0.25
06 - Worst	2400	18.00	4191.57	20.00	2810.44	19.00	3607.55	4.00	3109.92	6.00	1930.36	5.00	2898.35
Freq.		3	1	3	1	1	1	3	1	3	1	3	1
% to DACS 02	2400	-5.26	3.98	-4.76	-7.59	0.00	-9.02	0.00	-4.95	0.00	-12.95	0.00	-6.31
07 - Worst	2400	18.00	3556.67	20.00	2712.93	18.00	4155.83	4.00	2847.42	6.00	2014.68	5.00	2750.52
Freq.		3	1	3	1	3	1	3	1	3	1	2	1
% to DACS 02	2400	0.00	-22.49	0.00	-9.42	-5.26	10.00	0.00	1.75	-14.29	7.53	0.00	-7.43
08 - Worst	2400	18.00	3340.62	20.00	2698.59	18.00	3949.60	4.00	2156.84	6.00	1893.06	4.00	2658.17
Freq.		3	1	3	2	3	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-7.55	0.00	-7.74	-5.26	12.28	0.00	-10.37	0.00	-3.74	0.00	-6.67
09 - Worst	2400	18.00	4641.65	19.00	2781.71	18.00	3737.89	4.00	3242.82	6.00	2013.28	4.00	2506.11
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	-5.26	4.31	0.00	-10.71	-5.26	5.48	0.00	-10.83	-14.29	-4.79	0.00	-6.99
10 - Worst	2400	18.00	3791.68	19.00	2704.62	18.00	3585.85	4.00	2885.89	6.00	1889.71	4.00	2365.88
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS 02	2400	0.00	-12.06	0.00	-10.12	0.00	-21.36	0.00	-7.10	0.00	-14.57	0.00	0.10
DACS 03 - AVGs	2400	18.40	4092.11	19.30	2826.61	18.30	3749.42	4.20	3139.77	6.10	1937.42	4.60	2789.88
% to DACS 02	2400	-2.13	-7.45	-2.03	-10.78	-2.14	-5.69	0.00	-5.26	-4.69	-4.19	-8.00	-3.79
% to RVNSc [5]	720-1680	1.66	7.08	2.12	1.72	1.67	6.88	5.00	3.10	1.67	5.16	4.55	6.15
% to ES4C [2]	2400	1.10	10.45	2.12	1.60	1.67	5.47	5.00	2.77	1.67	4.95	6.98	4.29
DACS 02 - AVGs	2400	18.80	4421.49	19.70	3167.97	18.70	3975.43	4.20	3314.25	6.40	2022.11	5.00	2899.78
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table D.5: Comparison between the best case performances of DACS 03 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	4800	40.00	11397.37	40.00	7159.51	37.00	9384.98	9.00	9647.55	12.00	4185.82	13.00	7440.58
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	0.00	-10.67	0.00	-3.93	-2.63	-8.54	0.00	-6.68	0.00	-2.21	8.33	-9.45
02 - Best	4800	37.00	10174.36	37.00	8586.52	37.00	8717.26	8.00	8548.41	12.00	4071.19	11.00	7012.48
Freq.		3	1	2	1	3	1	3	1	3	1	2	1
% to DACS 02	4800	-2.63	-15.01	-7.50	-42.40	-2.63	-13.42	-11.11	-8.60	0.00	-15.08	0.00	-11.44
03 - Best	4800	37.00	9120.92	37.00	9472.76	36.00	10427.11	8.00	6989.15	12.00	4496.72	9.00	5667.09
Freq.		3	1	2	1	1	1	3	1	1	1	2	1
% to DACS 02	4800	0.00	-12.87	-2.63	-19.23	-2.70	10.12	0.00	-5.28	-7.69	-11.26	0.00	-11.48
04 - Best	4800	36.00	9310.95	37.00	8140.76	36.00	9311.47	8.00	6380.51	12.00	5672.34	8.00	4430.72
Freq.		3	1	3	1	3	1	3	1	1	1	3	1
% to DACS 02	4800	-2.70	4.87	0.00	-13.76	0.00	-17.11	0.00	-9.21	-7.69	18.85	0.00	-10.88
05 - Best	4800	37.00	10567.89	40.00	7193.94	37.00	9205.06	8.00	8028.99	12.00	3993.59	11.00	6980.22
Freq.		3	1	3	1	3	1	3	1	3	1	1	1
% to DACS 02	4800	0.00	-9.63	0.00	-5.79	-2.63	-5.44	0.00	-6.49	0.00	-9.77	-8.33	-5.01
06 - Best	4800	36.00	12299.66	40.00	7241.46	37.00	9157.76	8.00	6989.20	12.00	3973.19	10.00	6890.29
Freq.		1	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	-2.70	11.83	-2.44	-17.29	0.00	-11.50	0.00	-9.93	-7.69	-22.41	0.00	-7.92
07 - Best	4800	36.00	10883.91	40.00	7195.40	37.00	9216.70	8.00	6169.00	12.00	4097.51	9.00	6259.01
Freq.		2	1	3	1	3	1	3	1	1	1	1	1
% to DACS 02	4800	-2.70	7.08	0.00	-10.44	-2.63	-9.37	0.00	-8.83	-7.69	-20.00	-10.00	-4.86
08 - Best	4800	36.00	9032.29	39.00	7973.13	37.00	9156.49	8.00	4946.14	12.00	3958.10	9.00	5605.52
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	0.00	-16.42	-4.88	-5.51	0.00	-6.73	0.00	-8.39	0.00	-16.17	12.50	-6.49
09 - Best	4800	37.00	9973.00	38.00	7245.88	37.00	8969.09	8.00	7044.53	12.00	4328.57	8.00	5642.09
Freq.		3	1	3	1	3	1	3	1	1	1	3	1
% to DACS 02	4800	0.00	-11.53	-2.56	-16.67	0.00	-9.60	0.00	-6.85	-7.69	-15.55	0.00	-2.29
10 - Best	4800	36.00	11045.29	37.00	7565.48	36.00	10391.56	8.00	6836.22	12.00	4111.53	8.00	5432.19
Freq.		3	1	2	1	2	1	3	1	3	1	3	1
% to DACS 02	4800	-2.70	7.07	-2.63	-18.96	-2.70	9.81	0.00	-6.25	0.00	-8.56	0.00	-1.31
DACS 03 - AVGs	4800	36.80	10380.56	38.50	7777.48	36.70	9393.75	8.10	7057.97	12.00	4288.86	9.60	6136.02
% to DACS 02	4800	-1.34	-5.05	-2.28	-17.63	-1.61	-6.52	-1.22	-7.57	-4.00	-10.51	0.00	-7.30
% to RVNSc [5]	3900-7980	1.66	13.39	1.32	6.23	1.66	8.87	1.25	7.79	0.00	9.33	10.34	8.92
% to ES4C [2]	4800	1.38	16.31	1.32	2.55	1.66	7.20	1.25	8.55	0.00	8.99	11.63	11.20
DACS 02 - AVGs	4800	37.30	10932.73	39.40	9442.22	37.30	10048.59	8.20	7636.27	12.50	4792.68	9.60	6619.23
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LCO3 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table D.6: Comparison between the worst case performances of DACS 03 and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	4800	40.00	11634.79	40.00	7181.58	37.00	9859.57	9.00	10181.52	12.00	4335.42	13.00	7683.03
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	-4.76	-14.17	0.00	-5.38	-7.50	-1.28	-10.00	-4.38	0.00	0.11	-7.14	-2.55
02 - Worst	4800	37.00	10407.22	38.00	8397.29	37.00	8805.01	8.00	8714.61	12.00	4262.56	12.00	6760.72
Freq.		3	1	1	1	3	1	3	1	3	1	1	1
% to DACS 02	4800	-2.63	-20.40	-9.52	-35.86	-2.63	-13.53	-11.11	-10.30	-7.69	-26.17	0.00	-19.13
03 - Worst	4800	37.00	9352.20	38.00	8640.49	37.00	8681.91	8.00	7043.33	13.00	4209.86	10.00	5887.93
Freq.		3	1	1	1	2	1	3	1	2	1	1	1
% to DACS 02	4800	-2.63	-11.13	0.00	-34.87	0.00	-14.03	0.00	-10.80	0.00	-32.97	0.00	-7.17
04 - Worst	4800	36.00	9868.74	37.00	8427.21	36.00	9613.42	8.00	5727.28	13.00	4535.13	8.00	4499.36
Freq.		3	1	3	1	3	1	3	1	2	1	3	1
% to DACS 02	4800	-2.70	8.51	0.00	-16.57	-2.70	5.16	0.00	-4.11	0.00	-13.00	0.00	-12.71
05 - Worst	4800	37.00	11123.59	40.00	7270.18	37.00	9440.83	8.00	8694.41	12.00	4288.14	12.00	7251.42
Freq.		3	1	3	1	3	1	3	1	3	1	2	1
% to DACS 02	4800	-2.63	-10.01	-2.44	-12.58	-2.63	-10.35	0.00	-3.67	-7.69	-11.79	0.00	-2.17
06 - Worst	4800	37.00	9846.69	40.00	7284.73	37.00	9346.55	8.00	7361.78	12.00	4439.44	10.00	7033.73
Freq.		2	1	3	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	-2.63	-11.41	-6.98	-17.21	-5.13	-8.21	0.00	-9.44	-7.69	-19.34	-9.09	-1.74
07 - Worst	4800	37.00	9025.91	40.00	7434.10	37.00	9395.43	8.00	6581.60	13.00	4205.71	10.00	6256.57
Freq.		1	1	3	1	3	1	3	1	2	1	2	1
% to DACS 02	4800	0.00	-14.24	-2.44	-16.38	-2.63	-10.48	0.00	-4.98	-7.14	-15.97	0.00	-8.88
08 - Worst	4800	36.00	9265.54	40.00	7294.57	37.00	9360.87	8.00	5252.06	12.00	4138.42	9.00	6057.23
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS 02	4800	0.00	-16.22	-4.76	-20.48	0.00	-10.58	0.00	-10.33	0.00	-17.54	0.00	-0.34
09 - Worst	4800	37.00	9987.85	38.00	7632.27	37.00	9399.46	8.00	7433.43	13.00	4543.38	8.00	5869.42
Freq.		3	1	3	1	3	1	3	1	2	1	3	1
% to DACS 02	4800	-2.63	-10.41	-2.56	-29.95	0.00	-11.52	0.00	-4.51	-7.14	-15.58	0.00	0.28
10 - Worst	4800	36.00	11274.88	38.00	7490.80	37.00	8889.20	8.00	6913.50	12.00	4191.20	8.00	5491.40
Freq.		3	1	1	1	1	1	3	1	3	1	3	1
% to DACS 02	4800	-2.70	6.40	0.00	-37.60	0.00	-13.06	0.00	-7.06	0.00	-16.22	0.00	-1.46
DACS 03 - AVGs	4800	37.00	10178.74	38.90	7705.32	36.90	9279.23	8.10	7390.35	12.40	4314.93	10.00	6279.08
% to DACS 02	4800	-2.37	-9.97	-2.99	-24.55	-2.38	-8.97	-2.41	-6.91	-3.88	-17.61	-1.96	-5.83
% to RVNSc [5]	3900-7980	2.21	11.19	2.37	5.24	2.22	7.54	1.25	12.87	3.33	10.00	14.94	11.46
% to ES4C [2]	4800	1.93	14.05	2.37	1.60	2.22	5.89	1.25	13.66	3.33	9.66	16.28	13.79
DACS 02 - AVGs	4800	37.90	11305.57	40.10	10212.78	37.80	10193.59	8.30	7939.27	12.90	5237.09	10.20	6668.15
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Appendix E

Tables of results related to the system DACS+HLS

This section mentions in Tables E.1 to E.6 information about the experiments done to the system DACS+HLS, which uses the hybrid local search HLS as in Section 4.7.3. All tables, collectively, represent the best and worst case performances of DACS+HLS, on the problem groups PG100, PG200 and PG400 in Section 2.2, after thirty or three runs of 1800, 2400 or 4800 seconds.

Table E.1: Comparison between the best case performances of DACS+HLS and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	1800	19.00	1653.57	10.00	828.94	14.00	1732.39	4.00	1260.46	3.00	591.56	4.00	1421.94
Freq.		30	1	30	30	1	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	-0.20	0.00	0.00	0.00	-0.84	0.00	0.24	0.00	0.00	0.00	0.60
02 - Best	1800	17.00	1491.24	10.00	828.94	13.00	1495.76	3.00	1191.70	3.00	591.56	3.00	1368.23
Freq.		21	2	30	21	30	1	30	1	30	30	10	1
% to DACS 03	1800	0.00	0.10	0.00	0.00	0.00	0.68	0.00	0.00	0.00	0.00	-25.00	17.82
03 - Best	1800	13.00	1296.01	10.00	828.06	11.00	1265.71	3.00	951.66	3.00	591.17	3.00	1083.26
Freq.		19	1	30	16	27	1	30	1	30	23	30	1
% to DACS 03	1800	0.00	-0.13	0.00	0.00	0.00	-0.21	0.00	-0.04	0.00	0.00	0.00	0.65
04 - Best	1800	10.00	986.88	10.00	824.78	10.00	1142.94	2.00	849.34	3.00	590.60	3.00	801.82
Freq.		30	1	30	26	30	1	12	1	30	5	30	1
% to DACS 03	1800	0.00	-0.42	0.00	0.00	0.00	0.18	-33.33	12.67	0.00	0.00	0.00	-0.01
05 - Best	1800	14.00	1396.94	10.00	828.94	14.00	1578.48	3.00	1009.83	3.00	588.88	4.00	1319.50
Freq.		30	1	30	29	29	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	-0.77	0.00	0.00	0.00	1.64	0.00	-1.88	0.00	0.00	0.00	-0.41
06 - Best	1800	12.00	1265.93	10.00	828.94	12.00	1404.07	3.00	913.68	3.00	588.49	3.00	1169.34
Freq.		29	1	30	29	30	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	0.33	0.00	0.00	0.00	0.33	0.00	-0.03	0.00	0.00	0.00	-1.11
07 - Best	1800	10.00	1125.22	10.00	828.94	11.00	1235.00	2.00	916.14	3.00	588.29	3.00	1077.65
Freq.		7	1	30	28	30	1	10	1	30	30	30	1
% to DACS 03	1800	0.00	-0.59	0.00	0.00	0.00	-0.12	-33.33	12.28	0.00	0.00	0.00	-0.97
08 - Best	1800	9.00	981.87	10.00	828.94	10.00	1146.66	2.00	730.11	3.00	588.32	3.00	838.95
Freq.		2	1	30	29	23	1	30	1	30	30	30	1
% to DACS 03	1800	-10.00	4.15	0.00	0.00	0.00	0.36	0.00	0.13	0.00	0.00	0.00	-0.93
09 - Best	1800	11.00	1215.46	10.00	828.94	-	-	3.00	930.85	-	-	-	-
Freq.		6	1	30	30	-	-	30	1	-	-	-	-
% to DACS 03	1800	-8.33	4.85	0.00	0.00	-	-	0.00	0.47	-	-	-	-
10 - Best	1800	10.00	1142.25	-	-	-	-	3.00	950.06	-	-	-	-
Freq.		1	1	-	-	-	-	30	1	-	-	-	-
% to DACS 03	1800	0.00	-3.18	-	-	-	-	0.00	0.07	-	-	-	-
11 - Best	1800	11.00	1069.64	-	-	-	-	2.00	915.36	-	-	-	-
Freq.		30	1	-	-	-	-	3	1	-	-	-	-
% to DACS 03	1800	0.00	0.01	-	-	-	-	-33.33	17.52	-	-	-	-
12 - Best	1800	9.00	1024.99	-	-	-	-	-	-	-	-	-	-
Freq.		1	1	-	-	-	-	-	-	-	-	-	-
% to DACS 03	1800	-10.00	6.81	-	-	-	-	-	-	-	-	-	-
DACS+HLS - AVGs	1800	12.08	1220.83	10.00	828.38	11.88	1375.13	2.73	965.38	3.00	589.86	3.25	1135.09
% to DACS 03	1800	-2.03	0.70	0.00	0.00	0.00	0.25	-9.09	3.12	0.00	0.00	-3.70	2.08
% to SA+LNS [29]	1800	-0.71	1.41	0.00	0.00	2.11	-0.28	-0.10	-1.52	0.00	0.00	0.00	-2.06
DACS 03 - AVGs	1800	12.33	1212.33	10.00	828.38	11.88	1371.68	3.00	936.17	3.00	589.86	3.38	1111.91
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table E.2: Comparison between the worst case performances of DACS+HLS and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	1800	19.00	1712.54	10.00	828.94	15.00	1763.10	4.00	1332.27	3.00	591.56	4.00	1596.25
Freq.		30	1	30	30	29	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	-0.93	0.00	0.00	0.00	-0.88	0.00	-0.23	0.00	0.00	0.00	1.76
02 - Worst	1800	18.00	1490.40	10.00	949.02	13.00	1587.33	3.00	1267.16	3.00	591.56	4.00	1272.16
Freq.		9	1	30	1	30	1	30	1	30	30	20	1
% to DACS 03	1800	0.00	0.76	0.00	7.01	0.00	1.15	-25.00	12.78	0.00	0.00	0.00	-0.05
03 - Worst	1800	14.00	1240.75	10.00	917.67	12.00	1335.65	3.00	1015.36	3.00	799.39	3.00	1225.13
Freq.		11	1	30	1	3	1	30	1	30	1	30	1
% to DACS 03	1800	0.00	0.23	0.00	6.78	0.00	-0.72	0.00	1.14	0.00	33.11	0.00	0.17
04 - Worst	1800	10.00	1031.49	10.00	828.98	10.00	1216.54	3.00	785.27	3.00	749.78	3.00	861.99
Freq.		30	1	30	1	30	1	18	1	30	1	30	1
% to DACS 03	1800	-9.09	1.62	0.00	0.11	-9.09	3.27	0.00	-0.95	0.00	10.25	0.00	0.36
05 - Worst	1800	14.00	1486.89	10.00	873.28	15.00	1554.49	3.00	1117.91	3.00	588.88	4.00	1484.91
Freq.		30	1	30	1	1	1	30	1	30	30	30	1
% to DACS 03	1800	-6.67	2.34	0.00	1.27	0.00	-2.90	0.00	-1.20	0.00	0.00	0.00	2.10
06 - Worst	1800	13.00	1264.91	10.00	872.30	12.00	1492.19	3.00	991.07	3.00	588.49	3.00	1384.60
Freq.		1	1	30	1	30	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	-2.55	0.00	5.23	-7.69	6.55	0.00	1.58	0.00	0.00	-25.00	23.70
07 - Worst	1800	11.00	1122.86	10.00	863.70	11.00	1401.69	3.00	874.85	3.00	588.29	3.00	1260.44
Freq.		23	1	30	2	30	1	20	1	30	30	30	1
% to DACS 03	1800	0.00	-1.91	0.00	0.00	-8.33	9.96	0.00	1.13	0.00	-0.58	-25.00	25.40
08 - Worst	1800	10.00	981.62	10.00	863.27	11.00	1165.74	2.00	811.97	3.00	588.32	3.00	919.90
Freq.		28	1	30	1	7	1	30	1	30	30	30	1
% to DACS 03	1800	0.00	-0.61	0.00	-5.75	0.00	-0.60	-33.33	10.66	0.00	0.00	0.00	-2.91
09 - Worst	1800	12.00	1288.61	10.00	828.94	-	-	3.00	995.71	-	-	-	-
Freq.		24	1	30	30	-	-	30	1	-	-	-	-
% to DACS 03	1800	0.00	3.76	0.00	0.00	-	-	0.00	0.59	-	-	-	-
10 - Worst	1800	11.00	1195.96	-	-	-	-	3.00	1004.64	-	-	-	-
Freq.		29	1	-	-	-	-	30	1	-	-	-	-
% to DACS 03	1800	-8.33	7.38	-	-	-	-	0.00	-2.62	-	-	-	-
11 - Worst	1800	11.00	1139.99	-	-	-	-	3.00	839.89	-	-	-	-
Freq.		30	1	-	-	-	-	27	1	-	-	-	-
% to DACS 03	1800	-8.33	3.11	-	-	-	-	0.00	1.15	-	-	-	-
12 - Worst	1800	10.00	1023.17	-	-	-	-	-	-	-	-	-	-
Freq.		29	1	-	-	-	-	-	-	-	-	-	-
% to DACS 03	1800	0.00	-0.96	-	-	-	-	-	-	-	-	-	-
DACS+HLS - AVGs	1800	12.75	1248.27	10.00	869.57	12.38	1439.59	3.00	1003.28	3.00	635.78	3.38	1250.67
% to DACS 03	1800	-2.55	0.95	0.00	1.60	-2.94	1.74	-5.71	2.06	0.00	5.50	-6.90	5.88
% to SA+LNS [29]	1800	1.35	3.83	0.00	4.97	4.17	4.76	-2.91	1.94	0.00	-3.48	-0.15	5.40
DACS 03 - AVGs	1800	13.08	1236.54	10.00	855.90	12.75	1415.02	3.18	983.05	3.00	602.65	3.63	1181.20
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table E.3: Comparison between the best case performances of DACS+HLS and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	2400	20.00	5426.41	20.00	2716.46	18.00	4321.22	5.00	4312.94	6.00	1931.44	6.00	3364.34
Freq.		3	1	3	1	1	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	1.84	0.00	0.44	-5.26	17.55	0.00	2.23	0.00	0.00	0.00	4.23
02 - Best	2400	18.00	4744.19	18.00	3140.78	18.00	3828.50	4.00	3902.09	6.00	1863.16	5.00	2985.55
Freq.		3	1	3	1	3	1	3	1	3	2	3	1
% to DACS 03	2400	0.00	3.27	0.00	3.40	0.00	3.58	0.00	4.70	0.00	0.00	0.00	1.24
03 - Best	2400	18.00	3673.13	18.00	2912.88	18.00	3673.61	4.00	3135.36	6.00	1959.33	4.00	2733.24
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	-3.00	0.00	-0.58	0.00	1.94	0.00	3.62	0.00	6.56	0.00	0.66
04 - Best	2400	18.00	3374.99	18.00	2866.08	18.00	3355.44	4.00	2135.47	6.00	1835.86	4.00	2279.39
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	2.69	0.00	2.01	0.00	3.20	0.00	0.83	0.00	-1.47	0.00	3.38
05 - Best	2400	18.00	4961.26	20.00	2702.05	18.00	4152.20	4.00	3563.78	6.00	1909.56	5.00	2932.05
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	4.01	0.00	0.00	-5.26	18.93	0.00	-3.44	0.00	1.38	0.00	2.53
06 - Best	2400	18.00	4049.35	20.00	2701.04	18.00	4055.16	4.00	3014.86	6.00	1895.09	5.00	2839.97
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	0.67	0.00	0.00	0.00	-4.45	0.00	-2.13	0.00	1.27	0.00	0.14
07 - Best	2400	18.00	3526.69	20.00	2731.25	18.00	4145.20	4.00	2653.46	6.00	1888.15	4.00	2747.14
Freq.		3	1	3	2	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	0.88	0.00	1.12	0.00	4.42	0.00	-0.19	0.00	-0.72	0.00	-9.41
08 - Best	2400	18.00	3321.59	19.00	3000.19	18.00	3627.28	4.00	2026.20	6.00	1861.28	4.00	2555.50
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	3.28	-5.00	11.29	0.00	-2.23	0.00	-0.87	0.00	0.32	0.00	1.48
09 - Best	2400	18.00	4494.20	18.00	2960.78	18.00	3596.40	4.00	3268.74	6.00	2054.26	4.00	2381.91
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	0.76	-5.26	9.31	0.00	1.07	0.00	2.05	0.00	10.43	0.00	0.37
10 - Best	2400	18.00	3749.22	18.00	3074.21	18.00	3321.34	4.00	2826.42	6.00	1826.91	4.00	2183.44
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	3.06	0.00	6.53	0.00	-4.19	0.00	2.08	0.00	-0.63	0.00	-3.54
DACS+HLS - AVGs	2400	18.20	4132.10	18.90	2880.57	18.00	3807.64	4.10	3083.93	6.00	1902.50	4.50	2700.25
% to DACS 03	2400	0.00	1.78	-1.05	3.34	-1.10	3.84	0.00	1.00	0.00	1.70	0.00	0.08
% to RVNSc [5]	720-1680	0.55	8.13	0.00	3.66	0.00	8.54	2.50	1.27	0.00	3.26	2.27	2.74
% to ES4C [2]	2400	0.00	11.53	0.00	3.54	0.00	7.11	2.50	0.95	0.00	3.06	4.65	0.94
DACS 03 - AVGs	2400	18.20	4059.77	19.10	2787.60	18.20	3666.73	4.10	3053.53	6.00	1870.73	4.50	2698.01
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table E.4: Comparison between the worst case performances of DACS+HLS and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	2400	20.00	5465.07	20.00	2731.36	19.00	3789.08	5.00	4356.93	6.00	1996.86	6.00	3371.60
Freq.		3	1	3	1	2	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	-0.93	0.00	0.99	0.00	-2.00	0.00	-1.90	0.00	-0.89	0.00	-1.91
02 - Worst	2400	18.00	4758.08	18.00	3228.19	18.00	4143.18	4.00	4049.49	6.00	2000.62	5.00	3185.03
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	-5.26	12.30	-5.26	9.60	0.00	9.12	-20.00	14.14	0.00	5.01	0.00	4.96
03 - Worst	2400	18.00	3986.19	18.00	3210.79	18.00	3834.87	4.00	3224.24	6.00	2038.49	4.00	2774.20
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	4.73	0.00	1.07	0.00	3.37	0.00	4.90	0.00	9.00	0.00	-1.59
04 - Worst	2400	18.00	3509.64	18.00	2943.19	18.00	3550.35	4.00	2232.37	6.00	1875.45	4.00	2343.29
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	0.78	0.00	-2.45	0.00	2.91	0.00	2.33	-14.29	-1.21	0.00	-2.05
05 - Worst	2400	18.00	5222.65	20.00	2713.94	18.00	4651.84	4.00	3690.22	6.00	1920.63	5.00	3014.51
Freq.		3	1	3	2	3	1	3	1	3	1	3	1
% to DACS 03	2400	-5.26	19.86	0.00	0.00	-5.26	27.98	0.00	-5.64	0.00	-1.23	0.00	-0.72
06 - Worst	2400	18.00	4323.10	20.00	2941.11	18.00	4337.81	4.00	3183.40	6.00	2055.93	5.00	2967.75
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	3.14	0.00	4.65	-5.26	20.24	0.00	2.36	0.00	6.51	0.00	2.39
07 - Worst	2400	18.00	4060.41	20.00	2748.45	18.00	4338.20	4.00	2735.03	6.00	2037.88	4.00	2963.95
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	14.16	0.00	1.31	0.00	4.39	0.00	-3.95	0.00	1.15	-20.00	7.76
08 - Worst	2400	18.00	3365.30	20.00	2698.59	18.00	3893.36	4.00	2091.18	6.00	1888.71	4.00	2629.45
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	0.74	0.00	0.00	0.00	-1.42	0.00	-3.04	0.00	-0.23	0.00	-1.08
09 - Worst	2400	18.00	4699.25	19.00	2814.80	18.00	3748.47	4.00	3299.85	6.00	2253.75	4.00	2475.59
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	1.24	0.00	1.19	0.00	0.28	0.00	1.76	0.00	11.94	0.00	-1.22
10 - Worst	2400	18.00	3925.87	18.00	3328.05	18.00	3756.12	4.00	2936.51	6.00	1948.96	4.00	2408.98
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	2400	0.00	3.54	-5.26	23.05	0.00	4.75	0.00	1.75	0.00	3.14	0.00	1.82
DACS+HLS - AVGs	2400	18.20	4331.56	19.10	2935.85	18.10	4004.33	4.10	3179.92	6.00	2001.73	4.50	2813.44
% to DACS 03	2400	-1.09	5.85	-1.04	3.86	-1.09	6.80	-2.38	1.28	-1.64	3.32	-2.17	0.84
% to RVNSc [5]	720-1680	0.55	13.35	1.06	5.65	0.56	14.15	2.50	4.42	0.00	8.65	2.27	7.04
% to ES4C [2]	2400	0.00	16.91	1.06	5.53	0.56	12.64	2.50	4.09	0.00	8.44	4.65	5.18
DACS 03 - AVGs	2400	18.40	4092.11	19.30	2826.61	18.30	3749.42	4.20	3139.77	6.10	1937.42	4.60	2789.88
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table E.5: Comparison between the best case performances of DACS+HLS and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	4800	40.00	11585.86	40.00	7155.32	37.00	9553.25	8.00	10235.71	12.00	4126.65	12.00	7815.71
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	1.65	0.00	-0.06	0.00	1.79	-11.11	6.10	0.00	-1.41	-7.69	5.04
02 - Best	4800	36.00	12252.53	37.00	9126.85	36.00	10684.60	8.00	8509.41	12.00	3972.11	11.00	7110.25
Freq.		3	1	2	1	2	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	20.43	0.00	6.29	-2.70	22.57	0.00	-0.46	0.00	-2.43	0.00	1.39
03 - Best	4800	36.00	10596.49	37.00	8749.94	36.00	10320.97	8.00	7074.06	12.00	4684.67	9.00	6261.67
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	16.18	0.00	-7.63	0.00	-1.02	0.00	1.21	0.00	4.18	0.00	10.49
04 - Best	4800	36.00	9386.43	36.00	9301.43	36.00	9065.76	8.00	5371.46	12.00	4942.86	8.00	4471.13
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	0.81	-2.70	14.26	0.00	-2.64	0.00	-0.17	0.00	-12.86	0.00	0.91
05 - Best	4800	36.00	12198.59	40.00	7170.68	37.00	9182.58	8.00	8147.83	12.00	4116.28	11.00	6785.34
Freq.		1	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	15.43	0.00	-0.32	0.00	-0.24	0.00	1.48	0.00	3.07	0.00	-2.79
06 - Best	4800	36.00	11317.20	40.00	7153.53	36.00	12133.34	8.00	7078.25	12.00	4049.36	9.00	7015.00
Freq.		3	1	3	1	1	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-7.99	0.00	-1.21	-2.70	32.49	0.00	1.27	0.00	1.92	-10.00	1.81
07 - Best	4800	36.00	10329.92	40.00	7241.02	37.00	9402.78	8.00	6275.83	12.00	4106.80	9.00	6613.62
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-5.09	0.00	0.63	0.00	2.02	0.00	1.73	0.00	0.23	0.00	5.67
08 - Best	4800	36.00	9011.46	39.00	7509.53	36.00	10611.74	8.00	5195.05	12.00	4389.33	8.00	5843.84
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-0.23	0.00	-5.81	-2.70	15.89	0.00	5.03	0.00	10.89	-11.11	4.25
09 - Best	4800	36.00	11815.60	37.00	7689.17	36.00	10225.51	8.00	7241.00	12.00	4334.11	8.00	5659.31
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	18.48	-2.63	6.12	-2.70	14.01	0.00	2.79	0.00	0.13	0.00	0.31
10 - Best	4800	36.00	10577.37	37.00	7447.92	36.00	9928.97	8.00	6726.26	12.00	4105.54	8.00	5403.43
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-4.24	0.00	-1.55	0.00	-4.45	0.00	-1.61	0.00	-0.15	0.00	-0.53
DACS+HLS - AVGs	4800	36.40	10907.15	38.30	7854.54	36.30	10110.95	8.00	7185.49	12.00	4282.77	9.30	6297.93
% to DACS 03	4800	-1.09	5.07	-0.52	0.99	-1.09	7.63	-1.23	1.81	0.00	-0.14	-3.12	2.64
% to RVNSc [5]	3900-7980	0.55	19.15	0.79	7.28	0.55	17.18	0.00	9.74	0.00	9.18	6.90	11.60
% to ES4C [2]	4800	0.28	22.21	0.79	3.57	0.55	15.38	0.00	10.51	0.00	8.84	8.14	14.13
DACS 03 - AVGs	4800	36.80	10380.56	38.50	7777.48	36.70	9393.75	8.10	7057.97	12.00	4288.86	9.60	6136.02
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table E.6: Comparison between the worst case performances of DACS+HLS and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	4800	40.00	11676.80	40.00	7165.45	37.00	9758.69	8.00	11022.99	12.00	4427.92	12.00	7988.84
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	0.36	0.00	-0.22	0.00	-1.02	-11.11	8.26	0.00	2.13	-7.69	3.98
02 - Worst	4800	36.00	13137.95	38.00	9143.62	37.00	8924.52	8.00	8842.02	12.00	4191.67	11.00	7484.49
Freq.		3	1	1	1	1	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	26.24	0.00	8.89	0.00	1.36	0.00	1.46	0.00	-1.66	-8.33	10.71
03 - Worst	4800	36.00	10826.52	37.00	9082.62	36.00	10539.08	8.00	7238.29	12.00	5425.62	9.00	6436.73
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	15.76	-2.63	5.12	-2.70	21.39	0.00	2.77	-7.69	28.88	-10.00	9.32
04 - Worst	4800	36.00	9536.65	36.00	9718.43	36.00	9550.70	8.00	5572.98	12.00	5224.72	8.00	4755.53
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-3.37	-2.70	15.32	0.00	-0.65	0.00	-2.69	-7.69	15.21	0.00	5.69
05 - Worst	4800	37.00	11027.20	40.00	7212.59	37.00	9615.27	8.00	8407.91	12.00	4619.10	11.00	7295.81
Freq.		2	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-0.87	0.00	-0.79	0.00	1.85	0.00	-3.30	0.00	7.72	-8.33	0.61
06 - Worst	4800	36.00	11942.82	40.00	7291.89	37.00	9538.15	8.00	7310.54	12.00	4613.03	9.00	7326.31
Freq.		3	1	3	1	2	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	21.29	0.00	0.10	0.00	2.05	0.00	-0.70	0.00	3.91	-10.00	4.16
07 - Worst	4800	36.00	10795.82	40.00	7544.60	37.00	9616.32	8.00	6505.58	12.00	4649.88	9.00	6939.03
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	19.61	0.00	1.49	0.00	2.35	0.00	-1.16	-7.69	10.56	-10.00	10.91
08 - Worst	4800	36.00	9346.16	40.00	7214.10	36.00	11034.48	8.00	5414.36	12.00	4400.52	8.00	5985.56
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	0.87	0.00	-1.10	-2.70	17.88	0.00	3.09	0.00	6.33	-11.11	-1.18
09 - Worst	4800	36.00	12791.74	37.00	8003.03	36.00	10600.00	8.00	7696.85	12.00	4866.09	8.00	5751.92
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	-2.70	28.07	-2.63	4.86	-2.70	12.77	0.00	3.54	-7.69	7.10	0.00	-2.00
10 - Worst	4800	36.00	11065.08	37.00	7872.26	36.00	10765.03	8.00	6964.91	12.00	4388.52	8.00	5590.91
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS 03	4800	0.00	-1.86	-2.63	5.09	-2.70	21.10	0.00	0.74	0.00	4.71	0.00	1.81
DACS+HLS - AVGs	4800	36.50	11214.67	38.50	8024.86	36.50	9994.22	8.00	7497.64	12.00	4680.71	9.30	6555.51
% to DACS 03	4800	-1.35	10.18	-1.03	4.15	-1.08	7.71	-1.23	1.45	-3.23	8.48	-7.00	4.40
% to RVNSc [5]	3900-7980	0.83	22.50	1.32	9.60	1.11	15.82	0.00	14.51	0.00	19.32	6.90	16.37
% to ES4C [2]	4800	0.55	25.65	1.32	5.81	1.11	14.05	0.00	15.31	0.00	18.95	8.14	18.80
DACS 03 - AVGs	4800	37.00	10178.74	38.90	7705.32	36.90	9279.23	8.10	7390.35	12.40	4314.93	10.00	6279.08
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Appendix F

Tables of results related to the system DACS+HLS+2-Opt

This section mentions in Tables F.1 to F.6 information about the experiments done to the system DACS+HLS+2-Opt, which uses the 2-Opt move variant as in Section 4.7.4. All tables, collectively, represent the best and worst case performances of DACS+HLS+2-Opt, on the problem groups PG100, PG200 and PG400 in Section 2.2, after thirty or three runs of 1800, 2400 or 4800 seconds.

Table F.1: Comparison between the best case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	1800	19.00	1650.80	10.00	828.94	14.00	1762.39	4.00	1253.26	3.00	591.56	4.00	1413.52
Freq.		30	1	30	30	1	1	30	3	30	30	30	4
% to DACS+HLS	1800	0.00	-0.17	0.00	0.00	0.00	1.73	0.00	-0.57	0.00	0.00	0.00	-0.59
02 - Best	1800	17.00	1488.09	10.00	828.94	13.00	1479.61	3.00	1192.93	3.00	591.56	3.00	1371.07
Freq.		24	1	30	29	30	1	27	1	30	30	9	1
% to DACS+HLS	1800	0.00	-0.21	0.00	0.00	0.00	-1.08	0.00	0.10	0.00	0.00	0.00	0.21
03 - Best	1800	13.00	1293.69	10.00	828.06	11.00	1265.15	3.00	943.20	3.00	591.17	3.00	1073.02
Freq.		24	1	30	25	30	1	30	1	30	29	30	1
% to DACS+HLS	1800	0.00	-0.18	0.00	0.00	0.00	-0.04	0.00	-0.89	0.00	0.00	0.00	-0.95
04 - Best	1800	10.00	984.45	10.00	824.78	10.00	1139.69	2.00	858.03	3.00	590.60	3.00	800.46
Freq.		30	1	30	28	30	1	6	1	30	7	30	1
% to DACS+HLS	1800	0.00	-0.25	0.00	0.00	0.00	-0.28	0.00	1.02	0.00	0.00	0.00	-0.17
05 - Best	1800	14.00	1399.71	10.00	828.94	14.00	1549.14	3.00	1007.57	3.00	588.88	4.00	1306.41
Freq.		30	1	30	30	29	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	0.20	0.00	0.00	0.00	-1.86	0.00	-0.22	0.00	0.00	0.00	-0.99
06 - Best	1800	12.00	1258.90	10.00	828.94	12.00	1403.94	3.00	907.69	3.00	588.49	3.00	1156.10
Freq.		30	1	30	30	30	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	-0.56	0.00	0.00	0.00	-0.01	0.00	-0.66	0.00	0.00	0.00	-1.13
07 - Best	1800	10.00	1122.05	10.00	828.94	11.00	1233.21	2.00	912.09	3.00	588.29	3.00	1093.53
Freq.		15	1	30	30	30	1	9	1	30	30	30	1
% to DACS+HLS	1800	0.00	-0.28	0.00	0.00	0.00	-0.14	0.00	-0.44	0.00	0.00	0.00	1.47
08 - Best	1800	9.00	969.66	10.00	828.94	10.00	1154.43	2.00	729.04	3.00	588.32	3.00	832.36
Freq.		2	1	30	30	30	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	-1.24	0.00	0.00	0.00	0.68	0.00	-0.15	0.00	0.00	0.00	-0.79
09 - Best	1800	11.00	1208.39	10.00	828.94	-	-	3.00	927.63	-	-	-	-
Freq.		17	1	30	30	-	-	30	1	-	-	-	-
% to DACS+HLS	1800	0.00	-0.58	0.00	0.00	-	-	0.00	-0.35	-	-	-	-
10 - Best	1800	10.00	1176.39	-	-	-	-	3.00	951.37	-	-	-	-
Freq.		3	1	-	-	-	-	30	1	-	-	-	-
% to DACS+HLS	1800	0.00	2.99	-	-	-	-	0.00	0.14	-	-	-	-
11 - Best	1800	10.00	1175.77	-	-	-	-	2.00	914.69	-	-	-	-
Freq.		1	1	-	-	-	-	4	1	-	-	-	-
% to DACS+HLS	1800	-9.09	9.92	-	-	-	-	0.00	-0.07	-	-	-	-
12 - Best	1800	10.00	967.28	-	-	-	-	-	-	-	-	-	-
Freq.		30	1	-	-	-	-	-	-	-	-	-	-
% to DACS+HLS	1800	11.11	-5.63	-	-	-	-	-	-	-	-	-	-
DACS+HLS+2-Opt - AVGs	1800	12.08	1224.60	10.00	828.38	11.88	1373.45	2.73	963.41	3.00	589.86	3.25	1130.81
% to DACS+HLS	1800	0.00	0.31	0.00	0.00	0.00	-0.12	0.00	-0.20	0.00	0.00	0.00	-0.38
% to SA+LNS [29]	1800	-0.71	1.72	0.00	0.00	2.11	-0.40	-0.10	-1.72	0.00	0.00	0.00	-2.42
DACS+HLS - AVGs	1800	12.08	1220.83	10.00	828.38	11.88	1375.13	2.73	965.38	3.00	589.86	3.25	1135.09
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91
	7200	11.92	1213.25	10.00	828.38	11.50	1384.22	2.73	966.37	3.00	589.86	3.25	1141.24
HGA [28] - AVGs	1800	12.17	1230.22	10.00	828.48	11.75	1397.63	2.73	1009.53	3.00	589.93	3.25	1230.20
HGA+TA [28] - AVGs	2094	12.17	1208.57	10.00	828.38	11.75	1372.93	2.73	971.44	3.00	589.86	3.25	1154.04
LS [28] - AVGs	126	12.00	1235.22	10.00	828.38	11.50	1413.50	2.73	979.88	3.00	589.93	3.25	1152.37
LS+TA [28]- AVGs	156	12.00	1220.20	10.00	828.38	11.50	1398.76	2.73	970.38	3.00	589.86	3.25	1139.37

Table F.2: Comparison between the worst case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after thirty runs of 1800 seconds, on the problem group PG100.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	1800	19.00	1692.52	10.00	828.94	15.00	1757.58	4.00	1284.55	3.00	591.56	4.00	1496.48
Freq.		30	1	30	30	29	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	-1.17	0.00	0.00	0.00	-0.31	0.00	-3.58	0.00	0.00	0.00	-6.25
02 - Worst	1800	18.00	1475.42	10.00	834.64	13.00	1534.30	4.00	1096.31	3.00	591.56	4.00	1177.65
Freq.		6	1	30	1	30	1	3	1	30	30	21	1
% to DACS+HLS	1800	0.00	-1.01	0.00	-12.05	0.00	-3.34	33.33	-13.48	0.00	0.00	0.00	-7.43
03 - Worst	1800	14.00	1226.25	10.00	834.56	11.00	1417.98	3.00	978.23	3.00	600.21	3.00	1172.39
Freq.		6	1	30	1	30	1	30	1	30	1	30	1
% to DACS+HLS	1800	0.00	-1.17	0.00	-9.06	-8.33	6.16	0.00	-3.66	0.00	-24.92	0.00	-4.30
04 - Worst	1800	10.00	1024.77	10.00	825.54	10.00	1187.83	3.00	780.43	3.00	623.20	3.00	857.27
Freq.		30	1	30	2	30	1	24	1	30	1	30	1
% to DACS+HLS	1800	0.00	-0.65	0.00	-0.41	0.00	-2.36	0.00	-0.62	0.00	-16.88	0.00	-0.55
05 - Worst	1800	14.00	1463.47	10.00	828.94	15.00	1549.07	3.00	1089.29	3.00	588.88	4.00	1391.36
Freq.		30	1	30	30	1	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	-1.58	0.00	-5.08	0.00	-0.35	0.00	-2.56	0.00	0.00	0.00	-6.30
06 - Worst	1800	12.00	1306.78	10.00	828.94	12.00	1480.90	3.00	955.94	3.00	588.49	3.00	1285.85
Freq.		30	1	30	30	30	1	30	1	30	30	30	1
% to DACS+HLS	1800	-7.69	3.31	0.00	-4.97	0.00	-0.76	0.00	-3.54	0.00	0.00	0.00	-7.13
07 - Worst	1800	11.00	1125.23	10.00	828.94	11.00	1318.05	3.00	842.60	3.00	588.29	3.00	1195.51
Freq.		15	1	30	30	30	1	21	1	30	30	30	1
% to DACS+HLS	1800	0.00	0.21	0.00	-4.02	0.00	-5.97	0.00	-3.69	0.00	0.00	0.00	-5.15
08 - Worst	1800	10.00	980.67	10.00	828.94	10.00	1223.27	2.00	793.30	3.00	588.32	3.00	906.37
Freq.		28	1	30	30	30	1	30	1	30	30	30	1
% to DACS+HLS	1800	0.00	-0.10	0.00	-3.98	-9.09	4.94	0.00	-2.30	0.00	0.00	0.00	-1.47
09 - Worst	1800	12.00	1256.98	10.00	828.94	-	-	3.00	989.29	-	-	-	-
Freq.		13	1	30	30	-	-	30	1	-	-	-	-
% to DACS+HLS	1800	0.00	-2.45	0.00	0.00	-	-	0.00	-0.64	-	-	-	-
10 - Worst	1800	11.00	1167.40	-	-	-	-	3.00	991.34	-	-	-	-
Freq.		27	1	-	-	-	-	30	1	-	-	-	-
% to DACS+HLS	1800	0.00	-2.39	-	-	-	-	0.00	-1.32	-	-	-	-
11 - Worst	1800	11.00	1148.89	-	-	-	-	3.00	823.34	-	-	-	-
Freq.		29	1	-	-	-	-	26	1	-	-	-	-
% to DACS+HLS	1800	0.00	0.76	-	-	-	-	0.00	-1.97	-	-	-	-
12 - Worst	1800	10.00	1064.55	-	-	-	-	-	-	-	-	-	-
Freq.		30	1	-	-	-	-	-	-	-	-	-	-
% to DACS+HLS	1800	0.00	4.04	-	-	-	-	-	-	-	-	-	-
DACS+HLS+2-Opt - AVGs	1800	12.67	1244.41	10.00	829.82	12.13	1433.62	3.09	965.87	3.00	595.06	3.38	1185.36
% to DACS+HLS	1800	-0.65	-0.31	0.00	-4.57	-2.02	-0.41	3.03	-3.73	0.00	-6.40	0.00	-5.22
% to SA+LNS [29]	1800	0.69	3.51	0.00	0.17	2.06	4.32	0.03	-1.86	0.00	-9.66	-0.15	-0.10
DACS+HLS - AVGs	1800	12.75	1248.27	10.00	869.57	12.38	1439.59	3.00	1003.28	3.00	635.78	3.38	1250.67
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57
	7200	12.17	1209.12	10.00	828.38	12.00	1358.73	2.73	1002.90	3.00	599.82	3.38	1152.74
HGA [28] - AVGs	1800	12.17	1254.11	10.00	828.97	12.00	1424.60	2.73	1041.24	3.00	606.98	3.25	1291.51
HGA+TA [28] - AVGs	2094	12.17	1230.54	10.00	828.38	12.00	1386.22	2.73	997.85	3.00	589.86	3.25	1203.97
LS [28] - AVGs	126	12.17	1253.04	10.00	828.38	11.63	1441.28	2.73	1013.22	3.00	590.30	3.25	1186.98
LS+TA [28] - AVGs	156	12.17	1224.70	10.00	828.38	11.63	1418.72	2.73	990.08	3.00	589.86	3.25	1159.81

Table F.3: Comparison between the best case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	2400	20.00	4977.50	20.00	2704.57	18.00	4347.55	5.00	4074.51	6.00	1931.44	6.00	3243.54
Freq.		3	1	3	3	1	1	3	1	3	3	3	1
% to DACS+HLS	2400	0.00	-8.27	0.00	-0.44	0.00	0.61	0.00	-5.53	0.00	0.00	0.00	-3.59
02 - Best	2400	18.00	4560.41	18.00	2976.44	18.00	3668.35	4.00	3726.03	6.00	1863.16	5.00	2908.85
Freq.		3	1	3	1	3	1	3	1	3	3	3	1
% to DACS+HLS	2400	0.00	-3.87	0.00	-5.23	0.00	-4.18	0.00	-4.51	0.00	0.00	0.00	-2.57
03 - Best	2400	18.00	3600.72	18.00	2826.72	18.00	3406.91	4.00	3078.96	6.00	1803.45	4.00	2711.72
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-1.97	0.00	-2.96	0.00	-7.26	0.00	-1.80	0.00	-7.96	0.00	-0.79
04 - Best	2400	18.00	3260.10	18.00	2806.35	18.00	3147.91	4.00	2096.55	6.00	1809.40	4.00	2235.22
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-3.40	0.00	-2.08	0.00	-6.18	0.00	-1.82	0.00	-1.44	0.00	-1.94
05 - Best	2400	18.00	4850.46	20.00	2702.05	18.00	4151.96	4.00	3404.88	6.00	1879.31	5.00	2795.51
Freq.		3	1	3	3	3	1	3	1	3	2	3	1
% to DACS+HLS	2400	0.00	-2.23	0.00	0.00	0.00	-0.01	0.00	-4.46	0.00	-1.58	0.00	-4.66
06 - Best	2400	18.00	3777.88	20.00	2701.04	18.00	3960.48	4.00	2986.99	6.00	1858.72	5.00	2699.63
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-6.70	0.00	0.00	0.00	-2.33	0.00	-0.92	0.00	-1.92	0.00	-4.94
07 - Best	2400	18.00	3417.39	20.00	2701.04	18.00	3842.06	4.00	2531.20	6.00	1850.43	4.00	2678.76
Freq.		3	1	3	3	3	1	3	1	3	2	3	1
% to DACS+HLS	2400	0.00	-3.10	0.00	-1.11	0.00	-7.31	0.00	-4.61	0.00	-2.00	0.00	-2.49
08 - Best	2400	18.00	3166.62	19.00	2831.86	18.00	3649.94	4.00	1963.55	6.00	1833.27	4.00	2373.66
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-4.67	0.00	-5.61	0.00	0.62	0.00	-3.09	0.00	-1.50	0.00	-7.12
09 - Best	2400	18.00	4269.48	18.00	3002.19	18.00	3630.69	4.00	3198.67	6.00	1836.53	4.00	2327.37
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-5.00	0.00	1.40	0.00	0.95	0.00	-2.14	0.00	-10.60	0.00	-2.29
10 - Best	2400	18.00	3546.95	18.00	2800.31	18.00	3407.53	4.00	2745.38	6.00	1813.68	4.00	2138.43
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-5.39	0.00	-8.91	0.00	2.60	0.00	-2.87	0.00	-0.71	0.00	-2.06
DACS+HLS+2-Opt - AVGs	2400	18.20	3942.75	18.90	2805.26	18.00	3721.34	4.10	2980.67	6.00	1847.96	4.50	2611.27
% to DACS+HLS	2400	0.00	-4.58	0.00	-2.61	0.00	-2.27	0.00	-3.35	0.00	-2.87	0.00	-3.30
% to RVNSc [5]	720-1680	0.55	3.17	0.00	0.95	0.00	6.08	2.50	-2.12	0.00	0.30	2.27	-0.65
% to ES4C [2]	2400	0.00	6.42	0.00	0.84	0.00	4.68	2.50	-2.43	0.00	0.11	4.65	-2.38
DACS+HLS - AVGs	2400	18.20	4132.10	18.90	2880.57	18.00	3807.64	4.10	3083.93	6.00	1902.50	4.50	2700.25
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LC03 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table F.4: Comparison between the worst case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 2400 seconds, on the problem group PG200.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	2400	20.00	5136.22	20.00	2704.57	19.00	3735.11	5.00	4099.88	6.00	1931.44	6.00	3273.30
Freq.		3	1	3	3	2	1	3	1	3	3	3	1
% to DACS+HLS	2400	0.00	-6.02	0.00	-0.98	0.00	-1.42	0.00	-5.90	0.00	-3.28	0.00	-2.92
02 - Worst	2400	18.00	4703.84	18.00	3141.66	18.00	3819.38	4.00	3816.52	6.00	1863.16	5.00	3025.85
Freq.		3	1	3	1	3	1	3	1	3	3	3	1
% to DACS+HLS	2400	0.00	-1.14	0.00	-2.68	0.00	-7.82	0.00	-5.75	0.00	-6.87	0.00	-5.00
03 - Worst	2400	18.00	3652.78	18.00	3001.25	18.00	3504.70	4.00	3180.89	6.00	1952.13	4.00	2883.27
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-8.36	0.00	-6.53	0.00	-8.61	0.00	-1.34	0.00	-4.24	0.00	3.93
04 - Worst	2400	18.00	3287.76	18.00	2833.91	18.00	3307.03	4.00	2118.49	6.00	1983.59	4.00	2339.39
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-6.32	0.00	-3.71	0.00	-6.85	0.00	-5.10	0.00	5.77	0.00	-0.17
05 - Worst	2400	18.00	4976.04	20.00	2702.05	18.00	4206.85	4.00	3497.88	6.00	1891.21	5.00	2834.10
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-4.72	0.00	-0.44	0.00	-9.57	0.00	-5.21	0.00	-1.53	0.00	-5.98
06 - Worst	2400	18.00	4007.28	20.00	2701.04	18.00	4218.15	4.00	3000.18	6.00	1870.07	5.00	2809.91
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-7.31	0.00	-8.16	0.00	-2.76	0.00	-5.76	0.00	-9.04	0.00	-5.32
07 - Worst	2400	18.00	3540.19	20.00	2701.04	18.00	4106.28	4.00	2734.58	6.00	1919.22	4.00	2903.18
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-12.81	0.00	-1.72	0.00	-5.35	0.00	-0.02	0.00	-5.82	0.00	-2.05
08 - Worst	2400	18.00	3256.02	20.00	2695.84	18.00	3721.66	4.00	2016.11	6.00	1836.41	4.00	2543.75
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-3.25	0.00	-0.10	0.00	-4.41	0.00	-3.59	0.00	-2.77	0.00	-3.26
09 - Worst	2400	18.00	4359.37	18.00	3284.55	18.00	3846.28	4.00	3286.38	6.00	1880.73	4.00	2342.85
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-7.23	-5.26	16.69	0.00	2.61	0.00	-0.41	0.00	-16.55	0.00	-5.36
10 - Worst	2400	18.00	3862.33	18.00	2985.48	18.00	3530.28	4.00	2878.46	6.00	1823.80	4.00	2213.64
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	2400	0.00	-1.62	0.00	-10.29	0.00	-6.01	0.00	-1.98	0.00	-6.42	0.00	-8.11
DACS+HLS+2-Opt - AVGs	2400	18.20	4078.18	19.00	2875.14	18.10	3799.57	4.10	3062.94	6.00	1895.18	4.50	2716.92
% to DACS+HLS	2400	0.00	-5.85	-0.52	-2.07	0.00	-5.11	0.00	-3.68	0.00	-5.32	0.00	-3.43
% to RVNSc [5]	720-1680	0.55	6.72	0.53	3.47	0.56	8.31	2.50	0.58	0.00	2.86	2.27	3.37
% to ES4C [2]	2400	0.00	10.07	0.53	3.35	0.56	6.88	2.50	0.26	0.00	2.66	4.65	1.57
DACS+HLS - AVGs	2400	18.20	4331.56	19.10	2935.85	18.10	4004.33	4.10	3179.92	6.00	2001.73	4.50	2813.44
RVNSc [5]	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36
ES4C [2]	2400	18.20	3705.00	18.90	2782.00	18.00	3555.00	4.00	3055.00	6.00	1846.00	4.30	2675.00
LCO3 [33]	3000	18.20	3676.95	18.90	2743.66	18.00	3449.71	4.00	2986.01	6.00	1836.10	4.30	2613.75
AGES [48]	480	18.20	3618.68	18.80	2717.21	18.00	3221.34	4.00	2942.92	6.00	1833.57	4.40	2519.79
MSLS1 [17]	102	18.20	3884.95	18.90	2791.15	18.00	3543.36	4.00	3081.61	6.00	1860.71	4.40	2672.01
MSLS+TA1 [17]	144	18.20	3718.30	18.90	2749.83	18.00	3329.62	4.00	3014.28	6.00	1842.65	4.40	2585.89

Table F.5: Comparison between the best case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	4800	40.00	10832.31	40.00	7152.06	37.00	9102.66	8.00	9717.59	12.00	4119.14	12.00	7231.45
Freq.		3	1	3	3	3	1	3	1	3	2	3	1
% to DACS+HLS	4800	0.00	-6.50	0.00	-0.05	0.00	-4.72	0.00	-5.06	0.00	-0.18	0.00	-7.48
02 - Best	4800	36.00	10804.26	37.00	7877.58	36.00	9671.91	8.00	7958.65	12.00	3937.42	11.00	6213.07
Freq.		3	1	2	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-11.82	0.00	-13.69	0.00	-9.48	0.00	-6.47	0.00	-0.87	0.00	-12.62
03 - Best	4800	36.00	9722.42	36.00	9606.86	36.00	9708.02	8.00	6624.42	12.00	4131.15	8.00	6080.16
Freq.		3	1	2	1	3	1	3	1	3	1	1	1
% to DACS+HLS	4800	0.00	-8.25	-2.70	9.79	0.00	-5.94	0.00	-6.36	0.00	-11.82	-11.11	-2.90
04 - Best	4800	36.00	8936.47	36.00	9207.60	36.00	8901.71	8.00	5006.27	12.00	4345.63	8.00	4116.55
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-4.79	0.00	-1.01	0.00	-1.81	0.00	-6.80	0.00	-12.08	0.00	-7.93
05 - Best	4800	36.00	11896.50	40.00	7152.06	37.00	8930.16	8.00	7566.10	12.00	3954.29	11.00	5928.58
Freq.		2	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-2.48	0.00	-0.26	0.00	-2.75	0.00	-7.14	0.00	-3.94	0.00	-12.63
06 - Best	4800	36.00	10601.18	40.00	7153.45	36.00	9995.94	8.00	6785.43	12.00	3893.36	9.00	6263.05
Freq.		3	1	3	3	2	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-6.33	0.00	0.00	0.00	-17.62	0.00	-4.14	0.00	-3.85	0.00	-10.72
07 - Best	4800	36.00	9516.08	40.00	7149.43	36.00	10486.79	8.00	5895.42	12.00	3966.24	8.00	6016.84
Freq.		3	1	3	3	3	1	3	1	3	1	2	1
% to DACS+HLS	4800	0.00	-7.88	0.00	-1.26	-2.70	11.53	0.00	-6.06	0.00	-3.42	-11.11	-9.02
08 - Best	4800	36.00	8852.03	39.00	7250.67	36.00	10023.92	8.00	4859.24	12.00	3877.71	8.00	5429.49
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-1.77	0.00	-3.45	0.00	-5.54	0.00	-6.46	0.00	-11.66	0.00	-7.09
09 - Best	4800	36.00	10934.82	37.00	7506.09	36.00	9840.55	8.00	6849.99	12.00	4299.57	8.00	5221.77
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-7.45	0.00	-2.38	0.00	-3.76	0.00	-5.40	0.00	-0.80	0.00	-7.73
10 - Best	4800	36.00	10097.53	37.00	7260.35	36.00	9703.43	8.00	6525.49	12.00	3783.90	8.00	4976.89
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-4.54	0.00	-2.52	0.00	-2.27	0.00	-2.98	0.00	-7.83	0.00	-7.89
DACS+HLS+2-Opt - AVGs	4800	36.40	10219.36	38.20	7731.62	36.20	9636.51	8.00	6778.86	12.00	4030.84	9.10	5747.79
% to DACS+HLS	4800	0.00	-6.31	-0.26	-1.57	-0.28	-4.69	0.00	-5.66	0.00	-5.88	-2.15	-8.74
% to RVNSc [5]	3900-7980	0.55	11.63	0.53	5.60	0.28	11.68	0.00	3.53	0.00	2.76	4.60	2.03
% to ES4C [2]	4800	0.28	14.50	0.53	1.95	0.28	9.97	0.00	4.26	0.00	2.44	5.81	4.16
DACS+HLS - AVGs	4800	36.40	10907.15	38.30	7854.54	36.30	10110.95	8.00	7185.49	12.00	4282.77	9.30	6297.93
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Table F.6: Comparison between the worst case performances of DACS+HLS+2-Opt and other VRPTW algorithms, after three runs of 4800 seconds, on the problem group PG400.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	4800	40.00	11014.89	40.00	7152.06	37.00	9285.80	8.00	9794.74	12.00	4119.33	12.00	7498.61
Freq.		3	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-5.67	0.00	-0.19	0.00	-4.85	0.00	-11.14	0.00	-6.97	0.00	-6.14
02 - Worst	4800	36.00	11528.50	38.00	7788.22	36.00	10979.68	8.00	8245.06	12.00	3985.52	11.00	6468.85
Freq.		3	1	1	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-12.25	0.00	-14.82	-2.70	23.03	0.00	-6.75	0.00	-4.92	0.00	-13.57
03 - Worst	4800	36.00	10119.00	37.00	7745.44	36.00	9865.79	8.00	6757.33	12.00	4344.59	9.00	5728.26
Freq.		3	1	1	1	3	1	3	1	3	1	2	1
% to DACS+HLS	4800	0.00	-6.54	0.00	-14.72	0.00	-6.39	0.00	-6.64	0.00	-19.92	0.00	-11.01
04 - Worst	4800	36.00	8951.96	36.00	9581.90	36.00	8988.70	8.00	5220.57	12.00	4848.79	8.00	4272.46
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-6.13	0.00	-1.40	0.00	-5.88	0.00	-6.32	0.00	-7.20	0.00	-10.16
05 - Worst	4800	37.00	10034.17	40.00	7152.06	37.00	9021.16	8.00	7690.05	12.00	4076.12	11.00	6246.61
Freq.		1	1	3	3	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-9.01	0.00	-0.84	0.00	-6.18	0.00	-8.54	0.00	-11.76	0.00	-14.38
06 - Worst	4800	36.00	11039.21	40.00	7153.45	37.00	8907.25	8.00	6950.28	12.00	3916.58	9.00	6643.16
Freq.		3	1	3	3	1	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-7.57	0.00	-1.90	0.00	-6.61	0.00	-4.93	0.00	-15.10	0.00	-9.32
07 - Worst	4800	36.00	10275.61	40.00	7149.43	36.00	11222.61	8.00	6220.81	12.00	4230.88	9.00	5961.55
Freq.		3	1	3	3	3	1	3	1	3	1	1	1
% to DACS+HLS	4800	0.00	-4.82	0.00	-5.24	-2.70	16.70	0.00	-4.38	0.00	-9.01	0.00	-14.09
08 - Worst	4800	36.00	8986.03	39.00	7460.22	36.00	10683.98	8.00	4952.50	12.00	3896.14	8.00	5546.46
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-3.85	-2.50	3.41	0.00	-3.18	0.00	-8.53	0.00	-11.46	0.00	-7.34
09 - Worst	4800	36.00	11902.13	37.00	7851.19	36.00	10380.05	8.00	7060.64	12.00	4632.72	8.00	5317.48
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-6.95	0.00	-1.90	0.00	-2.08	0.00	-8.27	0.00	-4.80	0.00	-7.55
10 - Worst	4800	36.00	10390.17	37.00	7642.05	36.00	10145.29	8.00	6649.17	12.00	3867.99	8.00	5018.05
Freq.		3	1	3	1	3	1	3	1	3	1	3	1
% to DACS+HLS	4800	0.00	-6.10	0.00	-2.92	0.00	-5.76	0.00	-4.53	0.00	-11.86	0.00	-10.25
DACS+HLS+2-Opt - AVGs	4800	36.50	10424.17	38.40	7667.60	36.30	9948.03	8.00	6954.12	12.00	4191.87	9.30	5870.15
% to DACS+HLS	4800	0.00	-7.05	-0.26	-4.45	-0.55	-0.46	0.00	-7.25	0.00	-10.44	0.00	-10.45
% to RVNSc [5]	3900-7980	0.83	13.67	1.05	4.72	0.55	15.29	0.00	6.20	0.00	6.86	6.90	4.20
% to ES4C [2]	4800	0.55	16.80	1.05	1.10	0.55	13.52	0.00	6.95	0.00	6.53	8.14	6.38
DACS+HLS - AVGs	4800	36.50	11214.67	38.50	8024.86	36.50	9994.22	8.00	7497.64	12.00	4680.71	9.30	6555.51
RVNSc [5]	3900-7980	36.20	9154.50	38.00	7321.68	36.10	8628.74	8.00	6547.87	12.00	3922.71	8.70	5633.28
ES4C [2]	4800	36.30	8925.00	38.00	7584.00	36.10	8763.00	8.00	6502.00	12.00	3935.00	8.60	5518.00
LC03 [33]	6000	36.50	8839.28	37.90	7447.09	36.00	8652.01	8.00	6437.68	12.00	3940.87	8.60	5511.22
AGES [48]	1020	36.30	8530.03	37.90	7148.27	36.00	8066.44	8.00	6209.94	12.00	3840.85	8.80	5243.06
MSLS1 [17]	408	36.40	9225.95	37.90	7464.09	36.00	8836.49	8.00	6690.15	12.00	3984.57	8.90	5692.33
MSLS+TA1 [17]	474	36.40	8692.17	37.90	7230.48	36.00	8305.55	8.00	6382.63	12.00	3894.48	8.90	5407.87

Appendix G

Tables of results related to different SI1-Like algorithms

This section mentions in Tables G.1 to G.15 information about the experiments done to some SI1-Like approaches. Each of the tables, mentioned in C1 to C6, represents the average, best and worst case performance, on the problem group PG100 in Section 2.2, after only one run of an averaged amount of CPU time that is less than or equal to 235.09 seconds.

C1- Table G.1 represents the performance of the SI1-Like 05 approach as in Section 5.8, which its insertion procedure uses a descending ordering of customers according to how much constrained they are.

C2- Table G.2 represents the performance of the SI1-Like 07 approach as in Section 5.10, which its hybrid local search HLS sorts the tours of each solution, to be improved, in an ascending order according their sizes of the customers they have.

C3- Tables G.3 and G.4 represent respectively the performances of the SI1-Like 08 and SI1-Like 09 approaches as in Section 5.11. SI1-Like 08 uses the inversion move of type 1 only whereas SI1-Like 09 uses the inversion move of type 2 in addition to that of type 1.

C4- Tables G.5, G.6, G.7 and G.8 represent respectively the performances of the SI1-Like 10, SI1-Like 11, SI1-Like 12 and SI1-Like 13 approaches as in

Section 5.12. Each of the four SI1-Like approaches uses a particular region for ejection and insertion purposes in the "eject and insert" strategy.

a- SI1-Like 10 uses the sector region in the "eject and insert" strategy.

b- SI1-Like 11 uses the track region in the "eject and insert" strategy.

c- SI1-Like 12 uses the time intervals in the "eject and insert" strategy.

d- SI1-Like 13 uses the whole graph in the "eject and insert" strategy.

C5- Tables G.9, G.10 and G.11 represent respectively the performances of the SI1-Like 14, SI1-Like 15, and SI1-Like 16 approaches as in Section 5.13. Each of the three SI1-Like approaches merges the "eject and insert" strategy with the insertion procedure and the hybrid local search HLS in Section 5.3.4.

e- SI1-Like 14 uses the HLS to improve the feasible solutions out of the "eject and insert" strategy.

f- SI1-Like 15 and SI1-Like 16 use the insertion procedure and the HLS to improve the feasible and the infeasible solutions out of the "eject and insert" strategy. The difference between SI1-Like 15 and SI1-Like 16 is that the first does not apply the steps h and i in Figure 5.2 while the second uses such steps.

C6- Tables G.12, G.13, G.14 and G.15 represent respectively the performances of the SI1-Like 17, SI1-Like 18, SI1-Like 19 and SI1-Like 20 approaches as in Section 5.14. Each of the four SI1-Like approaches uses the waiting time functions in Equation 5.3, 5.4 or 5.5 instead of the servicing time functions Equation 2.25.

g- SI1-Like 17 uses the vehicle waiting time functions in Equation 5.3.

h- SI1-Like 18 uses the customer waiting time functions in Equation 5.4.

i- SI1-Like 19 uses a combination of the vehicle and customer waiting time functions in Equation 5.5.

j- SI1-Like 20 uses Equation 5.5 as in SI1-Like 19 but with panelizing the vehicle waiting time terms with the value -1.

Table G.1: Comparison between the average case performances of SI1-Like 05 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	15.22 - 114.55	20.00	1722.93	10.00	833.24	15.00	1754.91	4.00	1456.35	3.00	591.56	4.00	1597.21
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
02 - AVGs	15.22 - 114.55	18.00	1579.59	10.00	978.30	14.00	1617.26	4.00	1441.55	3.00	591.56	4.00	1642.33
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	1.04	0.00	0.00	0.00	7.87	0.00	0.00	0.00	0.23
03 - AVGs	15.22 - 114.55	14.00	1422.88	10.00	1037.99	12.00	1425.88	3.00	1298.97	3.00	737.92	3.00	1231.66
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	3.21	0.00	0.00	0.00	19.15	0.00	3.18	0.00	-0.07
04 - AVGs	15.22 - 114.55	10.00	1128.72	10.00	1134.71	11.00	1273.91	3.00	1027.94	3.00	859.94	3.00	1179.59
% to SI1-Like 04	15.78 - 137.64	0.00	-0.72	0.00	0.00	0.00	-1.51	0.00	0.00	0.00	5.35	0.00	14.24
05 - AVGs	15.22 - 114.55	14.00	1439.41	10.00	832.27	15.00	1704.79	3.00	1199.94	3.00	606.28	4.00	1475.95
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-4.94
06 - AVGs	15.22 - 114.55	13.00	1354.97	10.00	828.94	12.00	1521.72	3.00	1168.83	3.00	588.49	3.00	1335.46
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
07 - AVGs	15.22 - 114.55	11.00	1392.23	10.00	828.94	12.00	1362.18	3.00	1137.94	3.00	606.06	3.00	1364.75
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
08 - AVGs	15.22 - 114.55	10.00	1059.00	10.00	853.25	11.00	1254.11	2.00	919.66	3.00	615.08	3.00	1156.87
% to SI1-Like 04	15.78 - 137.64	0.00	-0.47	0.00	0.00	0.00	0.00	0.00	1.14	0.00	0.00	0.00	0.00
09 - AVGs	15.22 - 114.55	12.00	1281.02	10.00	884.16	-	-	3.00	1153.82	-	-	-	-
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	0.00	0.00	-	-	0.00	3.64	-	-	-	-
10 - AVGs	15.22 - 114.55	12.00	1230.79	-	-	-	-	3.00	1132.10	-	-	-	-
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	15.22 - 114.55	11.00	1226.06	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	15.22 - 114.55	10.00	1084.49	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 04	15.78 - 137.64	0.00	0.00	-	-	-	-	-	-	-	-	-	-
SI1-Like 05 -AVGs	15.22 - 114.55	12.92	1326.84	10.00	912.42	12.75	1489.35	3.09	1177.68	3.00	649.61	3.38	1372.98
Time(secs.)	15.22 - 114.55	18.00	-	15.22	-	16.62	-	114.55	-	61.75	-	95.00	-
% to SI1-Like 04	15.78 - 137.64	0.00	-0.08	0.00	0.52	0.00	-0.16	0.00	2.90	0.00	1.29	0.00	0.67
% to SI1-Like 03	5.88 - 46	-1.90	-7.28	0.00	-3.42	-2.86	-4.74	-2.86	-7.90	0.00	-5.81	-6.90	-11.15
% to RVNSa [5]	2220	7.64	7.92	0.00	10.15	10.87	6.82	13.22	19.00	0.00	10.05	3.85	20.32
% to PR1 [11]	1176	-3.10	-12.07	-6.28	-32.10	-4.71	-13.60	0.03	-15.07	-11.24	-18.55	-7.02	-16.84
% to TP [13]	108	-0.64	-2.22	0.00	-0.46	-1.92	-1.65	-2.80	-7.71	0.00	0.77	-9.03	-16.00
% to DACS 01	300 - 400	-8.28	-12.87	0.00	-6.60	-7.27	-12.12	-2.86	-12.62	0.00	-6.37	-8.99	-13.27
% to DACS 02	300	-2.52	6.56	0.00	6.60	-0.97	4.72	-2.86	10.71	-2.70	6.33	-7.95	7.83
% to DACS 03	300	1.17	7.68	0.00	9.70	2.34	6.09	-1.26	22.00	0.00	9.70	-2.41	18.11
% to DACS+HLS	300	2.51	7.16	0.00	9.38	5.05	5.74	3.03	20.29	0.00	9.50	0.12	17.01
% to DACS+HLS+2-Opt	300	2.22	8.38	0.00	10.08	5.26	6.72	2.20	22.79	0.00	9.86	0.12	20.19
SI1-Like 04 - AVGs	15.78 - 137.64	12.92	1327.94	10.00	907.72	12.75	1491.79	3.09	1144.51	3.00	641.31	3.38	1363.81
SI1-Like 03 - AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
PR1 [11] - AVGs	1176	13.33	1509.04	10.67	1343.69	13.38	1723.72	3.09	1386.67	3.38	797.59	3.63	1651.05
TP [13] - AVGs	108	13.00	1356.92	10.00	916.67	13.00	1514.29	3.18	1276	3.00	644.63	3.71	1634.43
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.2: Comparison between the average case performances of SI1-Like 07 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	19 - 144.09	20.00	1680.20	10.00	828.94	15.00	1701.50	4.00	1341.49	3.00	591.56	4.00	1469.28
% to SI1-Like 06	19.75 - 141.91	0.00	-0.94	0.00	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00	-3.13
02 - AVGs	19 - 144.09	18.00	1569.54	10.00	916.66	14.00	1576.84	4.00	1173.21	3.00	591.56	4.00	1240.88
% to SI1-Like 06	19.75 - 141.91	0.00	0.26	0.00	2.21	0.00	3.01	0.00	0.14	0.00	0.00	0.00	-2.55
03 - AVGs	19 - 144.09	14.00	1285.43	10.00	925.71	12.00	1349.47	3.00	985.01	3.00	591.17	3.00	1167.56
% to SI1-Like 06	19.75 - 141.91	0.00	-0.96	0.00	1.04	0.00	-0.82	0.00	-1.08	0.00	0.00	0.00	-1.33
04 - AVGs	19 - 144.09	10.00	1076.89	10.00	869.50	11.00	1215.68	3.00	864.52	3.00	648.56	3.00	936.61
% to SI1-Like 06	19.75 - 141.91	0.00	-0.54	0.00	-8.44	0.00	1.51	0.00	-0.18	0.00	2.23	0.00	1.03
05 - AVGs	19 - 144.09	14.00	1497.62	10.00	828.94	15.00	1656.19	3.00	1136.25	3.00	588.88	4.00	1427.75
% to SI1-Like 06	19.75 - 141.91	0.00	6.00	0.00	0.00	0.00	1.04	0.00	1.60	0.00	0.00	0.00	1.95
06 - AVGs	19 - 144.09	13.00	1320.39	10.00	828.94	12.00	1465.65	3.00	1034.76	3.00	588.49	3.00	1287.77
% to SI1-Like 06	19.75 - 141.91	0.00	-0.68	0.00	0.00	0.00	-0.55	0.00	0.98	0.00	0.00	0.00	0.95
07 - AVGs	19 - 144.09	11.00	1190.23	10.00	828.94	12.00	1329.55	3.00	1137.94	3.00	588.29	3.00	1181.34
% to SI1-Like 06	19.75 - 141.91	0.00	-6.52	0.00	0.00	0.00	-0.39	0.00	0.00	0.00	0.00	0.00	-4.97
08 - AVGs	19 - 144.09	10.00	1032.67	10.00	828.94	11.00	1219.55	2.00	826.90	3.00	588.32	3.00	1021.14
% to SI1-Like 06	19.75 - 141.91	0.00	0.27	0.00	0.00	0.00	1.10	0.00	0.35	0.00	0.00	0.00	4.92
09 - AVGs	19 - 144.09	12.00	1324.81	10.00	828.94	-	-	3.00	989.87	-	-	-	-
% to SI1-Like 06	19.75 - 141.91	0.00	4.86	0.00	0.00	-	-	0.00	0.00	-	-	-	-
10 - AVGs	19 - 144.09	12.00	1206.08	-	-	-	-	3.00	1005.56	-	-	-	-
% to SI1-Like 06	19.75 - 141.91	0.00	2.24	-	-	-	-	0.00	-0.09	-	-	-	-
11 - AVGs	19 - 144.09	11.00	1254.03	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 06	19.75 - 141.91	0.00	3.82	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	19 - 144.09	10.00	1018.47	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 06	19.75 - 141.91	0.00	-5.29	-	-	-	-	-	-	-	-	-	-
SI1-Like 07 -AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
Time(secs.)	19 - 144.09	21.75	-	21.56	-	19.00	-	144.09	-	72.50	-	110.62	-
% to SI1-Like 06	19.75 - 141.91	0.00	0.27	0.00	-0.66	0.00	0.61	0.00	0.20	0.00	0.30	0.00	-0.62
% to SI1-Like 05	15.22 - 114.55	0.00	-2.93	0.00	-6.41	0.00	-3.36	0.00	-11.13	0.00	-8.08	0.00	-11.39
% to SI1-Like 04	15.78 - 137.64	0.00	-3.01	0.00	-5.92	0.00	-3.52	0.00	-8.55	0.00	-6.89	0.00	-10.80
% to RVNSa [5]	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
% to AD [14]	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
% to PR2 [78]	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
% to DACS 01	300 - 400	-8.28	-15.42	0.00	-12.58	-7.27	-15.07	-2.86	-22.34	0.00	-13.94	-8.99	-23.16
% to DACS 02	300	-2.52	3.45	0.00	-0.23	-0.97	1.20	-2.86	-1.61	-2.70	-2.26	-7.95	-4.46
% to DACS 03	300	1.17	4.53	0.00	2.67	2.34	2.53	-1.26	8.42	0.00	0.83	-2.41	4.66
% to DACS+HLS	300	2.51	4.03	0.00	2.37	5.05	2.18	3.03	6.91	0.00	0.65	0.12	3.68
% to DACS+HLS+2-Opt	300	2.22	5.21	0.00	3.03	5.26	3.13	2.20	9.12	0.00	0.98	0.12	6.49
SI1-Like 06 -AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 05 - AVGs	15.22 - 114.55	12.92	1326.84	10.00	912.42	12.75	1489.35	3.09	1177.68	3.00	649.61	3.38	1372.98
SI1-Like 04 - AVGs	15.78 - 137.64	12.92	1327.94	10.00	907.72	12.75	1491.79	3.09	1144.51	3.00	641.31	3.38	1363.81
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.3: Comparison between the average case performances of SI1-Like 08 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	19.12 - 151.09	20.00	1680.20	10.00	828.94	15.00	1701.50	4.00	1341.69	3.00	591.56	4.00	1469.28
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00
02 - AVGs	19.12 - 151.09	18.00	1569.54	10.00	916.66	14.00	1561.54	4.00	1160.10	3.00	591.56	4.00	1256.63
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	-0.97	0.00	-1.12	0.00	0.00	0.00	1.27
03 - AVGs	19.12 - 151.09	14.00	1281.89	10.00	913.85	12.00	1349.47	3.00	985.01	3.00	591.17	3.00	1188.09
% to SI1-Like 07	19 - 144.09	0.00	-0.28	0.00	-1.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.76
04 - AVGs	19.12 - 151.09	10.00	1076.89	10.00	906.13	11.00	1228.27	3.00	839.00	3.00	629.20	3.00	916.78
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	4.21	0.00	1.04	0.00	-2.95	0.00	-2.99	0.00	-2.12
05 - AVGs	19.12 - 151.09	14.00	1497.62	10.00	828.94	15.00	1656.19	3.00	1126.15	3.00	588.88	4.00	1418.95
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.89	0.00	0.00	0.00	-0.62
06 - AVGs	19.12 - 151.09	13.00	1320.39	10.00	828.94	12.00	1465.65	3.00	1038.90	3.00	588.49	3.00	1268.71
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	-1.48
07 - AVGs	19.12 - 151.09	11.00	1190.23	10.00	828.94	12.00	1329.55	3.00	1137.94	3.00	588.29	3.00	1179.34
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.17
08 - AVGs	19.12 - 151.09	10.00	1020.52	10.00	828.94	11.00	1208.35	2.00	783.02	3.00	588.32	3.00	942.71
% to SI1-Like 07	19 - 144.09	0.00	-1.18	0.00	0.00	0.00	-0.92	0.00	-5.31	0.00	0.00	0.00	-7.68
09 - AVGs	19.12 - 151.09	12.00	1324.81	10.00	828.94	-	-	3.00	1005.72	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	-	-	0.00	1.60	-	-	-	-
10 - AVGs	19.12 - 151.09	12.00	1202.70	-	-	-	-	3.00	1008.45	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	-0.28	-	-	-	-	0.00	0.29	-	-	-	-
11 - AVGs	19.12 - 151.09	11.00	1254.03	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	0.00	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	19.12 - 151.09	10.00	1056.52	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	3.74	-	-	-	-	-	-	-	-	-	-
SI1-Like 08 - AVGs	19.12 - 151.09	12.92	1289.61	10.00	856.70	12.75	1437.57	3.09	1040.31	3.00	594.68	3.38	1205.06
Time(secs.)	19.12 - 151.09	22.00	-	21.67	-	19.12	-	151.09	-	79.12	-	119.88	-
% to SI1-Like 07	19 - 144.09	0.00	0.12	0.00	0.32	0.00	-0.12	0.00	-0.60	0.00	-0.41	0.00	-0.94
% to SI1-Like 06	19.75 - 141.91	0.00	0.40	0.00	-0.34	0.00	0.49	0.00	-0.41	0.00	-0.11	0.00	-1.56
% to RVNSa [5]	2220	7.64	4.89	0.00	3.42	10.87	3.11	13.22	5.12	0.00	0.74	3.85	5.61
% to AD [14]	132 - 253	0.68	-6.99	0.00	-10.33	2.00	-7.01	0.03	-23.87	0.00	-17.10	-0.15	-24.59
% to PR2 [78]	180	-3.10	-6.68	0.00	-5.12	-3.77	-6.97	-5.48	-19.57	-4.15	-8.96	-13.02	-24.45
% to DACS 01	300 - 400	-8.28	-15.31	0.00	-12.30	-7.27	-15.17	-2.86	-22.81	0.00	-14.29	-8.99	-23.88
% to DACS 02	300	-2.52	3.57	0.00	0.09	-0.97	1.08	-2.86	-2.21	-2.70	-2.66	-7.95	-5.36
% to DACS 03	300	1.17	4.66	0.00	3.00	2.34	2.40	-1.26	7.77	0.00	0.43	-2.41	3.67
% to DACS+HLS	300	2.51	4.16	0.00	2.70	5.05	2.06	3.03	6.26	0.00	0.24	0.12	2.70
% to DACS+HLS+2-Opt	300	2.22	5.34	0.00	3.36	5.26	3.01	2.20	8.46	0.00	0.57	0.12	5.49
SI1-Like 07 -AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 -AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.4: Comparison between the average case performances of SI1-Like 09 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	19.62 - 154.18	20.00	1680.20	10.00	828.94	15.00	1701.50	4.00	1341.69	3.00	591.56	4.00	1469.28
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00
02 - AVGs	19.62 - 154.18	18.00	1569.54	10.00	916.66	14.00	1561.54	4.00	1160.10	3.00	591.56	4.00	1256.63
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	-0.97	0.00	-1.12	0.00	0.00	0.00	1.27
03 - AVGs	19.62 - 154.18	14.00	1281.89	10.00	913.85	12.00	1349.47	3.00	985.01	3.00	591.17	3.00	1188.09
% to SI1-Like 07	19 - 144.09	0.00	-0.28	0.00	-1.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.76
04 - AVGs	19.62 - 154.18	10.00	1076.89	10.00	866.97	11.00	1228.27	3.00	839.00	3.00	629.20	3.00	907.55
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	-0.29	0.00	1.04	0.00	-2.95	0.00	-2.99	0.00	-3.10
05 - AVGs	19.62 - 154.18	14.00	1497.62	10.00	828.94	15.00	1656.19	3.00	1126.15	3.00	588.88	4.00	1418.95
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.89	0.00	0.00	0.00	-0.62
06 - AVGs	19.62 - 154.18	13.00	1320.39	10.00	828.94	12.00	1465.65	3.00	989.71	3.00	588.49	3.00	1255.59
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-4.35	0.00	0.00	0.00	-2.50
07 - AVGs	19.62 - 154.18	11.00	1153.97	10.00	828.94	12.00	1329.55	3.00	1137.94	3.00	588.29	3.00	1183.11
% to SI1-Like 07	19 - 144.09	0.00	-3.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15
08 - AVGs	19.62 - 154.18	10.00	1020.52	10.00	828.94	11.00	1208.35	2.00	783.14	3.00	588.32	3.00	975.32
% to SI1-Like 07	19 - 144.09	0.00	-1.18	0.00	0.00	0.00	-0.92	0.00	-5.29	0.00	0.00	0.00	-4.49
09 - AVGs	19.62 - 154.18	12.00	1324.81	10.00	828.94	-	-	3.00	1001.50	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	0.00	0.00	0.00	-	-	0.00	1.17	-	-	-	-
10 - AVGs	19.62 - 154.18	12.00	1202.70	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	-0.28	-	-	-	-	0.00	-0.46	-	-	-	-
11 - AVGs	19.62 - 154.18	11.00	1254.03	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	0.00	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	19.62 - 154.18	10.00	1056.52	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 07	19 - 144.09	0.00	3.74	-	-	-	-	-	-	-	-	-	-
SI1-Like 09 -AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
Time(secs.)	19.62 - 154.18	22.50	-	21.67	-	19.62	-	154.18	-	78.50	-	118.50	-
% to SI1-Like 07	19 - 144.09	0.00	-0.11	0.00	-0.19	0.00	-0.12	0.00	-1.13	0.00	-0.41	0.00	-0.80
% to SI1-Like 06	19.75 - 141.91	0.00	0.16	0.00	-0.84	0.00	0.49	0.00	-0.94	0.00	-0.11	0.00	-1.41
% to SI1-Like 08	19.12 - 151.09	0.00	-0.23	0.00	-0.51	0.00	0.00	0.00	-0.53	0.00	0.00	0.00	0.15
% to RVNSa [5]	2220	7.64	4.65	0.00	2.89	10.87	3.11	13.22	4.56	0.00	0.74	3.85	5.76
% to AD [14]	132 - 253	0.68	-7.20	0.00	-10.79	2.00	-7.01	0.03	-24.27	0.00	-17.10	-0.15	-24.48
% to PR2 [78]	180	-3.10	-6.90	0.00	-5.60	-3.77	-6.97	-5.48	-19.99	-4.15	-8.96	-13.02	-24.34
% to DACS 01	300 - 400	-8.28	-15.51	0.00	-12.75	-7.27	-15.17	-2.86	-23.22	0.00	-14.29	-8.99	-23.77
% to DACS 02	300	-2.52	3.33	0.00	-0.42	-0.97	1.08	-2.86	-2.72	-2.70	-2.66	-7.95	-5.22
% to DACS 03	300	1.17	4.42	0.00	2.47	2.34	2.40	-1.26	7.19	0.00	0.43	-2.41	3.82
% to DACS+HLS	300	2.51	3.91	0.00	2.18	5.05	2.06	3.03	5.70	0.00	0.24	0.12	2.85
% to DACS+HLS+2-Opt	300	2.22	5.09	0.00	2.83	5.26	3.01	2.20	7.89	0.00	0.57	0.12	5.64
SI1-Like 07 -AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 -AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 08 - AVGs	19.12 - 151.09	12.92	1289.61	10.00	856.70	12.75	1437.57	3.09	1040.31	3.00	594.68	3.38	1205.06
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.5: Comparison between the average case performances of SI1-Like 10 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	6.62 - 49	20.00	1825.93	10.00	878.36	15.00	1880.45	4.00	1731.51	3.00	591.56	4.00	1926.97
% to SI1-Like 09	19.62 - 154.18	0.00	8.67	0.00	5.96	0.00	10.52	0.00	29.05	0.00	0.00	0.00	31.15
02 - AVGs	6.62 - 49	18.00	1864.34	10.00	978.30	14.00	1701.87	4.00	1483.13	3.00	713.83	4.00	1642.33
% to SI1-Like 09	19.62 - 154.18	0.00	18.78	0.00	6.72	0.00	8.99	0.00	27.85	0.00	20.67	0.00	30.69
03 - AVGs	6.62 - 49	14.00	1642.49	10.00	1063.85	12.00	1495.64	3.00	1413.53	3.00	753.97	3.00	1451.60
% to SI1-Like 09	19.62 - 154.18	0.00	28.13	0.00	16.41	0.00	10.83	0.00	43.50	0.00	27.54	0.00	22.18
04 - AVGs	6.62 - 49	11.00	1258.95	10.00	1134.71	11.00	1341.12	3.00	1027.94	3.00	920.99	3.00	1179.59
% to SI1-Like 09	19.62 - 154.18	10.00	16.91	0.00	30.88	0.00	9.19	0.00	22.52	0.00	46.37	0.00	29.98
05 - AVGs	6.62 - 49	14.00	1523.92	10.00	878.78	15.00	1927.15	3.00	1385.52	3.00	606.28	4.00	1882.80
% to SI1-Like 09	19.62 - 154.18	0.00	1.76	0.00	6.01	0.00	16.36	0.00	23.03	0.00	2.95	0.00	32.69
06 - AVGs	6.62 - 49	13.00	1443.40	10.00	898.40	13.00	1542.89	3.00	1223.62	3.00	663.19	4.00	1519.09
% to SI1-Like 09	19.62 - 154.18	0.00	9.32	0.00	8.38	8.33	5.27	0.00	23.63	0.00	12.69	33.33	20.99
07 - AVGs	6.62 - 49	12.00	1315.21	10.00	903.91	12.00	1496.77	3.00	1137.94	3.00	633.27	4.00	1482.67
% to SI1-Like 09	19.62 - 154.18	9.09	13.97	0.00	9.04	0.00	12.58	0.00	0.00	0.00	7.65	33.33	25.32
08 - AVGs	6.62 - 49	10.00	1155.63	10.00	853.25	11.00	1331.62	3.00	930.37	3.00	634.38	3.00	1156.87
% to SI1-Like 09	19.62 - 154.18	0.00	13.24	0.00	2.93	0.00	10.20	50.00	18.80	0.00	7.83	0.00	18.61
09 - AVGs	6.62 - 49	13.00	1375.65	10.00	884.16	-	-	3.00	1297.73	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	8.33	3.84	0.00	6.66	-	-	0.00	29.58	-	-	-	-
10 - AVGs	6.62 - 49	12.00	1296.45	-	-	-	-	3.00	1361.47	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	7.79	-	-	-	-	0.00	36.01	-	-	-	-
11 - AVGs	6.62 - 49	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	6.62 - 49	10.00	1189.46	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	12.58	-	-	-	-	-	-	-	-	-	-
SI1-Like 10 - AVGs	6.62 - 49	13.17	1431.03	10.00	941.52	12.88	1589.69	3.18	1273.65	3.00	689.68	3.63	1530.24
Time(secs.)	6.62 - 49	6.92	-	7.44	-	6.62	-	49.00	-	29.88	-	38.00	-
% to SI1-Like 09	19.62 - 154.18	1.94	11.23	0.00	10.46	0.98	10.58	2.94	23.08	0.00	15.97	7.41	26.80
% to SI1-Like 07	19 - 144.09	1.94	11.10	0.00	10.26	0.98	10.45	2.94	21.69	0.00	15.50	7.41	25.79
% to SI1-Like 06	19.75 - 141.91	1.94	11.41	0.00	9.53	0.98	11.12	2.94	21.93	0.00	15.85	7.41	25.01
% to SI1-Like 03	5.88 - 46	0.00	0.00	0.00	-0.34	-1.90	1.68	0.00	-0.40	0.00	0.00	0.00	-0.98
% to RVNSa [5]	2220	9.72	16.39	0.00	13.66	11.96	14.02	16.55	28.70	0.00	16.84	11.54	34.11
% to AD [14]	132 - 253	2.62	3.21	0.00	-1.45	3.00	2.83	2.97	-6.79	0.00	-3.85	7.25	-4.24
% to PR2 [78]	180	-1.23	3.56	0.00	4.28	-2.83	2.87	-2.70	-1.53	-4.15	5.59	-6.57	-4.07
% to DACS 01	300 - 400	-6.51	-6.03	0.00	-3.62	-6.36	-6.19	0.00	-5.50	0.00	-0.59	-2.25	-3.34
% to DACS 02	300	-0.63	14.93	0.00	10.00	0.00	11.78	0.00	19.73	-2.70	12.89	-1.14	20.18
% to DACS 03	300	3.13	16.14	0.00	13.20	3.34	13.24	1.65	31.94	0.00	16.47	4.82	31.64
% to DACS+HLS	300	4.50	15.58	0.00	12.87	6.08	12.86	6.06	30.09	0.00	16.25	7.54	30.41
% to DACS+HLS+2-Opt	300	4.20	16.89	0.00	13.59	6.30	13.91	5.21	32.79	0.00	16.64	7.54	33.95
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 03 - AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1361.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.6: Comparison between the average case performances of SI1-Like 11 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	6.62 - 49.36	20.00	1825.93	10.00	878.36	16.00	1803.80	4.00	1751.55	3.00	591.56	4.00	1926.97
% to SI1-Like 09	19.62 - 154.18	0.00	8.67	0.00	5.96	6.67	6.01	0.00	30.55	0.00	0.00	0.00	31.15
02 - AVGs	6.62 - 49.36	18.00	1864.34	10.00	978.30	14.00	1701.87	4.00	1488.07	3.00	713.83	4.00	1642.33
% to SI1-Like 09	19.62 - 154.18	0.00	18.78	0.00	6.72	0.00	8.99	0.00	28.27	0.00	20.67	0.00	30.69
03 - AVGs	6.62 - 49.36	14.00	1622.81	10.00	1092.72	12.00	1545.31	3.00	1413.53	3.00	753.97	3.00	1457.58
% to SI1-Like 09	19.62 - 154.18	0.00	26.60	0.00	19.57	0.00	14.51	0.00	43.50	0.00	27.54	0.00	22.68
04 - AVGs	6.62 - 49.36	11.00	1258.95	10.00	1134.71	11.00	1341.12	3.00	1027.94	3.00	920.99	3.00	1179.59
% to SI1-Like 09	19.62 - 154.18	10.00	16.91	0.00	30.88	0.00	9.19	0.00	22.52	0.00	46.37	0.00	29.98
05 - AVGs	6.62 - 49.36	14.00	1523.92	10.00	878.78	15.00	1794.91	3.00	1388.65	3.00	606.28	4.00	1882.80
% to SI1-Like 09	19.62 - 154.18	0.00	1.76	0.00	6.01	0.00	8.38	0.00	23.31	0.00	2.95	0.00	32.69
06 - AVGs	6.62 - 49.36	13.00	1443.40	10.00	898.40	13.00	1542.89	3.00	1223.62	3.00	663.19	4.00	1519.09
% to SI1-Like 09	19.62 - 154.18	0.00	9.32	0.00	8.38	8.33	5.27	0.00	23.63	0.00	12.69	33.33	20.99
07 - AVGs	6.62 - 49.36	12.00	1315.21	10.00	903.91	12.00	1496.77	3.00	1137.94	3.00	633.27	4.00	1482.67
% to SI1-Like 09	19.62 - 154.18	9.09	13.97	0.00	9.04	0.00	12.58	0.00	0.00	0.00	7.65	33.33	25.32
08 - AVGs	6.62 - 49.36	10.00	1155.63	10.00	853.25	11.00	1331.62	3.00	930.37	3.00	634.38	3.00	1156.87
% to SI1-Like 09	19.62 - 154.18	0.00	13.24	0.00	2.93	0.00	10.20	50.00	18.80	0.00	7.83	0.00	18.61
09 - AVGs	6.62 - 49.36	12.00	1578.02	10.00	884.16	-	-	3.00	1297.73	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	19.11	0.00	6.66	-	-	0.00	29.58	-	-	-	-
10 - AVGs	6.62 - 49.36	12.00	1296.45	-	-	-	-	3.00	1350.35	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	7.79	-	-	-	-	0.00	34.90	-	-	-	-
11 - AVGs	6.62 - 49.36	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	6.62 - 49.36	10.00	1189.46	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	12.58	-	-	-	-	-	-	-	-	-	-
SI1-Like 11 - AVGs	6.62 - 49.36	13.08	1446.25	10.00	944.73	13.00	1569.79	3.18	1275.20	3.00	689.68	3.63	1530.99
Time(secs.)	6.62 - 49.36	7.17	-	8.44	-	6.62	-	49.36	-	31.62	-	40.25	-
% to SI1-Like 09	19.62 - 154.18	1.29	12.41	0.00	10.84	1.96	9.20	2.94	23.23	0.00	15.97	7.41	26.86
% to SI1-Like 07	19 - 144.09	1.29	12.28	0.00	10.63	1.96	9.07	2.94	21.84	0.00	15.50	7.41	25.85
% to SI1-Like 06	19.75 - 141.91	1.29	12.59	0.00	9.91	1.96	9.73	2.94	22.08	0.00	15.85	7.41	25.07
% to SI1-Like 03	5.88 - 46	-0.63	1.06	0.00	0.00	-0.95	0.40	0.00	-0.28	0.00	0.00	0.00	-0.93
% to RVNSa [5]	2220	9.03	17.63	0.00	14.05	13.04	12.59	16.55	28.86	0.00	16.84	11.54	34.17
% to AD [14]	132 - 253	1.97	4.31	0.00	-1.12	4.00	1.54	2.97	-6.68	0.00	-3.85	7.25	-4.20
% to PR2 [78]	180	-1.85	4.66	0.00	4.63	-1.89	1.58	-2.70	-1.41	-4.15	5.59	-6.57	-4.02
% to DACS 01	300 - 400	-7.10	-5.03	0.00	-3.29	-5.45	-7.37	0.00	-5.39	0.00	-0.59	-2.25	-3.29
% to DACS 02	300	-1.26	16.15	0.00	10.38	0.97	10.38	0.00	19.88	-2.70	12.89	-1.14	20.23
% to DACS 03	300	2.48	17.37	0.00	13.58	4.35	11.82	1.65	32.10	0.00	16.47	4.82	31.71
% to DACS+HLS	300	3.84	16.81	0.00	13.25	7.11	11.45	6.06	30.25	0.00	16.25	7.54	30.48
% to DACS+HLS+2-Opt	300	3.54	18.14	0.00	13.98	7.33	12.48	5.21	32.95	0.00	16.64	7.54	34.02
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 03 - AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.7: Comparison between the average case performances of SI1-Like 12 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	8.12 - 51.45	20.00	1825.93	10.00	878.36	16.00	1803.80	4.00	1751.55	3.00	591.56	4.00	1926.97
% to SI1-Like 09	19.62 - 154.18	0.00	8.67	0.00	5.96	6.67	6.01	0.00	30.55	0.00	0.00	0.00	31.15
02 - AVGs	8.12 - 51.45	18.00	1864.34	10.00	978.30	14.00	1701.87	4.00	1468.07	3.00	713.83	4.00	1642.33
% to SI1-Like 09	19.62 - 154.18	0.00	18.78	0.00	6.72	0.00	8.99	0.00	28.27	0.00	20.67	0.00	30.69
03 - AVGs	8.12 - 51.45	14.00	1642.49	10.00	1092.72	12.00	1545.31	3.00	1413.53	3.00	753.97	3.00	1449.74
% to SI1-Like 09	19.62 - 154.18	0.00	28.13	0.00	19.57	0.00	14.51	0.00	43.50	0.00	27.54	0.00	22.02
04 - AVGs	8.12 - 51.45	11.00	1258.95	10.00	1134.71	11.00	1341.12	3.00	1027.94	3.00	920.99	3.00	1179.59
% to SI1-Like 09	19.62 - 154.18	10.00	16.91	0.00	30.88	0.00	9.19	0.00	22.52	0.00	46.37	0.00	29.98
05 - AVGs	8.12 - 51.45	14.00	1523.92	10.00	878.78	16.00	1744.58	3.00	1388.65	3.00	606.28	4.00	1882.80
% to SI1-Like 09	19.62 - 154.18	0.00	1.76	0.00	6.01	6.67	5.34	0.00	23.31	0.00	2.95	0.00	32.69
06 - AVGs	8.12 - 51.45	13.00	1443.40	10.00	898.40	13.00	1542.89	3.00	1223.62	3.00	663.19	4.00	1519.09
% to SI1-Like 09	19.62 - 154.18	0.00	9.32	0.00	8.38	8.33	5.27	0.00	23.63	0.00	12.69	33.33	20.99
07 - AVGs	8.12 - 51.45	12.00	1315.21	10.00	903.91	12.00	1496.77	3.00	1137.94	3.00	633.27	3.00	1744.14
% to SI1-Like 09	19.62 - 154.18	9.09	13.97	0.00	9.04	0.00	12.58	0.00	0.00	0.00	7.65	0.00	47.42
08 - AVGs	8.12 - 51.45	10.00	1155.63	10.00	853.25	11.00	1331.62	3.00	930.37	3.00	634.38	3.00	1156.87
% to SI1-Like 09	19.62 - 154.18	0.00	13.24	0.00	2.93	0.00	10.20	50.00	18.80	0.00	7.83	0.00	18.61
09 - AVGs	8.12 - 51.45	13.00	1375.65	10.00	884.16	-	-	3.00	1297.73	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	8.33	3.84	0.00	6.66	-	-	0.00	29.58	-	-	-	-
10 - AVGs	8.12 - 51.45	12.00	1296.45	-	-	-	-	3.00	1357.14	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	7.79	-	-	-	-	0.00	35.58	-	-	-	-
11 - AVGs	8.12 - 51.45	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	8.12 - 51.45	10.00	1189.46	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	12.58	-	-	-	-	-	-	-	-	-	-
SI1-Like 12 - AVGs	8.12 - 51.45	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1275.81	3.00	689.68	3.50	1562.69
Time(secs.)	8.12 - 51.45	8.17	-	8.67	-	8.12	-	51.45	-	32.25	-	41.38	-
% to SI1-Like 09	19.62 - 154.18	1.94	11.23	0.00	10.84	2.94	8.76	2.94	23.29	0.00	15.97	3.70	29.49
% to SI1-Like 07	19 - 144.09	1.94	11.10	0.00	10.63	2.94	8.63	2.94	21.90	0.00	15.50	3.70	28.45
% to SI1-Like 06	19.75 - 141.91	1.94	11.41	0.00	9.91	2.94	9.29	2.94	22.14	0.00	15.85	3.70	27.66
% to SI1-Like 03	5.88 - 46	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.23	0.00	0.00	-3.45	1.12
% to RVNSa [5]	2220	9.72	16.39	0.00	14.05	14.13	12.14	16.55	28.92	0.00	16.84	7.69	36.95
% to AD [14]	132 - 253	2.62	3.21	0.00	-1.12	5.00	1.14	2.97	-6.63	0.00	-3.85	3.55	-2.21
% to PR2 [78]	180	-1.23	3.56	0.00	4.63	-0.94	1.18	-2.70	-1.36	-4.15	5.59	-9.79	-2.03
% to DACS 01	300 - 400	-6.51	-6.03	0.00	-3.29	-4.55	-7.74	0.00	-5.34	0.00	-0.59	-5.62	-1.29
% to DACS 02	300	-0.63	14.93	0.00	10.38	1.94	9.93	0.00	19.93	-2.70	12.89	-4.55	22.72
% to DACS 03	300	3.13	16.14	0.00	13.58	5.35	11.37	1.65	32.16	0.00	16.47	1.20	34.43
% to DACS+HLS	300	4.50	15.58	0.00	13.25	8.14	11.00	6.06	30.32	0.00	16.25	3.83	33.18
% to DACS+HLS+2-Opt	300	4.20	16.89	0.00	13.98	8.36	12.03	5.21	33.02	0.00	16.64	3.83	36.79
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 03 - AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.8: Comparison between the average case performances of SI1-Like 13 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	8.33 - 51.64	19.00	1984.92	10.00	878.36	16.00	1803.80	4.00	1751.55	3.00	591.56	4.00	1926.97
% to SI1-Like 09	19.62 - 154.18	-5.00	18.14	0.00	5.96	6.67	6.01	0.00	30.55	0.00	0.00	0.00	31.15
02 - AVGs	8.33 - 51.64	18.00	1864.34	10.00	978.30	14.00	1701.87	4.00	1488.07	3.00	713.83	4.00	1642.33
% to SI1-Like 09	19.62 - 154.18	0.00	18.78	0.00	6.72	0.00	8.99	0.00	28.27	0.00	20.67	0.00	30.69
03 - AVGs	8.33 - 51.64	14.00	1642.49	10.00	1092.72	12.00	1545.31	3.00	1413.53	3.00	753.97	3.00	1449.74
% to SI1-Like 09	19.62 - 154.18	0.00	28.13	0.00	19.57	0.00	14.51	0.00	43.50	0.00	27.54	0.00	22.02
04 - AVGs	8.33 - 51.64	11.00	1258.95	10.00	1134.71	11.00	1341.12	3.00	1027.94	3.00	920.99	3.00	1179.59
% to SI1-Like 09	19.62 - 154.18	10.00	16.91	0.00	30.88	0.00	9.19	0.00	22.52	0.00	46.37	0.00	29.98
05 - AVGs	8.33 - 51.64	14.00	1523.92	10.00	878.78	16.00	1744.58	3.00	1388.65	3.00	606.28	4.00	1882.80
% to SI1-Like 09	19.62 - 154.18	0.00	1.76	0.00	6.01	6.67	5.34	0.00	23.31	0.00	2.95	0.00	32.69
06 - AVGs	8.33 - 51.64	13.00	1443.40	10.00	898.40	13.00	1542.89	3.00	1223.62	3.00	663.19	4.00	1519.09
% to SI1-Like 09	19.62 - 154.18	0.00	9.32	0.00	8.38	8.33	5.27	0.00	23.63	0.00	12.69	33.33	20.99
07 - AVGs	8.33 - 51.64	12.00	1315.21	10.00	903.91	12.00	1496.77	3.00	1137.94	3.00	633.27	3.00	1792.86
% to SI1-Like 09	19.62 - 154.18	9.09	13.97	0.00	9.04	0.00	12.58	0.00	0.00	0.00	7.65	0.00	51.54
08 - AVGs	8.33 - 51.64	10.00	1155.63	10.00	853.25	11.00	1331.62	3.00	930.37	3.00	634.38	3.00	1156.87
% to SI1-Like 09	19.62 - 154.18	0.00	13.24	0.00	2.93	0.00	10.20	50.00	18.80	0.00	7.83	0.00	18.61
09 - AVGs	8.33 - 51.64	12.00	1542.77	10.00	884.16	-	-	3.00	1297.73	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	16.45	0.00	6.66	-	-	0.00	29.58	-	-	-	-
10 - AVGs	8.33 - 51.64	12.00	1296.45	-	-	-	-	3.00	1389.47	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	7.79	-	-	-	-	0.00	38.81	-	-	-	-
11 - AVGs	8.33 - 51.64	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	8.33 - 51.64	10.00	1189.46	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	12.58	-	-	-	-	-	-	-	-	-	-
SI1-Like 13 - AVGs	8.33 - 51.64	13.00	1458.21	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.50	1568.78
Time(secs.)	8.33 - 51.64	8.33	-	9.78	-	8.62	-	51.64	-	34.00	-	42.12	-
% to SI1-Like 09	19.62 - 154.18	0.65	13.34	0.00	10.84	2.94	8.76	2.94	23.58	0.00	15.97	3.70	29.99
% to SI1-Like 07	19 - 144.09	0.65	13.21	0.00	10.63	2.94	8.63	2.94	22.18	0.00	15.50	3.70	28.95
% to SI1-Like 06	19.75 - 141.91	0.65	13.52	0.00	9.91	2.94	9.29	2.94	22.42	0.00	15.85	3.70	28.16
% to SI1-Like 03	5.88 - 46	-1.27	1.90	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-3.45	1.52
% to RVNSa [5]	2220	8.33	18.60	0.00	14.05	14.13	12.14	16.55	29.22	0.00	16.84	7.69	37.48
% to AD [14]	132 - 253	1.33	5.17	0.00	-1.12	5.00	1.14	2.97	-6.42	0.00	-3.85	3.55	-1.83
% to PR2 [78]	180	-2.48	5.52	0.00	4.63	-0.94	1.18	-2.70	-1.13	-4.15	5.59	-9.79	-1.65
% to DACS 01	300 - 400	-7.69	-4.24	0.00	-3.29	-4.55	-7.74	0.00	-5.12	0.00	-0.59	-5.62	-0.91
% to DACS 02	300	-1.89	17.11	0.00	10.38	1.94	9.93	0.00	20.21	-2.70	12.89	-4.55	23.20
% to DACS 03	300	1.83	18.34	0.00	13.58	5.35	11.37	1.65	32.47	0.00	16.47	1.20	34.96
% to DACS+HLS	300	3.17	17.77	0.00	13.25	8.14	11.00	6.06	30.62	0.00	16.25	3.83	33.70
% to DACS+HLS+2-Opt	300	2.88	19.11	0.00	13.98	8.36	12.03	5.21	33.32	0.00	16.64	3.83	37.33
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 03 - AVGs	5.88 - 46	13.17	1431.03	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.63	1545.33
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.9: Comparison between the average case performances of SI1-Like 14 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	10.88 - 80.82	19.00	1721.36	10.00	828.94	16.00	1732.73	4.00	1311.58	3.00	591.56	4.00	1559.21
% to SI1-Like 09	19.62 - 154.18	-5.00	2.45	0.00	0.00	6.67	1.84	0.00	-2.24	0.00	0.00	0.00	6.12
02 - AVGs	10.88 - 80.82	18.00	1573.05	10.00	835.57	14.00	1587.33	4.00	1160.03	3.00	591.56	4.00	1308.40
% to SI1-Like 09	19.62 - 154.18	0.00	0.22	0.00	-8.85	0.00	1.65	0.00	-0.01	0.00	0.00	0.00	4.12
03 - AVGs	10.88 - 80.82	14.00	1327.98	10.00	954.11	12.00	1342.61	3.00	1023.43	3.00	617.80	3.00	1182.21
% to SI1-Like 09	19.62 - 154.18	0.00	3.60	0.00	4.41	0.00	-0.51	0.00	3.90	0.00	4.50	0.00	-0.49
04 - AVGs	10.88 - 80.82	11.00	1076.66	10.00	956.42	11.00	1240.43	3.00	848.65	3.00	920.99	3.00	885.69
% to SI1-Like 09	19.62 - 154.18	10.00	-0.02	0.00	10.32	0.00	0.99	0.00	1.15	0.00	46.37	0.00	-2.41
05 - AVGs	10.88 - 80.82	14.00	1523.92	10.00	828.94	16.00	1654.83	3.00	1093.72	3.00	588.88	4.00	1399.41
% to SI1-Like 09	19.62 - 154.18	0.00	1.76	0.00	0.00	6.67	-0.08	0.00	-2.88	0.00	0.00	0.00	-1.38
06 - AVGs	10.88 - 80.82	13.00	1358.61	10.00	898.40	13.00	1501.88	3.00	965.90	3.00	588.49	4.00	1255.94
% to SI1-Like 09	19.62 - 154.18	0.00	2.89	0.00	8.38	8.33	2.47	0.00	-2.41	0.00	0.00	33.33	0.03
07 - AVGs	10.88 - 80.82	12.00	1315.21	10.00	903.91	12.00	1361.62	3.00	1137.94	3.00	588.29	3.00	1221.31
% to SI1-Like 09	19.62 - 154.18	9.09	13.97	0.00	9.04	0.00	2.41	0.00	0.00	0.00	0.00	0.00	3.23
08 - AVGs	10.88 - 80.82	10.00	1038.40	10.00	834.01	11.00	1331.62	3.00	930.37	3.00	588.32	3.00	1005.02
% to SI1-Like 09	19.62 - 154.18	0.00	1.75	0.00	0.61	0.00	10.20	50.00	18.80	0.00	0.00	0.00	3.05
09 - AVGs	10.88 - 80.82	12.00	1379.64	10.00	828.94	-	-	3.00	983.24	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	4.14	0.00	0.00	-	-	0.00	-1.82	-	-	-	-
10 - AVGs	10.88 - 80.82	12.00	1201.76	-	-	-	-	3.00	1024.40	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-0.08	-	-	-	-	0.00	2.34	-	-	-	-
11 - AVGs	10.88 - 80.82	11.00	1280.93	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	2.15	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	10.88 - 80.82	10.00	1108.72	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	4.94	-	-	-	-	-	-	-	-	-	-
SI1-Like 14 -AVGs	10.88 - 80.82	13.00	1325.52	10.00	874.36	13.13	1469.13	3.18	1045.15	3.00	634.49	3.50	1227.15
Time(secs.)	10.88 - 80.82	12.25	-	18.56	-	10.88	-	80.82	-	49.88	-	62.12	-
% to SI1-Like 09	19.62 - 154.18	0.65	3.03	0.00	2.58	2.94	2.20	2.94	1.00	0.00	6.69	3.70	1.68
% to SI1-Like 07	19 - 144.09	0.65	2.91	0.00	2.39	2.94	2.07	2.94	-0.14	0.00	6.26	3.70	0.87
% to SI1-Like 06	19.75 - 141.91	0.65	3.19	0.00	1.72	2.94	2.70	2.94	0.06	0.00	6.58	3.70	0.25
% to SI1-Like 13	8.33 - 51.64	0.00	-9.10	0.00	-7.45	0.00	-6.04	0.00	-18.27	0.00	-8.00	0.00	-21.78
% to RVNSa [5]	2220	8.33	7.81	0.00	5.55	14.13	5.37	16.55	5.61	0.00	7.49	7.69	7.54
% to AD [14]	132 - 253	1.33	-4.40	0.00	-8.48	5.00	-4.97	2.97	-23.51	0.00	-11.55	3.55	-23.21
% to PR2 [78]	180	-2.48	-4.08	0.00	-3.16	-0.94	-4.93	-2.70	-19.19	-4.15	-2.86	-9.79	-23.07
% to DACS 01	300 - 400	-7.69	-12.96	0.00	-10.49	-4.55	-13.31	0.00	-22.45	0.00	-8.55	-5.62	-22.48
% to DACS 02	300	-1.89	6.46	0.00	2.15	1.94	3.30	0.00	-1.75	-2.70	3.85	-4.55	-3.63
% to DACS 03	300	1.83	7.57	0.00	5.12	5.35	4.65	1.65	8.27	0.00	7.15	1.20	5.57
% to DACS+HLS	300	3.17	7.06	0.00	4.81	8.14	4.30	6.06	6.76	0.00	6.95	3.83	4.58
% to DACS+HLS+2-Opt	300	2.88	8.27	0.00	5.49	8.36	5.27	5.21	8.97	0.00	7.31	3.83	7.42
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 13 - AVGs	8.33 - 51.64	13.00	1458.21	10.00	944.73	13.13	1563.50	3.18	1278.75	3.00	689.68	3.50	1568.78
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.10: Comparison between the average case performances of SI1-Like 15 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	22.12 - 178.82	19.00	1721.36	10.00	828.94	15.00	1718.22	4.00	1311.58	3.00	591.56	4.00	1499.18
% to SI1-Like 09	19.62 - 154.18	-5.00	2.45	0.00	0.00	0.00	0.98	0.00	-2.24	0.00	0.00	0.00	2.04
02 - AVGs	22.12 - 178.82	18.00	1543.66	10.00	835.57	14.00	1562.28	4.00	1160.03	3.00	591.56	4.00	1256.63
% to SI1-Like 09	19.62 - 154.18	0.00	-1.65	0.00	-8.85	0.00	0.05	0.00	-0.01	0.00	0.00	0.00	0.00
03 - AVGs	22.12 - 178.82	14.00	1290.92	10.00	954.11	12.00	1342.61	3.00	975.53	3.00	591.17	3.00	1164.61
% to SI1-Like 09	19.62 - 154.18	0.00	0.70	0.00	4.41	0.00	-0.51	0.00	-0.96	0.00	0.00	0.00	-1.98
04 - AVGs	22.12 - 178.82	11.00	1059.27	10.00	956.42	11.00	1219.48	3.00	848.65	3.00	629.20	3.00	885.69
% to SI1-Like 09	19.62 - 154.18	10.00	-1.64	0.00	10.32	0.00	-0.72	0.00	1.15	0.00	0.00	0.00	-2.41
05 - AVGs	22.12 - 178.82	14.00	1435.51	10.00	828.94	15.00	1617.58	3.00	1080.27	3.00	588.88	4.00	1399.41
% to SI1-Like 09	19.62 - 154.18	0.00	-4.15	0.00	0.00	0.00	-2.33	0.00	-4.07	0.00	0.00	0.00	-1.38
06 - AVGs	22.12 - 178.82	12.00	1335.19	10.00	828.94	12.00	1465.68	3.00	965.90	3.00	588.49	3.00	1304.75
% to SI1-Like 09	19.62 - 154.18	-7.69	1.12	0.00	0.00	0.00	0.00	0.00	-2.41	0.00	0.00	0.00	3.92
07 - AVGs	22.12 - 178.82	11.00	1175.08	10.00	828.94	12.00	1307.39	3.00	1137.94	3.00	588.29	3.00	1178.67
% to SI1-Like 09	19.62 - 154.18	0.00	1.83	0.00	0.00	0.00	-1.67	0.00	0.00	0.00	0.00	0.00	-0.38
08 - AVGs	22.12 - 178.82	10.00	1017.12	10.00	828.94	11.00	1197.86	2.00	783.14	3.00	588.32	3.00	976.44
% to SI1-Like 09	19.62 - 154.18	0.00	-0.33	0.00	0.00	0.00	-0.87	0.00	0.00	0.00	0.00	0.00	0.11
09 - AVGs	22.12 - 178.82	12.00	1282.95	10.00	828.94	-	-	3.00	983.24	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-3.16	0.00	0.00	-	-	0.00	-1.82	-	-	-	-
10 - AVGs	22.12 - 178.82	11.00	1276.06	-	-	-	-	3.00	999.25	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	-8.33	6.10	-	-	-	-	0.00	-0.17	-	-	-	-
11 - AVGs	22.12 - 178.82	11.00	1215.07	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-3.11	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	22.12 - 178.82	10.00	1040.90	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-1.48	-	-	-	-	-	-	-	-	-	-
SI1-Like 15 -AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
Time(secs.)	22.12 - 178.82	26.92	-	29.22	-	22.12	-	178.82	-	89.62	-	138.62	-
% to SI1-Like 09	19.62 - 154.18	-1.29	-0.30	0.00	0.63	0.00	-0.60	0.00	-1.05	0.00	0.00	0.00	0.11
% to SI1-Like 07	19 - 144.09	-1.29	-0.41	0.00	0.45	0.00	-0.72	0.00	-2.17	0.00	-0.41	0.00	-0.69
% to SI1-Like 06	19.75 - 141.91	-1.29	-0.14	0.00	-0.21	0.00	-0.12	0.00	-1.98	0.00	-0.11	0.00	-1.30
% to SI1-Like 14	10.88 - 80.82	-1.92	-3.23	0.00	-1.90	-2.86	-2.74	-2.86	-2.03	0.00	-6.27	-3.57	-1.55
% to RVNSa [5]	2220	6.25	4.33	0.00	3.55	10.87	2.48	13.22	3.46	0.00	0.74	3.85	5.88
% to AD [14]	132 - 253	-0.62	-7.48	0.00	-10.22	2.00	-7.57	0.03	-25.07	0.00	-17.10	-0.15	-24.40
% to PR2 [78]	180	-4.35	-7.17	0.00	-5.00	-3.77	-7.53	-5.48	-20.84	-4.15	-8.96	-13.02	-24.26
% to DACS 01	300 - 400	-9.47	-15.76	0.00	-12.19	-7.27	-15.68	-2.86	-24.03	0.00	-14.29	-8.99	-23.68
% to DACS 02	300	-3.77	3.02	0.00	0.21	-0.97	0.47	-2.86	-3.75	-2.70	-2.66	-7.95	-5.12
% to DACS 03	300	-0.13	4.10	0.00	3.12	2.34	1.78	-1.26	6.07	0.00	0.43	-2.41	3.94
% to DACS+HLS	300	1.19	3.60	0.00	2.82	5.05	1.44	3.03	4.58	0.00	0.24	0.12	2.97
% to DACS+HLS+2-Opt	300	0.90	4.78	0.00	3.48	5.26	2.39	2.20	6.75	0.00	0.57	0.12	5.76
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 14 - AVGs	10.88 - 80.82	13.00	1325.52	10.00	874.36	13.13	1469.13	3.18	1045.15	3.00	634.49	3.50	1227.15
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.11: Comparison between the average case performances of SI1-Like 16 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	27.33 - 191.09	20.00	1680.20	10.00	828.94	15.00	1661.46	4.00	1341.69	3.00	591.56	4.00	1469.28
% to SI1-Like 09	19.62 - 154.18	0.00	0.00	0.00	0.00	0.00	-2.35	0.00	0.00	0.00	0.00	0.00	0.00
02 - AVGs	27.33 - 191.09	18.00	1545.80	10.00	916.66	14.00	1561.54	3.00	1277.30	3.00	591.56	4.00	1256.63
% to SI1-Like 09	19.62 - 154.18	0.00	-1.51	0.00	0.00	0.00	0.00	-25.00	10.10	0.00	0.00	0.00	0.00
03 - AVGs	27.33 - 191.09	14.00	1281.89	10.00	913.85	12.00	1349.47	3.00	985.01	3.00	591.17	3.00	1188.09
% to SI1-Like 09	19.62 - 154.18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
04 - AVGs	27.33 - 191.09	10.00	1076.89	10.00	866.97	11.00	1218.61	2.00	929.12	3.00	629.20	3.00	907.55
% to SI1-Like 09	19.62 - 154.18	0.00	0.00	0.00	0.00	0.00	-0.79	-33.33	10.74	0.00	0.00	0.00	0.00
05 - AVGs	27.33 - 191.09	14.00	1442.08	10.00	828.94	15.00	1656.19	3.00	1077.21	3.00	588.88	4.00	1418.95
% to SI1-Like 09	19.62 - 154.18	0.00	-3.71	0.00	0.00	0.00	0.00	0.00	-4.35	0.00	0.00	0.00	0.00
06 - AVGs	27.33 - 191.09	13.00	1299.92	10.00	828.94	12.00	1465.65	3.00	989.71	3.00	588.49	3.00	1255.59
% to SI1-Like 09	19.62 - 154.18	0.00	-1.55	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
07 - AVGs	27.33 - 191.09	11.00	1153.97	10.00	828.94	12.00	1309.79	3.00	1137.94	3.00	588.29	3.00	1183.11
% to SI1-Like 09	19.62 - 154.18	0.00	0.00	0.00	0.00	0.00	-1.49	0.00	0.00	0.00	0.00	0.00	0.00
08 - AVGs	27.33 - 191.09	10.00	1016.23	10.00	828.94	11.00	1155.30	2.00	783.14	3.00	588.32	3.00	975.32
% to SI1-Like 09	19.62 - 154.18	0.00	-0.42	0.00	0.00	0.00	-4.39	0.00	0.00	0.00	0.00	0.00	0.00
09 - AVGs	27.33 - 191.09	12.00	1234.81	10.00	828.94	-	-	3.00	1001.50	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-6.79	0.00	0.00	-	-	0.00	0.00	-	-	-	-
10 - AVGs	27.33 - 191.09	11.00	1164.80	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	-8.33	-3.15	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	27.33 - 191.09	11.00	1200.78	-	-	-	-	3.00	1017.42	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-4.25	-	-	-	-	0.00	0.00	-	-	-	-
12 - AVGs	27.33 - 191.09	10.00	1038.84	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 09	19.62 - 154.18	0.00	-1.67	-	-	-	-	-	-	-	-	-	-
SI1-Like 16 -AVGs	27.33 - 191.09	12.83	1261.35	10.00	852.35	12.75	1422.25	2.91	1049.18	3.00	594.68	3.38	1206.82
Time(secs.)	27.33 - 191.09	30.83	-	27.33	-	28.12	-	191.09	-	88.38	-	157.38	-
% to SI1-Like 09	19.62 - 154.18	-0.65	-1.96	0.00	0.00	0.00	-1.07	-5.88	1.39	0.00	0.00	0.00	0.00
% to SI1-Like 07	19 - 144.09	-0.65	-2.07	0.00	-0.19	0.00	-1.18	-5.88	0.24	0.00	-0.41	0.00	-0.80
% to SI1-Like 06	19.75 - 141.91	-0.65	-1.80	0.00	-0.84	0.00	-0.58	-5.88	0.44	0.00	-0.11	0.00	-1.41
% to SI1-Like 15	22.12 - 178.82	0.65	-1.67	0.00	-0.63	0.00	-0.46	-5.88	2.47	0.00	0.00	0.00	-0.11
% to RVNSa [5]	2220	6.94	2.59	0.00	2.89	10.87	2.01	6.56	6.02	0.00	0.74	3.85	5.76
% to AD [14]	132 - 253	0.03	-9.02	0.00	-10.79	2.00	-8.00	-5.85	-23.22	0.00	-17.10	-0.15	-24.48
% to PR2 [78]	180	-3.73	-8.72	0.00	-5.60	-3.77	-7.96	-11.04	-18.88	-4.15	-8.96	-13.02	-24.34
% to DACS 01	300 - 400	-8.88	-17.17	0.00	-12.75	-7.27	-16.07	-8.57	-22.16	0.00	-14.29	-6.99	-23.77
% to DACS 02	300	-3.14	1.30	0.00	-0.42	-0.97	0.00	-8.57	-1.37	-2.70	-2.66	-7.95	-5.22
% to DACS 03	300	0.52	2.37	0.00	2.47	2.34	1.31	-7.07	8.69	0.00	0.43	-2.41	3.82
% to DACS+HLS	300	1.85	1.87	0.00	2.18	5.05	0.97	-3.03	7.17	0.00	0.24	0.12	2.85
% to DACS+HLS+2-Opt	300	1.56	3.03	0.00	2.83	5.26	1.91	-3.81	9.39	0.00	0.57	0.12	5.64
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
SI1-Like 15 - AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.12: Comparison between the average case performances of SI1-Like 17 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	24.75 - 190.55	20.00	1690.39	10.00	828.94	15.00	1765.59	4.00	1322.47	3.00	591.56	4.00	1465.87
% to SI1-Like 16	27.33 - 191.09	0.00	0.61	0.00	0.00	0.00	6.27	0.00	-1.43	0.00	0.00	0.00	-0.23
02 - AVGs	24.75 - 190.55	18.00	1864.34	10.00	828.94	14.00	1531.25	3.00	1238.08	3.00	591.56	4.00	1256.63
% to SI1-Like 16	27.33 - 191.09	0.00	20.61	0.00	-9.57	0.00	-1.94	0.00	-3.07	0.00	0.00	0.00	0.00
03 - AVGs	24.75 - 190.55	14.00	1291.56	10.00	913.85	12.00	1338.72	3.00	985.01	3.00	603.70	3.00	1201.49
% to SI1-Like 16	27.33 - 191.09	0.00	0.75	0.00	0.00	0.00	-0.80	0.00	0.00	0.00	2.12	0.00	1.13
04 - AVGs	24.75 - 190.55	10.00	1076.89	10.00	925.44	11.00	1198.00	2.00	929.12	3.00	625.55	3.00	914.60
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	6.74	0.00	-1.69	0.00	0.00	0.00	-0.58	0.00	0.78
05 - AVGs	24.75 - 190.55	14.00	1418.55	10.00	828.94	15.00	1656.19	3.00	1077.21	3.00	588.88	4.00	1393.88
% to SI1-Like 16	27.33 - 191.09	0.00	-1.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-1.77
06 - AVGs	24.75 - 190.55	13.00	1299.92	10.00	828.94	13.00	1444.10	3.00	993.63	3.00	588.49	3.00	1280.50
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	8.33	-1.47	0.00	0.40	0.00	0.00	0.00	1.98
07 - AVGs	24.75 - 190.55	11.00	1196.14	10.00	828.94	12.00	1301.02	3.00	904.44	3.00	588.29	3.00	1274.60
% to SI1-Like 16	27.33 - 191.09	0.00	3.65	0.00	0.00	0.00	-0.67	0.00	-20.52	0.00	0.00	0.00	7.73
08 - AVGs	24.75 - 190.55	10.00	1017.09	10.00	828.94	11.00	1155.30	2.00	770.62	3.00	588.32	3.00	960.57
% to SI1-Like 16	27.33 - 191.09	0.00	0.08	0.00	0.00	0.00	0.00	0.00	-1.60	0.00	0.00	0.00	-1.51
09 - AVGs	24.75 - 190.55	12.00	1256.12	10.00	834.01	-	-	3.00	982.30	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	1.73	0.00	0.61	-	-	0.00	-1.92	-	-	-	-
10 - AVGs	24.75 - 190.55	11.00	1175.93	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	0.96	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	24.75 - 190.55	11.00	1262.73	-	-	-	-	3.00	885.40	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	5.16	-	-	-	-	0.00	-12.98	-	-	-	-
12 - AVGs	24.75 - 190.55	10.00	1044.70	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	0.56	-	-	-	-	-	-	-	-	-	-
SI1-Like 17 -AVGs	24.75 - 190.55	12.83	1299.53	10.00	849.66	12.88	1423.77	2.91	1008.11	3.00	595.79	3.38	1218.52
Time(secs.)	24.75 - 190.55	30.33	-	31.78	-	24.75	-	190.55	-	110.12	-	155.75	-
% to SI1-Like 16	27.33 - 191.09	0.00	3.03	0.00	-0.32	0.98	0.11	0.00	-3.91	0.00	0.19	0.00	0.97
% to SI1-Like 15	22.12 - 178.82	0.65	1.31	0.00	-0.94	0.98	-0.36	-5.88	-1.54	0.00	0.19	0.00	0.86
% to SI1-Like 09	19.62 - 154.18	-0.65	1.01	0.00	-0.32	0.98	-0.96	-5.88	-2.58	0.00	0.19	0.00	0.97
% to SI1-Like 07	19 - 144.09	-0.65	0.89	0.00	-0.50	0.98	-1.08	-5.88	-3.68	0.00	-0.22	0.00	0.16
% to SI1-Like 06	19.75 - 141.91	-0.65	1.17	0.00	-1.15	0.98	-0.47	-5.88	-3.49	0.00	0.08	0.00	-0.46
% to RVNSa [5]	2220	6.94	5.70	0.00	2.57	11.96	2.12	6.56	1.87	0.00	0.93	3.85	6.79
% to AD [14]	132 - 253	0.03	-6.27	0.00	-11.07	3.00	-7.90	-5.85	-26.23	0.00	-16.94	-0.15	-23.75
% to PR2 [78]	180	-3.73	-5.96	0.00	-5.90	-2.83	-7.86	-11.04	-22.06	-4.15	-8.79	-13.02	-23.61
% to DACS 01	300 - 400	-8.88	-14.66	0.00	-13.02	-6.36	-15.98	-8.57	-25.20	0.00	-14.13	-8.99	-23.03
% to DACS 02	300	-3.14	4.37	0.00	-0.73	0.00	0.11	-8.57	-5.23	-2.70	-2.48	-7.95	-4.30
% to DACS 03	300	0.52	5.47	0.00	2.15	3.34	1.42	-7.07	4.43	0.00	0.61	-2.41	4.83
% to DACS+HLS	300	1.85	4.96	0.00	1.85	6.08	1.08	-3.03	2.97	0.00	0.43	0.12	3.85
% to DACS+HLS+2-Opt	300	1.56	6.15	0.00	2.51	6.30	2.02	-3.81	5.11	0.00	0.76	0.12	6.67
SI1-Like 16 - AVGs	27.33 - 191.09	12.83	1261.35	10.00	852.35	12.75	1422.25	2.91	1049.18	3.00	594.68	3.38	1206.82
SI1-Like 15 - AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.13: Comparison between the average case performances of SI1-Like 18 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	26.88 - 233.55	20.00	1690.39	10.00	828.94	15.00	1707.91	4.00	1329.43	3.00	591.56	4.00	1514.47
% to SI1-Like 16	27.33 - 191.09	0.00	0.61	0.00	0.00	0.00	2.80	0.00	-0.91	0.00	0.00	0.00	3.08
02 - AVGs	26.88 - 233.55	18.00	1864.34	10.00	865.78	14.00	1526.49	4.00	1151.86	3.00	591.56	4.00	1256.63
% to SI1-Like 16	27.33 - 191.09	0.00	20.61	0.00	-5.55	0.00	-2.24	33.33	-9.82	0.00	0.00	0.00	0.00
03 - AVGs	26.88 - 233.55	14.00	1257.59	10.00	913.85	11.00	1403.20	3.00	985.01	3.00	603.70	3.00	1203.79
% to SI1-Like 16	27.33 - 191.09	0.00	-1.90	0.00	0.00	-8.33	3.98	0.00	0.00	0.00	2.12	0.00	1.32
04 - AVGs	26.88 - 233.55	10.00	1070.88	10.00	985.62	10.00	1262.64	2.00	947.46	3.00	648.56	3.00	877.20
% to SI1-Like 16	27.33 - 191.09	0.00	-0.56	0.00	13.69	-9.09	3.61	0.00	1.97	0.00	3.08	0.00	-3.34
05 - AVGs	26.88 - 233.55	14.00	1442.08	10.00	828.94	15.00	1656.19	3.00	1068.12	3.00	588.88	4.00	1409.91
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.84	0.00	0.00	0.00	-0.64
06 - AVGs	26.88 - 233.55	13.00	1299.92	10.00	828.94	12.00	1469.00	3.00	982.76	3.00	588.49	3.00	1258.80
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	0.00	0.23	0.00	-0.70	0.00	0.00	0.00	0.26
07 - AVGs	26.88 - 233.55	11.00	1138.77	10.00	828.94	12.00	1312.78	3.00	869.30	3.00	588.29	3.00	1165.66
% to SI1-Like 16	27.33 - 191.09	0.00	-1.32	0.00	0.00	0.00	0.23	0.00	-23.61	0.00	0.00	0.00	-1.47
08 - AVGs	26.88 - 233.55	10.00	1008.94	10.00	828.94	11.00	1155.30	2.00	758.75	3.00	588.32	3.00	957.75
% to SI1-Like 16	27.33 - 191.09	0.00	-0.72	0.00	0.00	0.00	0.00	0.00	-3.11	0.00	0.00	0.00	-1.80
09 - AVGs	26.88 - 233.55	12.00	1280.97	10.00	849.98	-	-	3.00	971.29	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	3.74	0.00	2.54	-	-	0.00	-3.02	-	-	-	-
10 - AVGs	26.88 - 233.55	12.00	1128.37	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	9.09	-3.13	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	26.88 - 233.55	11.00	1173.51	-	-	-	-	3.00	883.39	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-2.27	-	-	-	-	0.00	-13.17	-	-	-	-
12 - AVGs	26.88 - 233.55	10.00	1005.48	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-3.21	-	-	-	-	-	-	-	-	-	-
SI1-Like 18 -AVGs	26.88 - 233.55	12.92	1280.10	10.00	862.21	12.50	1436.69	3.00	995.30	3.00	598.67	3.38	1205.53
Time(secs.)	26.88 - 233.55	32.50	-	30.33	-	26.88	-	233.55	-	101.62	-	181.00	-
% to SI1-Like 16	27.33 - 191.09	0.65	1.49	0.00	1.16	-1.96	1.02	3.13	-5.14	0.00	0.67	0.00	-0.11
% to SI1-Like 15	22.12 - 178.82	1.31	-0.21	0.00	0.52	-1.96	0.55	-2.94	-2.79	0.00	0.67	0.00	-0.22
% to SI1-Like 09	19.62 - 154.18	0.00	-0.50	0.00	1.16	-1.96	-0.06	-2.94	-3.82	0.00	0.67	0.00	-0.11
% to SI1-Like 07	19 - 144.09	0.00	-0.62	0.00	0.97	-1.96	-0.18	-2.94	-4.90	0.00	0.26	0.00	-0.91
% to SI1-Like 06	19.75 - 141.91	0.00	-0.34	0.00	0.31	-1.96	0.43	-2.94	-4.72	0.00	0.56	0.00	-1.52
% to RVNSa [5]	2220	7.64	4.12	0.00	4.08	8.70	3.04	9.89	0.57	0.00	1.42	3.85	5.65
% to AD [14]	132 - 253	0.68	-7.67	0.00	-9.75	0.00	-7.07	-2.91	-27.16	0.00	-16.54	-0.15	-24.56
% to PR2 [78]	180	-3.10	-7.37	0.00	-4.51	-5.66	-7.03	-8.26	-23.05	-4.15	-8.35	-13.02	-24.42
% to DACS 01	300 - 400	-8.28	-15.94	0.00	-11.74	-9.09	-15.22	-5.71	-26.15	0.00	-13.71	-8.99	-23.85
% to DACS 02	300	-2.52	2.81	0.00	0.73	-2.91	1.02	-5.71	-6.44	-2.70	-2.01	-7.95	-5.33
% to DACS 03	300	1.17	3.89	0.00	3.66	0.33	2.34	-4.16	3.10	0.00	1.10	-2.41	3.71
% to DACS+HLS	300	2.51	3.39	0.00	3.36	2.99	2.00	0.00	1.66	0.00	0.91	0.12	2.74
% to DACS+HLS+2-Opt	300	2.22	4.56	0.00	4.02	3.20	2.95	-0.80	3.77	0.00	1.25	0.12	5.53
SI1-Like 16 - AVGs	27.33 - 191.09	12.83	1261.35	10.00	852.35	12.75	1422.25	2.91	1049.18	3.00	594.68	3.38	1206.82
SI1-Like 15 - AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CLi [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.14: Comparison between the average case performances of SI1-Like 19 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	26.12 - 235.09	20.00	1690.39	10.00	828.94	16.00	1699.48	4.00	1320.48	3.00	591.56	4.00	1471.20
% to SI1-Like 16	27.33 - 191.09	0.00	0.61	0.00	0.00	6.67	2.29	0.00	-1.58	0.00	0.00	0.00	0.13
02 - AVGs	26.12 - 235.09	18.00	1536.69	10.00	897.04	14.00	1578.38	3.00	1353.45	3.00	591.56	4.00	1256.63
% to SI1-Like 16	27.33 - 191.09	0.00	-0.59	0.00	-2.14	0.00	1.08	0.00	5.96	0.00	0.00	0.00	0.00
03 - AVGs	26.12 - 235.09	14.00	1280.77	10.00	913.85	11.00	1404.43	3.00	985.01	3.00	603.70	3.00	1173.43
% to SI1-Like 16	27.33 - 191.09	0.00	-0.09	0.00	0.00	-8.33	4.07	0.00	0.00	0.00	2.12	0.00	-1.23
04 - AVGs	26.12 - 235.09	10.00	1076.89	10.00	985.62	11.00	1170.52	3.00	839.00	3.00	648.56	3.00	924.02
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	13.69	0.00	-3.95	50.00	-9.70	0.00	3.08	0.00	1.81
05 - AVGs	26.12 - 235.09	14.00	1422.90	10.00	828.94	15.00	1656.19	3.00	1068.63	3.00	588.88	4.00	1414.98
% to SI1-Like 16	27.33 - 191.09	0.00	-1.33	0.00	0.00	0.00	0.00	0.00	-0.80	0.00	0.00	0.00	-0.28
06 - AVGs	26.12 - 235.09	13.00	1299.92	10.00	828.94	12.00	1446.30	3.00	982.38	3.00	588.49	3.00	1255.84
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	0.00	-1.32	0.00	-0.74	0.00	0.00	0.00	0.02
07 - AVGs	26.12 - 235.09	11.00	1137.46	10.00	828.94	12.00	1307.94	3.00	886.65	3.00	588.29	3.00	1169.53
% to SI1-Like 16	27.33 - 191.09	0.00	-1.43	0.00	0.00	0.00	-0.14	0.00	-22.08	0.00	0.00	0.00	-1.15
08 - AVGs	26.12 - 235.09	10.00	1010.45	10.00	828.94	11.00	1155.30	2.00	777.32	3.00	588.32	3.00	927.99
% to SI1-Like 16	27.33 - 191.09	0.00	-0.57	0.00	0.00	0.00	0.00	0.00	-0.74	0.00	0.00	0.00	-4.85
09 - AVGs	26.12 - 235.09	12.00	1234.00	10.00	828.94	-	-	3.00	983.94	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.07	0.00	0.00	-	-	0.00	-1.75	-	-	-	-
10 - AVGs	26.12 - 235.09	11.00	1162.05	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.24	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	26.12 - 235.09	11.00	1235.68	-	-	-	-	3.00	896.71	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	2.91	-	-	-	-	0.00	-11.86	-	-	-	-
12 - AVGs	26.12 - 235.09	10.00	1043.59	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	0.46	-	-	-	-	-	-	-	-	-	-
SI1-Like 19 -AVGs	26.12 - 235.09	12.83	1260.90	10.00	863.35	12.75	1427.32	3.00	1008.60	3.00	598.67	3.38	1199.20
Time(secs.)	26.12 - 235.09	30.17	-	32.33	-	26.12	-	235.09	-	108.88	-	169.88	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.04	0.00	1.29	0.00	0.36	3.13	-3.87	0.00	0.67	0.00	-0.63
% to SI1-Like 15	22.12 - 178.82	0.65	-1.70	0.00	0.65	0.00	-0.11	-2.94	-1.50	0.00	0.67	0.00	-0.74
% to SI1-Like 09	19.62 - 154.18	-0.65	-2.00	0.00	1.29	0.00	-0.71	-2.94	-2.53	0.00	0.67	0.00	-0.63
% to SI1-Like 07	19 - 144.09	-0.65	-2.11	0.00	1.10	0.00	-0.83	-2.94	-3.63	0.00	0.26	0.00	-1.43
% to SI1-Like 06	19.75 - 141.91	-0.65	-1.84	0.00	0.44	0.00	-0.23	-2.94	-3.44	0.00	0.56	0.00	-2.03
% to RVNSa [5]	2220	6.94	2.56	0.00	4.22	10.87	2.37	9.89	1.92	0.00	1.42	3.85	5.09
% to AD [14]	132 - 253	0.03	-9.06	0.00	-9.63	2.00	-7.67	-2.91	-26.19	0.00	-16.54	-0.15	-24.96
% to PR2 [78]	180	-3.73	-8.76	0.00	-4.38	-3.77	-7.63	-8.26	-22.02	-4.15	-8.35	-13.02	-24.82
% to DACS 01	300 - 400	-8.88	-17.20	0.00	-11.62	-7.27	-15.78	-5.71	-25.17	0.00	-13.71	-8.99	-24.25
% to DACS 02	300	-3.14	1.27	0.00	0.87	-0.97	0.36	-5.71	-5.19	-2.70	-2.01	-7.95	-5.82
% to DACS 03	300	0.52	2.33	0.00	3.80	2.34	1.67	-4.16	4.48	0.00	1.10	-2.41	3.16
% to DACS+HLS	300	1.85	1.84	0.00	3.49	5.05	1.33	0.00	3.02	0.00	0.91	0.12	2.20
% to DACS+HLS+2-Opt	300	1.56	3.00	0.00	4.16	5.26	2.27	-0.80	5.16	0.00	1.25	0.12	4.98
SI1-Like 16 - AVGs	27.33 - 191.09	12.83	1261.35	10.00	852.35	12.75	1422.25	2.91	1049.18	3.00	594.68	3.38	1206.82
SI1-Like 15 - AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Table G.15: Comparison between the average case performances of SI1-Like 20 and other VRPTW algorithms, after only one run, on the problem group PG100. This table represents also the best and worst case scenarios of the SI1-Like performance. On each problem instance and set, the SD values equal to zero.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	28 - 231.09	20.00	1669.74	10.00	828.94	15.00	1675.05	4.00	1343.87	3.00	591.56	4.00	1503.15
% to SI1-Like 16	27.33 - 191.09	0.00	-0.62	0.00	0.00	0.00	0.82	0.00	0.16	0.00	0.00	0.00	2.31
02 - AVGs	28 - 231.09	18.00	1524.07	10.00	883.86	14.00	1535.91	4.00	1166.44	3.00	591.56	4.00	1182.78
% to SI1-Like 16	27.33 - 191.09	0.00	-1.41	0.00	-3.58	0.00	-1.64	33.33	-8.68	0.00	0.00	0.00	-5.88
03 - AVGs	28 - 231.09	14.00	1273.82	10.00	913.85	11.00	1306.88	3.00	985.01	3.00	603.70	3.00	1196.73
% to SI1-Like 16	27.33 - 191.09	0.00	-0.63	0.00	0.00	-8.33	-3.16	0.00	0.00	0.00	2.12	0.00	0.73
04 - AVGs	28 - 231.09	10.00	1049.89	10.00	985.62	11.00	1164.21	2.00	963.46	3.00	624.54	3.00	880.22
% to SI1-Like 16	27.33 - 191.09	0.00	-2.51	0.00	13.69	0.00	-4.46	0.00	3.70	0.00	-0.74	0.00	-3.01
05 - AVGs	28 - 231.09	14.00	1442.08	10.00	828.94	15.00	1577.51	3.00	1061.34	3.00	588.88	4.00	1425.71
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	0.00	-4.75	0.00	-1.47	0.00	0.00	0.00	0.48
06 - AVGs	28 - 231.09	13.00	1299.92	10.00	828.94	13.00	1424.21	3.00	969.39	3.00	588.49	3.00	1303.69
% to SI1-Like 16	27.33 - 191.09	0.00	0.00	0.00	0.00	8.33	-2.83	0.00	-2.05	0.00	0.00	0.00	3.83
07 - AVGs	28 - 231.09	11.00	1162.52	10.00	828.94	12.00	1299.84	3.00	905.91	3.00	588.29	3.00	1184.50
% to SI1-Like 16	27.33 - 191.09	0.00	0.74	0.00	0.00	0.00	-0.76	0.00	-20.39	0.00	0.00	0.00	0.12
08 - AVGs	28 - 231.09	10.00	992.88	10.00	828.94	11.00	1155.30	2.00	764.08	3.00	588.32	3.00	1010.42
% to SI1-Like 16	27.33 - 191.09	0.00	-2.30	0.00	0.00	0.00	0.00	0.00	-2.43	0.00	0.00	0.00	3.60
09 - AVGs	28 - 231.09	12.00	1277.54	10.00	828.94	-	-	3.00	977.94	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	3.46	0.00	0.00	-	-	0.00	-2.35	-	-	-	-
10 - AVGs	28 - 231.09	11.00	1161.42	-	-	-	-	3.00	1000.98	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.29	-	-	-	-	0.00	0.00	-	-	-	-
11 - AVGs	28 - 231.09	11.00	1215.81	-	-	-	-	3.00	874.15	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	1.25	-	-	-	-	0.00	-14.08	-	-	-	-
12 - AVGs	28 - 231.09	10.00	1032.74	-	-	-	-	-	-	-	-	-	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.59	-	-	-	-	-	-	-	-	-	-
SI1-Like 20 -AVGs	28 - 231.09	12.83	1258.54	10.00	861.89	12.75	1392.36	3.00	1001.14	3.00	595.67	3.38	1210.90
Time(secs.)	28 - 231.09	31.75	-	29.11	-	28.00	-	231.09	-	105.12	-	167.75	-
% to SI1-Like 16	27.33 - 191.09	0.00	-0.22	0.00	1.12	0.00	-2.10	3.13	-4.58	0.00	0.17	0.00	0.34
% to SI1-Like 15	22.12 - 178.82	0.65	-1.89	0.00	0.48	0.00	-2.56	-2.94	-2.22	0.00	0.17	0.00	0.23
% to SI1-Like 09	19.62 - 154.18	-0.65	-2.18	0.00	1.12	0.00	-3.14	-2.94	-3.25	0.00	0.17	0.00	0.34
% to SI1-Like 07	19 - 144.09	-0.65	-2.29	0.00	0.93	0.00	-3.26	-2.94	-4.35	0.00	-0.24	0.00	-0.46
% to SI1-Like 06	19.75 - 141.91	-0.65	-2.02	0.00	0.27	0.00	-2.67	-2.94	-4.16	0.00	0.06	0.00	-1.08
% to RVNSa [5]	2220	6.94	2.36	0.00	4.04	10.87	-0.14	9.89	1.16	0.00	0.91	3.85	6.12
% to AD [14]	132 - 253	0.03	-9.23	0.00	-9.79	2.00	-9.93	-2.91	-26.74	0.00	-16.96	-0.15	-24.23
% to PR2 [78]	180	-3.73	-8.93	0.00	-4.54	-3.77	-9.90	-8.26	-22.60	-4.15	-8.81	-13.02	-24.09
% to DACS 01	300 - 400	-8.88	-17.35	0.00	-11.77	-7.27	-17.84	-5.71	-25.72	0.00	-14.14	-8.99	-23.51
% to DACS 02	300	-3.14	1.08	0.00	0.70	-0.97	-2.10	-5.71	-5.89	-2.70	-2.50	-7.95	-4.90
% to DACS 03	300	0.52	2.14	0.00	3.62	2.34	-0.82	-4.16	3.71	0.00	0.59	-2.41	4.17
% to DACS+HLS	300	1.85	1.65	0.00	3.32	5.05	-1.15	0.00	2.26	0.00	0.41	0.12	3.20
% to DACS+HLS+2-Opt	300	1.56	2.80	0.00	3.98	5.26	-0.23	-0.80	4.38	0.00	0.74	0.12	6.00
SI1-Like 16 - AVGs	27.33 - 191.09	12.83	1261.35	10.00	852.35	12.75	1422.25	2.91	1049.18	3.00	594.68	3.38	1206.82
SI1-Like 15 - AVGs	22.12 - 178.82	12.75	1282.76	10.00	857.75	12.75	1428.89	3.09	1023.90	3.00	594.68	3.38	1208.17
SI1-Like 09 - AVGs	19.62 - 154.18	12.92	1286.59	10.00	852.35	12.75	1437.57	3.09	1034.79	3.00	594.68	3.38	1206.82
SI1-Like 07 - AVGs	19 - 144.09	12.92	1288.03	10.00	853.95	12.75	1439.30	3.09	1046.63	3.00	597.10	3.38	1216.54
SI1-Like 06 - AVGs	19.75 - 141.91	12.92	1284.50	10.00	859.58	12.75	1430.55	3.09	1044.57	3.00	595.34	3.38	1224.11
AKRed [60] - AVGs	50 - 223	12.50	1241.89	10.00	834.05	12.38	1408.87	2.91	995.39	3.00	591.78	3.38	1139.70
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.62	3.00	590.30	3.25	1141.07
CL1 [77] - AVGs	300	12.42	1233.34	10.00	828.38	12.00	1403.74	3.09	990.99	3.00	596.63	3.38	1220.99
AD [14] - AVGs	132 - 253	12.83	1386.46	10.00	955.39	12.50	1545.92	3.09	1366.48	3.00	717.31	3.38	1598.06
PR2 [78] - AVGs	180	13.33	1381.90	10.00	902.90	13.25	1545.30	3.27	1293.40	3.13	653.20	3.88	1595.10
PS [79] - AVGs	N/A	13.50	1242.40	10.00	843.84	13.50	1408.76	4.09	977.12	3.13	607.58	5.13	1111.37
DACS 01 - AVGs	300 - 400	14.08	1522.81	10.00	976.86	13.75	1694.65	3.18	1347.79	3.00	693.81	3.71	1583.11
DACS 02 - AVGs	300	13.25	1245.12	10.00	855.92	12.88	1422.21	3.18	1063.77	3.08	610.94	3.67	1273.33
DACS 03 - AVGs	300	12.77	1232.18	10.00	831.77	12.46	1403.85	3.13	965.34	3.00	592.17	3.46	1162.42
DACS+HLS - AVGs	300	12.60	1238.15	10.00	834.20	12.14	1408.55	3.00	979.02	3.00	593.26	3.37	1173.38
DACS+HLS+2-Opt - AVGs	300	12.64	1224.23	10.00	828.86	12.11	1395.58	3.02	959.13	3.00	591.29	3.37	1142.36

Appendix H

Algorithms related to DACS 01

This section mentions in Figures H.1, H.2 and H.3 the algorithms used in the system DACS 01 talked about in Section 4.5.

D1- Figure H.1 represents the algorithm of DACS 01.

D2- Figure H.2 represents the algorithm of the ants' routing builder in DACS 01.

D3- Figure H.3 represents the algorithm of the XCHNG local search used by the ants of DACS 01.

```
//Initialise phase

I1- Create an initial solution using the nearest neighbourhood
    heuristic NN that uses an unlimited number of vehicles;
I2- Assign the initial solution as the best global solution;
I3- V = the number of the vehicles used in the best global solution;

//Cycle phase

C1-Create two colonies VMIN and DMIN;
C2-Initialise VMIN with V-1 and DMIN with V;

while (the number of iterations is less than the maximum number of
       iterations allowed) do

      C3-Activate the cycle of VMIN;
      C4-Activate the cycle of DMIN;

      if (killFlag == true) do

        C5- V = the number of the vehicles used in the new best
              global solution;
        C6- Kill the two colonies VMIN and DMIN currently active;
        C7- Create two new colonies called VMIN and DMIN also;
        C8- Initialise VMIN with V-1 and DMIN with V;
        C9- killFlag = false;

      od if

od while
```

Figure H.1: The coordinator DACS 01

```
heuristicIndex = 0;
Put an inactive vehicle at the depot;

while (the heuristicIndex is less than the size of
       the customers in a problem instance) do

  visitArrangement = false;

  While (visitArrangement == false) do

    //Choose an edge.
    aCVpair = use the probabilistic state transition
              component with its exploitation and
              exploration parts;

    if (there is no feasible edge or CVPair) do
        break;

    Otherwise
        //Pick the customer of the chosen edge.
        chosenCustomer = getCustomerFromCVPair();
        //Pick the vehicle (either active or inactive)
        //of the chosen edge.
        chosenVehicle = getVehicleFromCVPair();

        Arrange a visit to the chosen customer by
        the chosen vehicle;
        Remove the customer from the list
        of candidates;
        Local update the pheromone trail of the
        feasible edge or CVPair already chosen;

    od if-otherwise

  od while

  if (there is no inactive vehicle at the depot) do
      In addition to all the vehicles already activated,
      put an inactive vehicle at the depot;
  od if

  heuristicIndex = heuristicIndex + 1;

od while

Return all the active vehicles to the depot;

Insert the remaining unvisited customers, if the
solution of an ant is infeasible;

Use the XCHNG local search to improve the solution
of an ant, only if the solution of the ant is feasible
and the ant is coming from the DMIN colony;
```

Figure H.2: Ants' routing builder in DACS 01.

```
//The XCHNG local search of ants…
If (an ant has built a feasible solution and is coming from the DMIN
colony) do

  while (the number of iterations is less than the maximum number of
         iterations allowed) do

      (a) Choose randomly two different tours from the solution of an
          ant;
      for (each tour of the two tours) do
          (b) Choose two random indices of customer nodes in a tour;
          (c) Choose the lowest index of the two indices selected as
              the start of a segment;
          (d) Choose the highest index of the two indices selected as
              the end of a segment;
          if (the difference value between the highest index and the
              lowest index is greater than 3) do
              (e) randomNumber = Generate a random number between 1
                                 and 3;
              (f) The highest index = the lowest index + randomNumber;
          od if
      od for

      (g) Exchange the two selected segments of the two tours randomly
          selected to create two new tours;
      If (the total of travelled distances of the two new tours is
          less than the total of travelled distances of the two
          tours selected randomly) do
          (h) Replace the two tours selected randomly with the two
              new tours;
      od if
  od while

od if
```

Figure H.3: The XCHNG local search.

Appendix I

Information related to VRPTW algorithms in the literature

This section mentions in Tables I.1 to I.10 information about the experiments done using the state-of-the-art VRPTW algorithms in the literature on the problem groups PG100, PG200 and PG400 in Section 2.2.

E1- Tables I.1, I.2 and I.3 represent the best results computed, by different VRPTW algorithms, for the 176 problem instances of the problem groups PG100, PG200 and PG400.

E2- Tables I.4, I.5 and I.6 represent the average, best and worst case performances of the algorithm SA+LNS [29] on the problem group PG100.

E3- Tables I.7, I.8 and I.9 represent the average, best and worst case performances of the algorithms RVNSa [5] and RVNSc [5] on the problem groups PG100, PG200 and PG400.

E4- Table I.10 represents information about the hardware and software features used by the VRPTW algorithms of the literature. These hardware and software features are such as the PC machine used and the speed, the RAM capacity and the operating system of that PC machine in addition to the programming language used in coding a particular algorithm and the number of runs done during the experimentation on that algorithm. Each VRPTW algorithm is run for an amount of CPU time in seconds and such piece of information is available in tables in previous chapters where it is appropriate.

Table I.1: The best results computed by different VRPTW algorithms on the problem group PG100 mentioned in Section 2.2.

	R1		C1		RC1		R2		C2		RC2	
PNo.	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	19.00	1645.79	10.00	828.94	14.00	1696.94	4.00	1252.37	3.00	591.56	4.00	1406.91
Ref.	JH	[80]	RT	[26]	TB	[31]	HG	[65]	RT	[26]	MBD	[81]
02 - Best	17.00	1486.12	10.00	828.94	12.00	1554.75	3.00	1191.70	3.00	591.56	3.00	1365.65
Ref.	RT	[26]	RT	[26]	TB	[31]	RGP	[66]	RT	[26]	GC	[82]
03 - Best	13.00	1292.68	10.00	828.06	11.00	1261.67	3.00	939.54	3.00	591.17	3.00	1049.62
Ref.	LL	[62]	RT	[26]	S98	[16]	MBD	[81]	RT	[26]	CC	[83]
04 - Best	9.00	1007.24	10.00	824.78	10.00	1135.48	2.00	825.52	3.00	590.60	3.00	798.41
Ref.	MBD	[81]	RT	[26]	CLM	[73]	SA+LNS	[29]	RT	[26]	MBD	[81]
05 - Best	14.00	1377.11	10.00	828.94	13.00	1629.44	3.00	994.42	3.00	588.88	4.00	1297.19
Ref.	RT	[26]	RT	[26]	BBB	[46]	RGP	[66]	RT	[26]	MBD	[81]
06 - Best	12.00	1251.98	10.00	828.94	11.00	1424.73	3.00	906.14	3.00	588.49	3.00	1146.32
Ref.	MBD	[81]	RT	[26]	BBB	[46]	SSSD	[84]	RT	[26]	JH	[80]
07 - Best	10.00	1104.66	10.00	828.94	11.00	1230.48	2.00	890.61	3.00	588.29	3.00	1061.14
Ref.	S97	[85]	RT	[26]	S97	[85]	RP	[86]	RT	[26]	SA+LNS	[29]
08 - Best	9.00	960.88	10.00	828.94	10.00	1139.82	2.00	726.75	3.00	588.32	3.00	828.14
Ref.	BBB	[46]	RT	[26]	TB	[31]	MBD	[81]	RT	[26]	IBK	[70]
09 - Best	11.00	1194.73	10.00	828.94	-	-	3.00	909.16	-	-	-	-
Ref.	HG	[65]	RT	[26]	-	-	JH	[80]	-	-	-	-
10 - Best	10.00	1118.59	-	-	-	-	3.00	939.34	-	-	-	-
Ref.	MBD	[81]	-	-	-	-	MBD	[81]	-	-	-	-
11 - Best	10.00	1096.72	-	-	-	-	2.00	892.71	-	-	-	-
Ref.	RGP	[66]	-	-	-	-	SA+LNS	[29]	-	-	-	-
12 - Best	9.00	982.14	-	-	-	-	-	-	-	-	-	-
Ref.	MACS-VRPTW	[4]	-	-	-	-	-	-	-	-	-	-
AVGs	11.92	1209.89	10.00	828.38	11.50	1384.16	2.73	951.66	3.00	589.86	3.25	1119.17

Table I.2: The best results computed by different VRPTW algorithms on the problem group PG200 mentioned in Section 2.2.

	R1		C1		RC1		R2		C2		RC2	
PNo.	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	19	5024.65	20	2704.57	18	3637.8	4	4501.8	6	1931.44	6	3103.48
Ref.	RVNSc	[5]	GH	[3]	AGES	[48]	AGES	[48]	GH	[3]	AGES	[48]
02 - Best	18	4049.69	18	2917.89	18	3269.3	4	3645.38	6	1863.16	5	2827.45
Ref.	AGES	[48]	SA+LNS	[29]	AGES	[48]	AGES	[48]	GH	[3]	MBD	[81]
03 - Best	18	3382.65	18	2708.08	18	3025.9	4	2883.16	6	1775.11	4	2613.12
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	MBD	[81]	RP	[86]
04 - Best	18	3067.93	18	2644.61	18	2852.62	4	1981.29	6	1705.05	4	2043.05
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
05 - Best	18	4112.88	20	2702.05	18	3419.81	4	3367.55	6	1878.85	4	2912.13
Ref.	AGES	[48]	GH	[3]	AGES	[48]	SAMOPT	[87]	SA+LNS	[29]	RP	[86]
06 - Best	18	3586.8	20	2701.04	18	3338.84	4	2914.56	6	1857.35	4	2975.13
Ref.	RP	[86]	GH	[3]	AGES	[48]	AGES	[48]	RVNSc	[5]	RP	[86]
07 - Best	18	3151.42	20	2701.04	18	3219.86	4	2453.62	6	1849.46	4	2529.3
Ref.	AGES	[48]	GH	[3]	AGES	[48]	AGES	[48]	GH	[3]	AGES	[48]
08 - Best	18	2963.9	18	2769.19	18	3109.44	4	1849.87	6	1820.53	4	2298.12
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	RP	[86]	AGES	[48]
09 - Best	18	3784.33	18	2642.82	18	3083.41	4	3111.41	6	1830.05	4	2175.61
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	RP	[86]	AGES	[48]
10 - Best	18	3307.78	18	2643.51	18	3012.52	4	2657	6	1806.6	4	2015.6
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	MBD	[81]	AGES	[48]
AVGs	18.10	3643.20	18.80	2713.48	18.00	3196.95	4.00	2936.56	6.00	1831.76	4.30	2549.30

Table I.3: The best results computed by different VRPTW algorithms on the problem group PG400 mentioned in Section 2.2.

	R1		C1		RC1		R2		C2		RC2	
PNo.	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	38	11084	40	7152.02	36	8813.43	8	9257.92	12	4116.05	11	6834.02
Ref.	RVNSc	[5]	MBD	[81]	RP	[86]	AGES	[48]	MBD	[81]	RP	[86]
02 - Best	36	9053.18	36	7733.55	36	7985.5	8	7649.87	12	3929.89	9	6355.59
Ref.	AGES	[48]	RP	[86]	AGES	[48]	RP	[86]	AGES	[48]	RP	[86]
03 - Best	36	7941.53	36	7082.13	36	7627.3	8	5988.02	12	3739.72	8	5055.02
Ref.	AGES	[48]	RP	[86]	AGES	[48]	AGES	[48]	GH	[3]	RP	[86]
04 - Best	36	7332.93	36	6816.17	36	7355.29	8	4300.95	12	3535.99	8	3635.04
Ref.	AGES	[48]	RP	[86]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
05 - Best	36	9437.28	40	7152.02	36	8321.91	8	7143.55	12	3939.42	9	6063.46
Ref.	AGES	[48]	MBD	[81]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
06 - Best	36	8534.05	40	7153.41	36	8304.99	8	6163.81	12	3875.94	8	5997.24
Ref.	AGES	[48]	MBD	[81]	AGES	[48]	AGES	[48]	AGES	[48]	RP	[86]
07 - Best	36	7710.41	39	7546.78	36	8051.71	8	5082.1	12	3894.13	8	5476.57
Ref.	AGES	[48]	RP	[86]	AGES	[48]	AGES	[48]	MBD	[81]	RP	[86]
08 - Best	36	7385.29	37	7546.32	36	7917.68	8	4051.98	12	3787.08	8	4854.16
Ref.	AGES	[48]	RP	[86]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
09 - Best	36	8878.19	36	7524.32	36	7890.45	8	6493.13	12	3876.1	8	4599.57
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
10 - Best	36	8227.49	36	6907.26	36	7716.32	8	5844.77	12	3684.89	8	4316.36
Ref.	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]	AGES	[48]
AVGs	36.20	8558.44	37.60	7261.40	36.00	7998.46	8.00	6197.61	12.00	3837.92	8.50	5318.70

Table I.4: The average case performance of SA+LNS [29], after five runs, on the problem group PG100 mentioned in Section 2.2. Check Tables I.5 and I.6 for more information about the best and worst case performances.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	1800	19.00	1650.80	10.00	828.94	14.80	1639.09	4.00	1300.26	3.00	591.56	4.00	1481.45
02 - AVGs	1800	17.00	1486.12	10.00	828.94	12.00	1554.75	3.60	1185.94	3.00	614.04	3.60	1312.92
03 - AVGs	1800	14.00	1214.48	10.00	828.07	11.00	1267.47	3.00	985.32	3.00	656.84	3.00	1109.04
04 - AVGs	1800	10.00	984.13	10.00	824.78	10.00	1144.97	2.40	833.51	3.00	619.72	3.00	850.46
05 - AVGs	1800	14.00	1401.83	10.00	828.94	13.60	1587.11	3.00	1050.06	3.00	588.88	4.00	1353.91
06 - AVGs	1800	12.00	1270.19	10.00	828.94	12.00	1378.52	3.00	981.85	3.00	607.99	3.00	1217.93
07 - AVGs	1800	10.20	1109.85	10.00	828.94	11.00	1231.85	2.20	912.30	3.00	607.78	3.00	1111.60
08 - AVGs	1800	9.20	976.07	10.00	828.94	10.00	1162.00	2.00	758.77	3.00	588.32	3.00	900.61
09 - AVGs	1800	11.20	1209.17	10.00	828.94	-	-	3.00	955.90	-	-	-	-
10 - AVGs	1800	10.20	1127.46	-	-	-	-	3.00	982.66	-	-	-	-
11 - AVGs	1800	10.20	1098.88	-	-	-	-	2.20	909.35	-	-	-	-
12 - AVGs	1800	10.00	971.79	-	-	-	-	-	-	-	-	-	-
SA+LNS [29] - AVGs	1800	12.25	1208.40	10.00	828.38	11.80	1370.72	2.85	986.90	3.00	609.39	3.33	1167.24

Table I.5: The best case performance of SA+LNS [29], after five runs, on the problem group PG100 mentioned in Section 2.2.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Best	1800	19.00	1650.80	10.00	828.94	14.00	1697.43	4.00	1287.67	3.00	591.56	4.00	1466.02
Freq.		5	5	5	5	1	1	5	1	5	5	5	1
02 - Best	1800	17.00	1486.12	10.00	828.94	12.00	1554.75	3.00	1237.04	3.00	591.56	3.00	1387.38
Freq.		5	5	5	5	5	5	2	1	5	1	2	1
03 - Best	1800	14.00	1213.62	10.00	828.07	11.00	1261.67	3.00	967.82	3.00	591.17	3.00	1097.31
Freq.		5	1	5	5	5	1	5	1	5	1	5	1
04 - Best	1800	10.00	981.23	10.00	824.78	10.00	1135.48	2.00	833.88	3.00	590.60	3.00	841.28
Freq.		5	1	5	5	5	1	3	1	5	1	5	1
05 - Best	1800	14.00	1387.14	10.00	828.94	13.00	1635.90	3.00	1036.83	3.00	588.88	4.00	1322.64
Freq.		5	1	5	5	2	1	5	1	5	5	5	1
06 - Best	1800	12.00	1257.96	10.00	828.94	12.00	1376.26	3.00	956.29	3.00	588.49	3.00	1187.28
Freq.		5	1	5	5	5	1	5	1	5	1	5	1
07 - Best	1800	10.00	1114.78	10.00	828.94	11.00	1230.95	2.00	901.09	3.00	588.29	3.00	1093.75
Freq.		4	1	5	5	5	1	4	1	5	1	5	1
08 - Best	1800	9.00	966.86	10.00	828.94	10.00	1139.82	2.00	737.37	3.00	588.32	3.00	875.61
Freq.		4	1	5	5	5	1	5	1	5	5	5	1
09 - Best	1800	11.00	1197.42	10.00	828.94	-	-	3.00	943.71	-	-	-	-
Freq.		4	1	5	5	-	-	5	1	-	-	-	-
10 - Best	1800	10.00	1126.63	-	-	-	-	3.00	968.00	-	-	-	-
Freq.		4	1	-	-	-	-	5	1	-	-	-	-
11 - Best	1800	10.00	1096.74	-	-	-	-	2.00	913.75	-	-	-	-
Freq.		4	1	-	-	-	-	4	1	-	-	-	-
12 - Best	1800	10.00	966.79	-	-	-	-	-	-	-	-	-	-
Freq.		5	1	-	-	-	-	-	-	-	-	-	-
SA+LNS [29] - AVGs	1800	12.17	1203.84	10.00	828.38	11.63	1379.03	2.73	980.31	3.00	589.86	3.25	1158.91

Table I.6: The worst case performance of SA+LNS [29], after five runs, on the problem group PG100 mentioned in Section 2.2.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - Worst	1800	19.00	1650.80	10.00	828.94	15.00	1627.29	4.00	1317.98	3.00	591.56	4.00	1519.08
Freq.		5	5	5	5	4	1	5	1	5	5	5	1
02 - Worst	1800	17.00	1486.12	10.00	828.94	12.00	1554.75	4.00	1166.00	3.00	703.99	4.00	1301.23
Freq.		5	5	5	5	5	5	3	1	5	1	3	1
03 - Worst	1800	14.00	1217.92	10.00	828.07	11.00	1278.55	3.00	1026.83	3.00	753.14	3.00	1125.80
Freq.		5	1	5	5	5	1	5	1	5	1	5	1
04 - Worst	1800	10.00	989.80	10.00	824.78	10.00	1156.05	3.00	798.70	3.00	672.16	3.00	865.93
Freq.		5	1	5	5	5	1	2	1	5	1	5	1
05 - Worst	1800	14.00	1426.17	10.00	828.94	14.00	1563.76	3.00	1061.80	3.00	588.88	4.00	1395.88
Freq.		5	1	5	5	3	1	5	1	5	5	5	1
06 - Worst	1800	12.00	1292.16	10.00	828.94	12.00	1387.57	3.00	1018.26	3.00	685.96	3.00	1239.49
Freq.		5	1	5	5	5	1	5	1	5	1	5	1
07 - Worst	1800	11.00	1072.12	10.00	828.94	11.00	1232.26	3.00	866.58	3.00	685.76	3.00	1130.36
Freq.		1	1	5	5	5	1	1	1	5	1	5	1
08 - Worst	1800	10.00	961.36	10.00	828.94	10.00	1193.45	2.00	773.32	3.00	588.32	3.00	914.76
Freq.		1	1	5	5	5	1	5	1	5	5	5	1
09 - Worst	1800	12.00	1166.24	10.00	828.94	-	-	3.00	980.10	-	-	-	-
Freq.		1	1	5	5	-	-	5	1	-	-	-	-
10 - Worst	1800	11.00	1114.28	-	-	-	-	3.00	1006.61	-	-	-	-
Freq.		1	1	-	-	-	-	5	1	-	-	-	-
11 - Worst	1800	11.00	1063.30	-	-	-	-	3.00	809.54	-	-	-	-
Freq.		1	1	-	-	-	-	1	1	-	-	-	-
12 - Worst	1800	10.00	986.75	-	-	-	-	-	-	-	-	-	-
Freq.		5	1	-	-	-	-	-	-	-	-	-	-
SA+LNS [29] - AVGs	1800	12.58	1202.25	10.00	828.38	11.88	1374.21	3.09	984.16	3.00	658.72	3.38	1186.57

Table I.7: The average, best and worst case performance of RVNSa [5], after only one run, on the problem group PG100 mentioned in Section 2.2.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	2220	19.00	1652.22	10.00	828.94	14.00	1698.82	4.00	1260.91	3.00	591.56	4.00	1428.09
02 - AVGs	2220	17.00	1486.95	10.00	828.94	12.00	1579.75	3.00	1198.56	3.00	591.56	3.00	1375.45
03 - AVGs	2220	13.00	1311.11	10.00	828.06	11.00	1280.14	3.00	957.02	3.00	591.17	3.00	1062.52
04 - AVGs	2220	10.00	999.59	10.00	824.78	10.00	1143.86	2.00	894.69	3.00	594.06	3.00	812.81
05 - AVGs	2220	14.00	1361.45	10.00	828.94	13.00	1632.34	3.00	1032.96	3.00	588.88	4.00	1326.83
06 - AVGs	2220	12.00	1262.49	10.00	828.94	11.00	1432.12	3.00	929.62	3.00	588.49	3.00	1197.46
07 - AVGs	2220	10.00	1155.29	10.00	828.94	11.00	1234.30	2.00	982.01	3.00	588.29	3.00	1071.48
08 - AVGs	2220	9.00	974.85	10.00	828.94	10.00	1152.77	2.00	737.17	3.00	588.32	3.00	853.90
09 - AVGs	2220	11.00	1238.58	10.00	828.94	-	-	3.00	944.94	-	-	-	-
10 - AVGs	2220	10.00	1132.37	-	-	-	-	3.00	975.26	-	-	-	-
11 - AVGs	2220	10.00	1139.44	-	-	-	-	2.00	972.55	-	-	-	-
12 - AVGs	2220	9.00	1019.41	-	-	-	-	-	-	-	-	-	-
RVNSa [5] - AVGs	2220	12.00	1229.48	10.00	828.38	11.50	1394.26	2.73	989.61	3.00	590.29	3.25	1141.07

Table I.8: The average, best and worst case performance of RVNSc [5], after only one run, on the problem group PG200 mentioned in Section 2.2.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	720 - 1680	19.00	5024.65	20.00	2704.57	18.00	4172.32	4.00	4854.36	6.00	1931.44	6.00	3167.78
02 - AVGs	720 - 1680	18.00	4234.35	18.00	3140.52	18.00	3674.81	4.00	3823.58	6.00	1863.16	5.00	2873.11
03 - AVGs	720 - 1680	18.00	3552.48	18.00	2832.99	18.00	3284.38	4.00	3023.78	6.00	1808.60	4.00	2743.38
04 - AVGs	720 - 1680	18.00	3177.67	18.00	2696.14	18.00	3044.29	4.00	2020.95	6.00	1754.79	4.00	2110.85
05 - AVGs	720 - 1680	18.00	4464.08	20.00	2702.05	18.00	3887.09	4.00	3485.74	6.00	1879.31	4.00	3379.67
06 - AVGs	720 - 1680	18.00	3824.87	20.00	2701.04	18.00	3705.53	4.00	3009.36	6.00	1857.35	5.00	2666.24
07 - AVGs	720 - 1680	18.00	3330.56	20.00	2701.04	18.00	3508.34	4.00	2534.06	6.00	1850.13	4.00	2704.06
08 - AVGs	720 - 1680	18.00	3085.72	19.00	2799.85	18.00	3341.86	4.00	1856.32	6.00	1822.65	4.00	2371.97
09 - AVGs	720 - 1680	18.00	4026.50	18.00	2775.27	18.00	3263.93	4.00	3135.40	6.00	1848.12	4.00	2231.63
10 - AVGs	720 - 1680	18.00	3493.38	18.00	2734.56	18.00	3198.18	4.00	2709.33	6.00	1808.72	4.00	2034.94
RVNSc [5] - AVGs	720 - 1680	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36

Table I.9: The average, best and worst case performance of RVNSc [5], after only one run, on the problem group PG400 mentioned in Section 2.2.

		R1		C1		RC1		R2		C2		RC2	
PNo.	Time(secs.)	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
01 - AVGs	3900 - 7980	19.00	5024.65	20.00	2704.57	18.00	4172.32	4.00	4854.36	6.00	1931.44	6.00	3167.78
02 - AVGs	3900 - 7980	18.00	4234.35	18.00	3140.52	18.00	3674.81	4.00	3823.58	6.00	1863.16	5.00	2873.11
03 - AVGs	3900 - 7980	18.00	3552.48	18.00	2832.99	18.00	3284.38	4.00	3023.78	6.00	1808.60	4.00	2743.38
04 - AVGs	3900 - 7980	18.00	3177.67	18.00	2696.14	18.00	3044.29	4.00	2020.95	6.00	1754.79	4.00	2110.85
05 - AVGs	3900 - 7980	18.00	4464.08	20.00	2702.05	18.00	3887.09	4.00	3485.74	6.00	1879.31	4.00	3379.67
06 - AVGs	3900 - 7980	18.00	3824.87	20.00	2701.04	18.00	3705.53	4.00	3009.36	6.00	1857.35	5.00	2666.24
07 - AVGs	3900 - 7980	18.00	3330.56	20.00	2701.04	18.00	3508.34	4.00	2534.06	6.00	1850.13	4.00	2704.06
08 - AVGs	3900 - 7980	18.00	3085.72	19.00	2799.85	18.00	3341.86	4.00	1856.32	6.00	1822.65	4.00	2371.97
09 - AVGs	3900 - 7980	18.00	4026.50	18.00	2775.27	18.00	3263.93	4.00	3135.40	6.00	1848.12	4.00	2231.63
10 - AVGs	3900 - 7980	18.00	3493.38	18.00	2734.56	18.00	3198.18	4.00	2709.33	6.00	1808.72	4.00	2034.94
RVNSc [5] - AVGs	3900 - 7980	18.10	3821.43	18.90	2778.80	18.00	3508.07	4.00	3045.29	6.00	1842.43	4.40	2628.36

Table I.10: The hardware and software features used and the number of runs done by the different VRPTW algorithms tried on the problem groups PG100, PG200 and PG400.

	Algo. Name	PC machine	Speed	RAM	Programming Language	Operating system	Number of runs
1	RS [59]	Pentium	200 MHz	N/A	N/A	N/A	1
2	AKRed [60]	Pentium	166 MHz	N/A	Fortran 77	N/A	1
3	RTa [26]	Silicon Graphics Indigo	100 MHz	N/A	N/A	N/A	1
4	RVNSa, RVNSb, RVNSc [5]	Pentium	200 MHz	N/A	Java	N/A	1
5	CL1 [77]	Pentium	300 MHz	N/A	N/A	N/A	1
6	I1 [1]	DEC 10	N/A	N/A	Fortran	N/A	1
7	I1-AD [14]	RS6000/530	N/A	N/A	ANSI-C	N/A	1
8	ES, ES4, ES4C [2]	4 X Pentium	200 MHz	N/A	C	N/A	1
9	LC03, LCK05 [33]	5 X Pentium III	850 MHz	512 MB	N/A	Linux	1
10	AGES [48]	Pentium IV	2 GHz	512 MB	Visual Basic 6.0	N/A	1
11	BBB [46]	Pentium	400 MHz	128 MB	C++	N/A	3
12	MACS-VRPTW [4]	Sun UltraSparc 1	167 MHz	N/A	C++	N/A	3
13	TD-MACS [53]	Pentium IV	2.66 GHz	N/A	N/A	N/A	3
14	HGA+TA [28]	Pentium	400 MHz	128 MB	C++	N/A	3
15	LS+TA [28]	AMD	700 MHz	128 MB	C++	N/A	3
14 and 15	LS+HGA+TA [28]	Pentium and AMD	400 and 700 MHz	128 MB	C++	N/A	6
16	KPS [15]	DEC Alpha	25 Mflops/s	N/A	N/A	N/A	3
17	HG05a [64]	Pentium	400 MHz	N/A	N/A	N/A	3
18	ALV [61]	Pentium 4	2.4 GHz	N/A	N/A	N/A	3
19	LL [62]	Pentium III	545 MHz	N/A	C++	Linux kernel 2.4.0	3
20	BRDU1, BRDU2 [63]	AMD	700 MHz	128 MB	Java	N/A	3
21	MSLS1, MSLS+TA1 [17]	AMD	700 MHz	128 MB	C++	Win 98	3
22	PR1 [11]	IBM PC	N/A	N/A	N/A	N/A	3
23	TP [13]	PC/AT	12 MHz	N/A	N/A	N/A	4
24	AD [14]	RS6000/530	N/A	N/A	ANSI-C	N/A	4
25	HG05b [64]	Pentium	400 MHz	N/A	N/A	N/A	5
26	RTb [26]	Silicon Graphics Indigo	100 MHz	N/A	N/A	N/A	5
27	TB [31]	Sun Sparc 10	50 MHz	N/A	N/A	N/A	5
28	HGA+EA [27]	Pentium Celeron	366 MHz	N/A	C++ and Java	N/A	5
29	SA+LNS [29]	Sun Ultra 10	440 MHz	256 MB	C++	N/A	5
30	HM4, HM4C [3]	4 X Pentium	400 MHz	N/A	N/A	N/A	5
31	HGES1, HGES2 [65]	Pentium	200 MHz	N/A	C	N/A	10
32	RGP [66]	Sun Ultra 10	N/A	N/A	N/A	N/A	10
33	MSSA [30]	Centrino	1.7 GHz	N/A	N/A	N/A	15
34	MSLS2, MSLS+TA2 [17]	AMD	700 MHz	128 MB	C++	Win 98	30
35	VGA1, VGA2 [7]	Pentium III	1 GHz	N/A	C++	N/A	100
36	LSH [67]	HP-9000	720 MHz	N/A	Fortran 77	HP-UX 9.01	N/A
37	TH [25]	NeXT 68040	25 Mhz	N/A	C	N/A	N/A
38	PB [41]	Sun Sparc 10	N/A	N/A	N/A	N/A	N/A
39	TLOL [68]	N/A	N/A	N/A	N/A	N/A	N/A
40	DBF [69]	N/A	N/A	N/A	N/A	N/A	N/A
41	ILS, AMLS [70]	Pentium III	800 MHz	256 MB	C	N/A	N/A
42	CR1 [71]	486DX	66 MHz	N/A	N/A	N/A	N/A
43	CR2 [24]	486DX	66 MHz	N/A	N/A	N/A	N/A
44	CR3 [72]	Pentium	166 MHz	N/A	N/A	N/A	N/A
45	CLM [73]	Sun Ultra 2	300 MHz	N/A	N/A	N/A	N/A
46	PR2 [78]	Sparc	N/A	N/A	N/A	N/A	N/A
47	PS [79]	N/A	N/A	N/A	N/A	N/A	N/A

Appendix J

Published work

We have two published articles and they are located in the section of Bibliography with the reference numbers [55] and [56].

Bibliography

[1] Solomon, M.M.: Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. Operations Research 35(2) (1987) pp. 254–265.

[2] Gehring, H., Homberger, J.: A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. K. Miettinen, M. Mäkelä, J. Toivanen, eds. Proceedings of EUROGEN99-Short Course on Evolutionary Algorithms in Engineering and Computer Science, Reports of the Department of Mathematical Information Technology, Series A. Collections, No. A 2/1999. University of Jyväskylä, Jyväskylä, Finland, (1999) pp. 57–64.

[3] Gehring, H., Homberger, J.: Parallelization of a Two-Phase Metaheuristic for Routing Problems with Time Windows. Asia-Pacific Journal of Operational Research 18, (2001) pp. 35–47.

[4] Gambardella, L.M., Taillard, E., Agazzi, G.: MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne, M. Dorigo and F. Glover, editors, New Ideas in Optimization. McGraw-Hill, UK, (1999) pp. 63–76. Also available as Technical Report IDSIA-06-99, IDSIA, Lugano, Switzerland.

[5] Bräysy, O.: A Reactive Variable Neighbourhood Search for the Vehicle Routing Problem with Time Windows. INFORMS Journal on Computing 2002 INFORMS Vol. 00, No. 0, (2002) pp. 1–22.

[6] Ellabib, I., Basir, O.A., Calamai, P.: An Experimental Study of a Simple Ant Colony System for the Vehicle Routing Problem with Time Windows. M. Dorigo et al. (Eds.): ANTS 2002, LNCS 2463, Springer-Verlag Berlin Heidelberg, (2002) pp. 53–64.

[7] Jung, S., Moon, B.R.: A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows. Proceedings of 4th International Conference on Genetic and Evolutionary Computation (GECCO 2002), New York City, New York, USA, Morgan Kaufmann, San Francisco California, (2002) pp 1309–1316.

[8] Bräysy, O.: Local search and variable neighborhood search algorithms for the vehicle routing problem with time windows. Doctoral thesis, Department of Mathematics and Statistics, University of Vaasa, Finland, (2001) .

[9] Larsen, J.: Parallelization of the vehicle routing problem time windows. Ph.D. thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, (1999) .

[10] Larson, R.C., Odoni, A.R.: Urban Operations Research. first published by Prentice-Hall, NJ. Massachusetts Institute of Technology 1997-99, (1981) http://web.mit.edu/urban_or_book/www/book/chapter6/6.4.12.html.

[11] Potvin, J.Y., Dubé, D., Robillard, C.: A Hybrid Approach to Vehicle Routing using Neural Network and Genetic Algorithms. Applied Intelligence 6:3, (1996) pp. 241–252.

[12] Kohout, R., Erol, K.: In-Time Agent-Based Vehicle Routing with a Stochastic Improvement Heuristic. Eleventh Conference on Innovative Applications of Artificial Intelligence, Orlando, FL, (1999) , Available at http://www.cs.umd.edu/users/kohout/.

[13] Thompson, P.M., Psaraftis, H.N.: Cyclic transfer algorithms for multivehicle routing and scheduling problems. Operations Research 41:5 (1993) pp. 935–946.

[14] Antes, J., Derigs, U.: A New Parallel Tour Construction Algorithm for the Vehicle Routing Problem with Time Windows. Working Paper, Department of Economics and Computer Science, University of Köln, Germany. (1995)

[15] Kilby, P., Prosser, P., Shaw, P.: Guided Local Search for the Vehicle Routing Problems With Time Windows. In Meta-heuristics: Advances and Trends in Local Search for Optimisation, S. Voss, S. Martello, I. H. Osman and C. Roucairol (eds.), Kluwer Academic Publishers, Boston, (1999) pp. 473–486.

[16] Shaw, P.: Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. Proceedings of the Fourth International Conference on Principles and Practice of Constraint Programming (CP '98), M. Maher and J.-F. Puget (eds.), Springer-Verlag, (1998) pp. 417–431.

[17] Bräysy, O., Hasle, G., Dullaert, W.: A Multi-Start Local Search Algorithm for the Vehicle Routing Problem with Time Windows. European Journal of Operational Research, Volume 159, Issue 3 (2004) pp. 586–605.

[18] Bäck, T., Fogel, D.B., Michalewicz, Z.: Evolutionary Computation 1; Basic Algorithms and Operators. Number ISBN 0-7503-0664-5. IOP Publishing Ltd (2000) .

[19] Ferber, J.: Multi-Agent Systems; An Introduction to Distributed Artificial Intelligence. Number ISBN 0-201-36048-9. Addison-Wesley (1999) .

[20] Taillard, E.D., Gambardella, L.M., Gendreau, M., Potvin, J.Y.: Adaptive Memory Programming: A Unified View of Metaheuristics. Technical Report IDSIA-19-98, IDSIA, Lugano, Switzerland, 1998. Also published in EURO XVI Conference Tutorial and Research Reviews booklet (semi-plenary sessions), Brussels, (1998) , http://citeseer.nj.nec.com/44690.html.

[21] Louis, S.J., Yin, X., Yuan, Z.Y.: Multiple Vehicle Routing with Time Windows Using Genetic Algorithms. Proceedings of the Congress on Evolutionary Computation - CEC99, July 1999 (1999) http://www.cs.unr.edu/ sushil/papers/conference/.

[22] Tan, K.C., Hay, L.L., O., K.: Hybrid Genetic Algorithm in Solving Vehicle Routing Problems with Time Windows Constraints. (2000) , Available at http://www.ise.nus.edu.sg/proceedings/apors2000/fullpapers/38-02-fp.htm.

[23] Xu, J., Kelly, J.P.: A Network Flow-based Tabu Search Heuristic for the Vehicle Routing Problem. Working paper, 34, (1996) , Available at http://citeseer.nj.nec.com/xu96network.html.

[24] Chiang, W.C., Russel, R.: Simulated annealing metaheuristics for the vehicle routing problem with time windows. Annals of Operations Research 63, (1996) pp. 3–27.

[25] Thangiah, S., Osman, I., Sun, T.: Hybrid Genetic Algorithm Simulated Annealing and Tabu Search Methods for Vehicle Routing Problem with Time Windows. Technical Report 27, Computer Science Department, Slippery Rock University, (1994) .

[26] Rochat, Y., Taillard, E.D.: Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. Journal of Heuristics 1 (1995) pp. 147–167.

[27] Bräysy, O., Berger, J., Barkaoui, M.: A new hybrid evolutionary algorithm for the vehicle routing problem with time windows. Presented at the Route 2000-Workshop, Skodsborg, Denmark (2000) .

[28] Bräysy, O., Berger, J., Barkaoui, M., Dullaert, W.: A Threshold Accepting Metaheuristic for the Vehicle Routing Problem with Time Windows. Central European Journal of Operations Research, 4 / (2003) .

[29] Bent, R., Hentenryck, P.V.: A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows. Transportation Science, 38(4) (2004) pp. 515–530.

[30] de Oliveira, H.C.B., Vasconcelos, G.C., Alvarenga, G.B.: Reducing Traveled Distance in the Vehicle Routing Problem with Time Windows using a Multi-Start Simulated Annealing. Proceedings of the International Joint Conference on Neural Networks, July (2006) pp. 5320–5327.

[31] Taillard, E.D., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.Y.: A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation science 31 (1997) pp. 170–186.

[32] Bouthillier, A.L., Crainic, T.G., Kropf, P.: A Guided Cooperative Search for the Vehicle Routing Problem with Time Windows. IEEE Intelligent Systems, vol.20, no.4, (2005) pp. 36–42.

[33] Bouthillier, A.L., Crainic, T.G.: A Cooperative Parallel Meta-Heuristic for the Vehicle Routing Problem with Time Windows. Computers and Operations Research 32(7) (2005) pp. 1685–1708.

[34] Bouthillier, A.L., Crainic, T.G., Kropf, P.: Towards a Guided Cooperative Search. Publication CRT-05-09, Centre for Research on Transportation, Univ. de Montreal, (2005) .

[35] Bäck, T., Fogel, D.B., Michalewicz, Z.: Evolutionary Computation 2; Advanced Algorithms and Operators. Number ISBN 0-7503-0665-3. IOP Publishing Ltd (2000) .

[36] Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Number ISBN 0-201-15767-5. Addison Wesley (1989) .

[37] Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Third, Revised and Extended Edition, Chapter 10, Transportation problems, Springer-Verlag (1999) .

[38] Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Number ISBN 0-471-87339-X. John Wiley and Sons LTD, (2001) .

[39] Benhamida, S., Roli, A., Belgasem, A., Tavares, J., Saadah, S.: Solving BCSP using GA and PBIL approaches. A working paper presented to the EvoNet Summer School 2001, Technological Educational Institution of Thessaloniki, Greece, (2001) .

[40] Urquhart, N., Ross, P., Paechter, B., Chisholm, K.: Improving street based routing using building block mutation. Applications of Evolutionary Computing, S. Cagnoni et al. (Eds.): EvoWorkshops 2002, LNCS 2279 Springer-Verlag Berlin Heidelberg, (2002) pp. 334–341.

[41] Potvin, J.Y., Bengio, S.: The Vehicle Routing Problem with Time Windows - Part II: Genetic Search. INFORMS Journal of Computing 8, (1996) pp. 165–172.

[42] Lanzi, P.L., Stolzmann, W., Wilson, S.W.: Learning Classifier Systems: From Foundations to Applications. Number ISBN 3540677291. (Lecture Notes in Computer Science, 1813.), Paperback - 347 pages 1st edition (July 2000), Springer Verlag; (2000) .

[43] Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming; An Introduction On the Automatic Evolution of Computer Programs and Its Applications. Number ISBN 1-55860-510-X. Morgan Kaufmann Publishers, Inc. San Francisco, California, (1998) .

[44] Yao, X., Liu, Y.: Fast evolution strategies. Control and Cybernetics, 26(3) (1997) pp. 467–496.

[45] Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation, 3(2) (1999) pp. 82–102.

[46] Berger, J., Barkaoui, M., Bräysy, O.: A Route-directed Hybrid Genetic Approach for the Vehicle Routing Problem with Time Windows. Information Systems and Operational Research 41:2 (2003) pp. 179–194.

[47] Jung, S., Moon, B.R.: The Natural Crossover for the 2D Euclidean TSP. Genetic and Evolutionary Computation Conference, (2002) , Available at http://soar.snu.ac.kr/publication.html.

[48] Mester, D., Bräysy, O.: Active Guided Evolution Strategies for Large Scale Vehicle Routing Problems with Time Windows. Computers and Operations Research 32(6) (2005) pp. 1593–1614.

[49] Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1), (1996) pp. 29–41.

[50] Gambardella, L.M., Dorigo, M.: Solving Symmetric and Asymmetric TSPs by Ant Colonies. ICEC96, Proceedings of the IEEE Conference on Evolutionary Computation, Nagoya, Japan, May 20-22. (1996) .

[51] Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation 1 (1997) pp. 53–66.

[52] Gambardella, L.M., Dorigo, M.: HAS-SOP: Hybrid Ant System for the Sequential Ordering Problem. Technical Report IDSIA-11-97, IDSIA, Lugano, Switzerland, (1997) .

[53] Donati, A.V., Montemanni, R., Casagrande, N., Rizzoli, A.E., Gambardella, L.M.: Time Dependent Vehicle Routing Problem with a Multi Ant Colony System. To appear in the European Journal of Operational Research (2007) .

[54] Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Number ISBN 0-19-513159-2. Oxford University Press, (1999) .

[55] Sa'adah, S., Ross, P., Paechter, B.: Improving Vehicle Routing Using A Customer Waiting Time Colony. Proceedings of the 4th European Conference on Evolutionary Computation in Combinatorial Optimization, Coimbra, Portugal, Springer-Verlag, Berlin (2004) pp. 188–198.

[56] Sa'adah, S., Ross, P., Paechter, B.: Solving Vehicle Routing Using Different Multiple Ant Colony Systems. Proceedings of the 5th International Conference on Recent Advances in Soft Computing, RASC2004, Nottingham, UK (2004) pp. 560–565.

[57] Reimann, M., Doerner, K., Doerner, K.: A Savings based Ant System for the Vehicle Routing Problem. Proceedings of 4th International Conference on Genetic and Evolutionary Computation (GECCO 2002), New York City, New York, USA, Morgan Kaufmann, San Francisco California, (2002) pp. 1317–1325.

[58] Zhu, Q., Qian, L., Li, Y., Zhu, S.: An improved Particle Swarm Optimization Algorithm for Vehicle Routing Problem with Time Windows. Proceedings of the IEEE Congress on Evolutionary Computation - CEC2006, July (2006) pp. 5535–5539.

[59] Riise, A., Stlevik, M.: Implementation of Guided Local Search for the Vehicle Routing Problem. SINTEF Internal report STF42 A99013, SINTEF Applied Mathematics, Norway, (1999) .

[60] Cordone, R., Wolfer-Calvo, R.: A heuristic for the vehicle routing problem with time windows. Journal of Heuristics 7 (2001) pp 107–129.

[61] Alvarenga, G.B.: Um algoritmo hbrido para os problemas de roteamento de veculos esttico e dinmico com janela de tempo. PhD Thesis, Department of Science Computer, Federal University of Minas Gerais, Brazil, (2005) .

[62] Li, H., Lim, A., Huang, J.: Local search with annealing-like restarts to solve the VRPTW. European Journal of Operational Research, volume 150, No 1, (2003) pp. 115–127.

[63] Bräysy, O., Dullaert, W.: A Fast Evolutionary Metaheuristic for the Vehicle Routing Problem with Time Windows. International Journal on Artificial Intelligence Tools, 12, (2003) pp. 153–172.

[64] Homberger, J., Gehring, H.: A two phase hybrid metaheuristic for the vehicle routing problem with time windows. European Journal of Operational Research, Vol. 162, (2005) pp. 220–238.

[65] Homberger, J., Gehring, H.: Two evolutionary metaheuristics for the vehicle routing problem with time windows. Information Systems and Operational Research - Special issue: Metaheuristics for Location and Routing Problems 37, (1999) pp. 297–318.

[66] Rousseau, L.M., Gendreau, M., Pesant, G.: Using constraint-based operators to solve the vehicle routing problem with time windows. Journal of Heuristics 8, (2002) pp. 43–58.

[67] Liu, F.H., Shen, S.Y.: A Route-Neighborhood-based Metaheuristic for Vehicle Routing Problem with Time Windows. European Journal of Operational Research 118, (1999) pp. 485–504.

[68] Tan, K., Lee, T., Ou, K., Lee, L.: A messy genetic algorithm for the vehicle routing problem with time window constraints. In IEEE Congress on Evolutionary Computation (2001) pp. 679–686.

[69] Backer, B.D., Furnon, V.: Meta-heuristics in Constraint Programming Experiments with Tabu Search on the Vehicle Routing Problem. presented at the Second International Conference on Metaheuristics (MIC97), Sophia Antipolis, France, (1997) .

[70] Ibaraki, T., Kubo, M., Masuda, T., Uno, T., Yagiura, M.: Effective Local Search Algorithms for the Vehicle Routing Problem with General Time Window Constraints. 4th Metaheuristics International Conference (MIC2001), Porto, Portugal, (2001) pp. 293–297.

[71] Chiang, W.C., Russel, R.: Hybrid Heuristics for the Vehicle Routing Problem with Time Windows. Hybrid Heuristics for the Vehicle Routing Problem with Time WindowsWorking Paper, Department of Quantitative Methods, University of Tulsa, OK, USA. (1993).

[72] Chiang, W.C., Russell, R.A.: A reactive tabu search metaheuristic for the vehicle routing problem with time windows. INFORMS Journal on Computing 9, (1997) pp. 417–430.

[73] Cordeau, J.F., Laporte, G., Mercier, A.: A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society, 52, (2001) pp. 928–936.

[74] Taillard, E.D., Gambardella, L.M.: Adaptive Memories for the Quadratic Assignment Problem. Technical Report IDSIA-87-97, IDSIA, Lugano, Switzerland, (1997) pp. 170–186.

[75] Gambardella, L.M., Dorigo, M.: Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem. Twelfth International Conference on Machine Learning, A. Prieditis and S. Russell (Eds.), Morgan Kaufmann (1995) pp. 252–260.

[76] Gambardella, L.M., Taillard, E., Dorigo, M.: Ant Colonies for the Quadratic Assignment Problem. Journal of the Operational Research Society, 50 (1999) pp. 167–176. .

[77] Caseau, Y., Laburthe, F.: Heuristics for large constrained vehicle routing problems. Journal of Heuristics 5:3, (1999) pp. 281–303.

[78] Potvin, J.Y., Robillard, C.: Clustering for vehicle routing with a competitive neural network. Neurocomputing 8:2, (1995) pp. 125–139.

[79] Prosser, P., Shaw, P.: Study of Greedy Search with Multiple Improvement Heuristics for Vehicle Routing Problems. Glasgow: University of Strathclyde, (1996) , Available at http://www.apes.cs.strath.ac.uk/1996.html.

[80] Homberger, J.: Verteilt-parallele Metaheuristiken zur Tourenplanung. Gaber, Wiesbaden, (2000) .

[81] Mester, D., Bräysy, O., Dullaert, W.: A Multi-parametric Evolution Strategies Algorithm for Vehicle Routing Problems. Expert Systems with Applications, Volume 32, Issue 2, (2007) pp. 508–517.

[82] Debudaj-Grabysz, A., Czech, Z.J.: A Concurrent Implementation of Simulated Annealing and Its Application to the VRPTW Optimization Problem. The International Series in Engineering and Computer Science, Distributed and Parallel Systems, Springer Netherlands, Volume 777, Part VI, (2005) pp. 201–209 .

[83] Czech, Z.J., Czarnas, P.: A Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows. Proc. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, Canary Islands, Spain, (2002) pp. 376–383.

[84] Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., Dueck, G.: Record Breaking Optimization Results Using the Ruin and Recreate Principle. Journal of Computational Physics, Vol. 159, (2000) pp. 139–171.

[85] Shaw, P.: A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems. Technical report, APES group, Department of Computer Science, University of Strathclyde, Glasgow, Scotland, (1997) , Available at http://citeseer.ist.psu.edu/shaw97new.html.

[86] Pisinger, D., Ropke, S.: A general heuristic for vehicle routing problems. Computers and Operations Research, 34, (2007) pp. 2403–2435.

[87] OPT, S. SINTEF Applied Mathematics - Department of Optimisation, Technical Report in progress, (2007) .