Automated, Explainable Rule Extraction from MAP-Elites archives

No Author Given

No Institute Given

Abstract. Quality-diversity (QD) algorithms that return a large archive of elite solutions to a problem provide insights into how high-performing solutions are distributed throughout a feature-space defined by a user — they are often described as *illuminating* the feature-space, providing a qualitative illustration of relationships between features and objective quality. However, if there are 1000s of solutions in an archive, extracting a *succinct* set of rules that capture these relationships in a quantitative manner (i.e. as a set of rules) is challenging. We propose two methods for the automated generation of rules from data contained in an archive; the first uses Genetic Programming and the second, a rule-induction method known as CN2. Rules are generated from large archives of data produced by running MAP-Elites on an urban logistics problem. A quantitative and qualitative evaluation that includes the end-user demonstrate that the rules are capable of fitting the data, but also highlights some mismatches between the model used by the optimiser and that assumed by the user.

Keywords: Real-World · Logistics · Optimisation.

1 Introduction

When solving real-world problems, the role of the automated solver is often to support a domain expert, such as a logistics planner, in finding a solution that meets their specific requirements. Typically, the domain expert specifies the problem instance and "owns" the final solution. It is often beneficial if the domain expert has a choice of solutions to chose from: this allows the expert to use their judgement when comparing solution characteristics and trade-offs. Quality-Diversity (QD) algorithms [13] — methods that generate an archive of high-performing but diverse solutions in a user-defined feature space — support the expert user when making these decisions by providing a large range of solutions.

However, often the size of the archives returned by QD methods can be daunting to user, with hundreds or even thousands of solutions within them. It would therefore be beneficial to be able to summarise the archive in a succinct set of policies derived from the solutions contained within the archive, to capture relationships between problem characteristics and decision variables of interest. The derived policies should be congruent with beliefs and expert domain-knowledge of the user, thus increasing their trust in the algorithm outputs. In situations

where the policies do not align with the user beliefs then they should provide a rationale explanation to the user.

In this paper we investigate methods for automatically generating policies to explain the data generated by running a QD algorithm (MAP-Elites) on an optimisation problem in the multi-objective urban logistics domain [17]. A typical run of the technique generates 1000s of high-quality solutions which are diverse in multiple dimensions defined by the user — specifically in this case, we define 9 dimensions that include for example, the emissions associated with a solution, the time taken to route and the overall cost.

We compare two popular rule-generation methods (Genetic Programming [] and CN2 [])in order to answer the following research questions:

1. *Which method produces most accurate (and therefore trustworthy) policies in terms of describing the data?*
2. *Which method produces the least complex policies and how does this complexity relate to accuracy/trustworthiness??*
3. *From a qualitative perspective, do the policies match user beliefs and can they highlight mismatches between the users' model and the implemented model?*

A domain expert is consulted to state their expectations with regards to the factors that might influence solutions in order to evaluate the final question.

The remainder of this paper is organised as follows, section 2 provides a review of relevant previous work in order to place this work into context. The urban logistics problem and the MAP-Elites algorithm applied to it is described in section 3. The two methodologies for rule extraction (CN2 and GP) are discussed in section 4. The results obtained (both quantitative and qualitative) are presented in section 5 and finally, section 6 provides our responses to the research questions.

2 Previous Work

The Multi-dimensional Archive of Phenotypic Elites (MAP-Elites) was first introduced by Mouret *et al* [13] and provides a mechanism for illuminating search spaces by evolving an archive of high-performing solutions mapped onto solution characteristics defined by the user. The algorithm aims to fill a multi-dimensional space defined by the features of interest and discretised into cells with the best performing solution for each cell. To date, the majority of applications of illumination algorithms have been to *design* problems [13, 18] or within Evolutionary Robotics [5].

However, more recently they have been deployed in combinatorial optimisation [**?**] and constrained-optimisation [8]. Other applications in discrete feature-spaces include optimising the hyper-parameters describing the architecture of deep neural networks []. With respect to the former, in our own previous work we have shown that MAP-Elites can be used to generate diverse solutions to a multi-objective urban logistics problem [17].

Despite the success of MAP-Elites, few of the reported applications consider the question of guiding the user towards selecting an appropriate solution from the large numbers of solutions generated, while to the best of our knowledge, none of them address the question of how to summarise or explain the data generated. An exception to the former point is recent work from [11] who develop an interactive version of MAP-Elites in which a user guides the search towards areas of interest, while an automated search procedure based on Bayesian optimisation is used by [5] to select an appropriate behaviour for a given situation in a robot application.

Generating rules from data can be tackled in multiple ways. Genetic programming (GP) has been frequently used to generate classifiers [7]. Tokinaga *et al* [16] used GP to extract rules from a neural network to provide explanatory classifications of data. Other work used GP to generate a set of comprehensible decision rules to identify cases of bankruptcy [10]. Decision-trees have been used for classification in a wide range of applications, with recent proposals for developing meta decision trees for explainable recommendation systems [14]. Based on this, we select GP and decision trees as appropriate candidates to generate rules. We compare their performance to a well known rule-induction algorithm CN2 [4] as a baseline. CN2 uses entropy as a fitness measure to find a set of $IF...THEN...$ rules that classify examples. CN2 is designed to find a set of rules that cover the most examples in a training set with the correct classification. CN2 constructs rules, using entropy as a search heuristic. CN2 is based on constructing rules, whereas GP commences from a random starting point and evolves its tree-based rules according to their fitness. The stochastic nature of the recombination and mutation operators within GP is a direct contract to the non-stochastic nature of CN2. CN2 was initially utilised in a medical domain [4] [3], but it has also been applied to domains ranging from bio-diversity [15] to traffic accident analysis [12].

3 Problem Domain

3.1 The Micro Depot Routing Problem

The Micro Depot Routing Problem (MDRP) was previously described in [17] and concerns the optimal deployment of couriers for city-centre deliveries. Traditional courier deliveries have been made using vans, but this contributes to pollution and congestion. These impacts may be reduced by making deliveries using walking couriers, cycle couriers and electric vehicle couriers within city centers. As such couriers have a small capacity and limited range they are operated from micro depots (MDs). MDs are located near the city centre, but in locations which may be serviced by larger vehicles. Deliveries can be stored in depots prior to couriers making the final deliveries. We consider problems in which there are a fixed number of pre-identified MD locations: the goal is to find a solution that specifies whether an item should be delivered by a courier (and if so, from what MD, and what type of courier) and which deliveries should

still be made by the large vehicle. Solutions to the problem have the following characteristics:

- Emissions - the emissions produced by the travel activities associated withe the solution
- Time - the time between leaving the central depot and the last delivery being made
- Running Cost - Costs that can be apportioned to vehicle running costs and wages
- Fixed Cost - Vehicle purchase and other fixed costs
- byMD - The % of deliveries that travel via a micro-depot
- MDs in use - The number of micro-depots in use
- WalkDels - The number of deliveries carried out using walking couriers
- CycleDels - The number of deliveries carried out using cycle couriers
- EvanDels - The number of deliveries carried out using Electric Van couriers

3.2 Experimental Method

We use an representation and operators that are fully described by the authors in [17]. The representation uses a *grand tour* which represents the route to be taken using the supply vehicle to visit all customers, this route is constructed using the nearest-neighbour heuristic. This default solution is modified by having some deliveries made by a courier operating from an MD. Within our representation, a chromosome contains instructions to transfer groups of deliveries to a courier:

$$< TourPoint >, < Length >, < MD >, < Courier >$$

An example gene might be : $5, 3, D1, WALK$ which would remove the customers at positions 5,6 and 7 in the grand tour and have them delivered by a walking courier based at depot D1 (the 3 items removed from the grand tour are replaced by a single visit to D1). Mutation operators can create new genes, delete old genes or randomly alter the properties of a selected gene. The recombination operator creates new chromosomes by randomly selecting genes from two parents.

The five problem instances used within this paper are based on the city of Frankfurt, Germany and are described in detail in [17]. The underlying street graph is based upon Open StreetMap [9] data and delivery data is based upon information supplied by commercial CEP companies.

We use the MAP-Elites implementation written in Python by [13] and described in detail in our previous work [17]. The feature-space is defined in 9 dimensions, with each dimension divided into 5 bins, giving an archive size of 5^9. The algorithm was executed with a mutation probability of 0.2 and an evaluation budget of 500,000 evaluations.

At the end of a run of MAP-Elites, each 9-dimensional solution generated in the archive can represented using parallel coordinates, for example using the ElVis tool[1], visualised in figure 1. Each solution is represented by a polyline that

[1] https://commute.napier.ac.uk

intersects the 9 axis. On each axis the solutions are normalised on a scale of 1-5. Although figure 1 only shows a relatively small number of solutions, it is clear that even with few solutions it can be difficult for the user to discern trends between the problem characteristics.

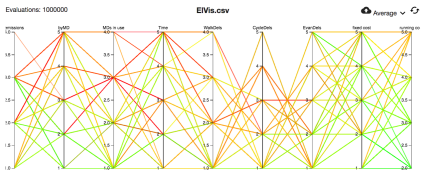


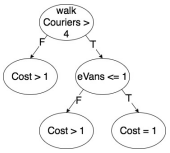
Fig. 1. A sample archive as generated by MAP-Elites, visualised using the ElVis tool. The colour of each polyline represents the overall fitness of that solution, green being the lowest and red the highest.

4 Methodologies for extracting policies

From the perspective of an expert user (e.g. a logistics planner), four of the nine features can be considered as *outcome* variables, i.e. the user would like to know which of the other features lead to the outcome variable being classified as high or low. These four outcome variables are Emissions, Time , Running Cost and Fixed Cost; a value of low is preferred for each. A *policy*, in this context, describes a rule that links a desired outcome (e.g. low emissions) to a set of the remaining solution attributes. In plain English a policy might be:
To achieve low costs, make maximum use of walking couriers and make minimal use of electric vans
This policy could be expressed as an $IF..THEN...$ rule by CN2 (where 1 indicates low cost and 0 would indicate high cost) :

IF walkCouriers > 4 AND eVans <= 1 THEN costs =1

Or as a GP tree:



Policies that cover a subset of the solutions in the archive are useful to the end-user in summarising the data, and reflect the assumptions in the underlying model. The goal of the rule-extraction phase is therefore to extract a succinct set of policies that explain how the 4 outcome variables are influenced by the

remaining 5 characteristics that describe a solution: byMD (delivered from a micro-depot); number of MDs used; number of deliveries by (walking, cycling) couriers and by electric van.

The processes used in extracting the policies as described in this section and the results as described in section 5 are implemented within a Jupyter notebook which can be downloaded along with the associated Java files for FlexGP and the checker program[2].

4.1 Data Preparation and Analysis

(see table 1) in order to take account of the stochastic nature of the algorithm. Each of these 10 runs produces its own archive, which are combined into a single combined archive, retaining the single best solution found per bin. The quantity of solutions contained within these archives is given in table 1. Table 2 shows the attributes of the problem instances as well as the outcome variables. All of the problem attributes have the range 1-5, as the MAP-Elites algorithm has 5 "buckets" per attribute. The outcome variables are represented as binary where a '1' represents a low value and 0 corresponds to bins 2-5 that all are interpreted as high.

Problem	**Combined Archive Size**
f1	1137
f2	113
f6	1523
f8	573
fCity	514

Table 1. Combined Archive Sizes

Outcome Variable(binary low/high)	**Attribute**(1-5)
emissions	byMD
time	MDsInUse
running cost	WalkDels
fixed cost	CycleDels
	EVanDels

Table 2. The solution attributes of the micro depot problem

Our interest is in finding policies that are indicators of low values of emissions, time, fixed costs and running costs. The values of the other attributes can be directly inferred from the genotype. In order to find policies for each of

[2] these will be made available if paper is published

these outcome variables we create 4 datasets (see table 3), one for each of the outcome variables. Each data set is created by combining the outcome variable (emissions,time,fixedCost or runningCost) and the five attributes (see table 2).

The data sets from the problem instances are combined, so that each of the 4 data sets covers all of the problem instances (see table 3). When combined the data sets are unbalanced with an unequal number of low and high values of the outcome variable. We balance the data by employing *random under-sampling* which removes a random selection of examples from the majority class leaving a balanced data set.

Problem	**Unbalanced (low / high)**	**Balanced**
Emissions	1609 / 2251	3219
Time	535 / 3325	1070
FixedCost	166 / 3694	332
RunningCost	35 / 3825	70

Table 3. The sizes of the data sets before and after balancing.

4.2 Extraction Methodologies

We use two contrasting techniques to extract policies, CN2 and Genetic Programming. In both cases we treat the entire data set as the training data. It is important to note that we do not need to apply the generated rules to *unseen* instances; the goal is simply to describe the existing data. Hence we use all the data as training data to generate rules and report results on the full dataset.

CN2 The CN2 rule induction algorithm [**?**] is designed to construct simple rules in the $IF < condition > THEN < classify >$ format, which can then be used as the basis for policies. The CN2 algorithm is applied to each of the 4 data sets to find policies of the $IF...THEN...$ format that will attempt to predict the outcome variable, as either low (1) or not low (greater then 1). Each of these rules will *cover* a number of rows within the training data. A row is *covered* by a rule if the rule can be evaluated against the values within the row and the rule output of the rule aligns with the decision variable with the row. The CN2 algorithm as a *minimum coverage* parameter, any rule that does not cover at least the specified number of rows is disregarded, which reduces the overall number of rules found. In this paper we use the CN2 algorithm as implemented within the Orange [6] data analysis framework.

Genetic Programming Genetic Programming (GP) refers to the evolution of programs or expressions, commonly using a tree representation. As noted in section 2, the use of GP to evolve classifier trees is very common. Many platforms

provide implementations: here we utilise the RuleTree learner provided as part of the FlexGP Platform [1] [3].

The flexGP learner uses a two stage approach

- **Stage 1** A set of conditions are constructed which divides the attributes within the problem domain into intervals. These form conditions which are ultimately used as terminals on the evolved trees. An example of a condition used by flexGp would be:
  X4 in [1.0 ; 2.3]
  Which specifies that attribute $X4$ must be between 1.0 and 2.3

- **Stage 2** Genetic Programming is used to evolve a Pareto front of tree based expressions.

During stage two FlexGP uses a bi-objective approach to evaluate solutions and place them within a Pareto front. The objectives used are accuracy and *Subtree Complexity* as defined in [2].

The GP expressions make use of AND, OR and NOT operators which are applied to conditions defined in stage 1. An example of an expression produced by FlexGP would be :

(and (or (and C9 C3) C11) C2)

Where $C9$ etc are expressions created during stage 1. Substituting the full expressions gives :

(((2.3<=X4<=5.99 and 1.0<=X2<=1.68)
or 2.35<=X5<=5.99) and 3.75<=X1<=5.99)

We run flexGP for a limit of 5 minutes, all other parameters are as defined in [1].

4.3 Evaluating the policies

We undertake quantitative and qualitative and evaluations of the policies discovered. Both the FlexGP and Orange libraries calculate a range of metrics in respect of their methods. In order to ensure a simple and fair comparison, we construct an independent checker. The checker takes policies in either format (CN2 or GP) counts the True Positive and False Positive coverage across the data sets (see table 3). The checker calculates the precision of each rule as follows:

$$precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Precision gives an indication as to how much trust may be placed in a policy. A high precision denotes that policy has a higher number of true positives within the cases that it covers. The checker was written in Java and the source code made be downloaded along with the Jupyter Notebook files and data.

[3] http://flexgp.github.io/gp-learners/ruletree.html

5 Results

Table 5 shows the best policies derived using both methods of generation, ordered by precision . The policies are compared using the independent checker which determines the coverage (true positives plus false positives) of each policy on the original balanced data set and calculates their precision as described in section 4.3. We select the three most precise policies generated by each method. It will be noticed that the GP derived policies have a consistently higher precision than their CN2 counterparts.

5.1 Policies produced using CN2

The policies produced using CN2, mostly have lower precision than those produced using GP which lessens confidence in the CN2 policies.

Emissions If we examine the three emissions policies produced by CN2, we note that they all propose that large amounts of deliveries should be made via a Micro Depot (byMD ¿= 4). Policy E2 suggests that emissions are reduced with the maximal use of eVan based couriers.

Time The three policies for time all promote the use of eVan couriers. What is significant is that none of the policies mention walking couriers - not surprising as they are slowest form of delivery and have the least capacity.

Running Cost The establishment of policies to reduce running costs was the biggest challenge of this work. In the case of CN2, only 2 policies were returned. The two policies returned have very low precision, combined with low overall coverage ($truePositive + falsePositive$), suggesting that they are not of significant value.

Fixed Cost A factor common across all three policies for lower fixed costs is byMD, suggesting that more limited use of micro depots will lower fixed costs. Of the three policies only F2 suggests that limited use of electric vans will lower fixed costs, despite electric vans having a higher fixed cost than any other form of courier..

5.2 Policies produced using GP

The policies produced using GP have far higher precision and TruePositive scores than those produced by CN2, suggesting that a greater degree of confidence may be placed in them.

Emissions Of the three emissions policies produced, they exhibit the following common factors:

- CycleDels 2.3 - 6.0
- eVanDels 2.35 - 6

Both of the above encourage the use of cycle and electric van based couriers, which are both low emission means of deliveries. Only two of the policies mention the number of micro depots to be used suggesting that the number of MDs used has less influence on emissions.

Time Three common factors run through the Time rules (T4-T6):

- WalkDels in [1.0 - 1.32]
- EVanDels in [2.5 - 6.0]
- MDSinUse in [1.64 - 4.96]

The policies suggest that time can be reduced by minimising the use of walking couriers (they are the slowest form of courier and have the lowest capacity) and encouraging the use of electric vans which are quicker than walking or cycling.

Running Cost The running cost policies exhibit consistently high scores (0.78,0.76,0.75), suggesting that confidence may be placed in them, but all three of the have a low coverage.

Common factors across all three policies are:

- Low use of each courier mode is suggested/
- byMD in [1.0 - 3.65]) - have between 0 and half of deliveries made via a micro depot
- MDSinUse in [1.0 - 1.04] (policies R3 and R5)- Use very few micro depots

These policies suggest that to minimise running costs very selective use of micro depots should be made. These policies should be viewed in the context that no policies were found that reduced running costs to 1, the only policies that could be found reduced them to 2. The overall conclusion to be drawn from this is that the use of micro depots does not lend itself to a reduction in running costs.

Fixed Cost A consistently high precision (1) suggest that we can place a high degree of confidence in these policies, but the low coverage of these polices should be noted.

Common factors across all three policies include:

- EVanDels in [1.0 - 1.05] in F5 and F6, suggests reducing the use of electric vehicles will reduce the fixed cost of the solution.

– MDSinUse in [1.0 - 1.32], byMD in [1.0 - 3.5], suggesting lower numbers of MDs to be used and no more than half of the deliveries to be made via MDs

The policies show that the fixed costs are reduced through selective use of MDs and minimal use of electric vehicles.

5.3 Matching the expectations of the domain expert

In order to provide a qualitative evaluation of the policies found, the authors had a domain expert state their expectations and beliefs with regards to the 4 objectives. The expectations reflect the measures that the expert would associate with reducing these objectives. Table 4 shows the expectations and how they are satisfied (or not) by the policies.

This qualitative evaluation raises a number issues of interest. Firstly the domain expert may well have a more detailed mental model of the domain than is represented in the software model. Expectation Emissions 3 is a case in point; the expert has an expectation that use of electric vehicle couriers must be reduced when reducing emissions, owing to the environmental impact of battery production. This knowledge of the impact of battery production is not included in the model and so this expectation will never be explicitly addressed (it might be satisfied, but that will be due to the solver optimising against other criterion). Some expectations such as Emissons 1,Fixed Cost 1 Time 1,Running cost 1 and Running cost 2 are satisfied by a range of policies. Where a rule satisfies multiple expectations, the rule tends to be longer and more complex.

A final observation on table 4 suggests that the first expectation (E1,F1,T1 and R1) is more likely to be covered by the policies. The expectations are numbered in the order they were supplied by the expert: it may be that there is a form of precedence within the observations in that the earlier observations (e.g. R1) are more "obvious" than the later observations (e.g. R3).

6 Conclusions and future work

Based on the results described above, we provide answers to the questions posed in Section 1:

Which method produces most accurate (and therefore trustworthy) policies in terms of describing the data? It is clear from table 5 that the GP derived policies exhibit a far greater coverage and precision. This suggests that he GP derived policies are more representative of the relations that are implicit with the model that is being optimised.

Which method produces the least complex policies and how does this complexity relate to accuracy/trustworthiness? The styles of policies

	Beliefs	Policy	Satsfies
Emissions	1 low number of trucks delivering MD	E1	1
	2 low number of MD	E2	
	3 low number of EV˜ (battery's production causes emissions)	E3	1,2
		E4	1
		E5	1
		E6	1
Time	1 low number of slowest staff (walking couriers)	T1	2
	2 depending on delivery area low˜ number of EV or cycle couriers˜ (in a dense area a cycle courier is faster)	T2	2
	3 high number of parcel shops˜ (locations where people can pickup˜ their parcels instead of home delivery)	T3	
		T4	1,2
		T5	1
		T6	1
Running Cost	1 low number of MD	R1	1,2
	2 low number of expensive staff	R2	
	3 low number of expensive vehicles,˜ cheap vehicles to lower running costs,˜	R3	1,2,3
		R4	1,2,3
		R5	1,2,3
Fixed Cost	1 low number of MD	F1	
	2 low number of EV	F2	2
	3 low number of cycle couriers	F3	1
		F4	1
		F5	1,2
		F6	1,2

Table 4. The expert users' beliefs' are are divided into 4 categories to align with the decision variables. Each policy may then be evaluated to see which (if any) beliefs that it satisfies. The Satisfies column notes the beliefs which are present within the policy(e.g. policy E3 satisfies beliefs 1 and 2.)

produced by FlexGP and the CN2 algorithm differ fundamentally. The policies produced by CN2 are of the $IF...THEN..$ format, which many individuals (especially those without a computing science background) will find easier to interpret than the tree-based policies produced by FlexGP. If we consider the FlexGP derived policies and examine scores in relation to operators we note that higher scores are obtained by those policies that contain the most operators. We conclude, that in this study at least, the more complex FlexGP policies are the most trustworthy and the more operators contained within the tree, the more trust can be placed in that policy.

From a qualitative perspective, do the policies match user beliefs and can they highlight mismatches between the users' model and the implemented model? Table 4 shows that the initial expectations (E1, F1, T1 etc) are covered by most methods, the differences occur when considering the later expectations (F2, R3 etc).

6.1 Recommendations and Future Work

In conclusion, the automated extraction of underlying policies from MAP-Elites is a worthwhile activity. Comparing the extracted policies with the expectations of an expert user can highlight differences between the model assumed by the expert and the model used within optimisation. This provides an opportunity to update the optimisation model in order to move it closer to the users' expectation. In the case examined here, rules E3 and F3 might be incorporated into an updated model. An area for future work might be the automatic updating a weights and rules within the model to move the result closer to the users' expectations. Given that GP and CN2 perform well on differing areas, it may be worth using both of these methods in order to create a wide ranging set of rules, but conflicting rules and duplication would have to be managed.

Table 5. The most effective (as determined by the checker) derived from the Elite solutions within the archives.

Characteristic	Policy	Policy ID	Source	Precision	True Positives	False Positives
Emissions	IF byMD=4.0 AND MDs in use=3.0 THEN emissions=1	E1	CN2	0.79	48	13
	IF byMD=5.0 AND EvanDels=5.0 THEN emissions=1	E2	CN2	0.76	40	16
	IF byMD>=5.0 AND MDs in use<=2.0 AND MDs in use>=2.0 AND EvanDels<=4.0 AND EvanDels>=4.0 THEN emissions=1	E3	CN2	0.67	269	132
	(and EVanDels in [2.35 - 6.0] (and CycleDels in [2.3 - 6.0] MDSinUse in [1.68 - 5.0]))	E4	GP	**0.80**	16	4
	(and EVanDels in [2.35 - 6.0] CycleDels in [2.3 - 6.0])	E5	GP	0.72	48	19
	(and (or EVanDels in [2.35 - 6.0] WalkDels in [1.64 - 4.24]) (and MDSinUse in [1.68 - 5.0] CycleDels in [2.3 - 6.0]))	E6	GP	0.71	52	21
Time	IF EvanDels>=3.0 AND CycleDels>=2.0 AND EvanDels>=4.0 THEN Time=1	T1	CN2	0.64	96	55
	IF byMD=4.0 AND EvanDels=4.0 THEN Time=1	T2	CN2	0.64	96	55
	IF EvanDels>=2.0 AND byMD<=4.0 AND EvanDels>=4.0 THEN Time=1	T3	CN2	0.64	96	55
	(and (and CycleDels in [1.0 - 2.35] WalkDels in [1.0 - 1.32]) (and EVanDels in [2.5 - 6.0] MDSinUse in [1.64 - 4.96]))	T4	GP	**0.74**	65	23
	(and (and byMD in [1.2 - 4.45] MDSinUse in [1.64 - 4.96]) (and WalkDels in [1.0 - 1.32] EVanDels in [2.5 - 6.0]))	T5	GP	0.73	33	12
	(and EVanDels in [2.5 - 6.0] (and MDSinUse in [1.64 - 4.96] WalkDels in [1.0 - 1.32]))	T6	GP	0.72	71	27
Running Cost	IF byMD<=4.0 AND CycleDels<=3.0 AND byMD<=3.0 AND CycleDels<=2.0 THEN running cost=1	R1	CN2	0.55	29	24
	IF CycleDels=2.0 THEN running cost=1	R2	CN2	0.50	18	18
	(and (not (and CycleDels in [1.0 - 2.8] (and byMD in [1.0 - 3.65] EVanDels in [1.0 - 2.6]))) (and (or byMD in [1.0 - 3.65] (and CycleDels in [1.0 - 2.8] EVanDels in [1.0 - 2.6])) (and WalkDels in [1.0 - 1.32] MDSinUse in [1.0 - 1.04])))	R3	GP	**0.78**	7	2
	(and (or (and CycleDels in [1.0 - 2.8] EVanDels in [1.0 - 2.6]) byMD in [1.0 - 3.65]) (and WalkDels in [1.0 - 1.32] MDSinUse in [1.0 - 1.04]))	R4	GP	0.76	28	9
	(and (or byMD in [1.0 - 3.65] CycleDels in [1.0 - 2.8]) (and EVanDels in [1.0 - 2.6] (and WalkDels in [1.0 - 1.32] MDSinUse in [1.0 - 1.04])))	R5	GP	0.75	27	9
Fixed Cost	IF WalkDels<=2.0 AND byMD<=4.0 AND CycleDels>=2.0 AND CycleDels>=4.0 THEN fixed cost=1	F1	CN2	0.64	18	10
	IF byMD=3.0 AND EvanDels=2.0 THEN fixed cost=1	F2	CN2	0.59	166	115
	IF byMD<=4.0 AND CycleDels>=2.0 AND MDs in use<=2.0 THEN fixed cost=1	F3	CN2	0.50	166	165
	(and byMD in [1.0 - 3.5] (and CycleDels in [2.9 - 6.0] MDSinUse in [1.0 - 1.32]))	F4	GP	**1.00**	17	0
	(and (and EVanDels in [1.0 - 1.05] byMD in [1.0 - 3.5]) (or (and CycleDels in [2.9 - 6.0] MDSinUse in [1.0 - 1.32]) (and WalkDels in [1.68 - 4.76] MDSinUse in [1.32 - 5.0])))	F5	GP	**1.00**	19	0
	(and (and EVanDels in [1.0 - 1.05] byMD in [1.0 - 3.5]) (and WalkDels in [1.68 - 4.76] MDSinUse in [1.32 - 5.0]))	F6	GP	**1.00**	2	0

References

1. Arnaldo, I., Veeramachaneni, K., Song, A., O'Reilly, U.: Bring your own learner: A cloud-based, data-parallel commons for machine learning. IEEE Computational Intelligence Magazine **10**(1), 20–32 (2015). https://doi.org/10.1109/MCI.2014.2369892
2. Arnaldo, I., Krawiec, K., O'Reilly, U.M.: Multiple regression genetic programming. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation. pp. 879–886 (2014)
3. Clark, P., Boswell, R.: Rule induction with cn2: Some recent improvements. Proc. European Working Session on Learning (07 1998). https://doi.org/10.1007/BFb0017011
4. Clark, P., Niblett, T.: The cn2 induction algorithm. Machine learning **3**(4), 261–283 (1989)
5. Cully, A., Clune, J., Tarapore, D., Mouret, J.B.: Robots that can adapt like animals. Nature **521**(7553), 503 (2015)
6. Demšar, J., Curk, T., Erjavec, A., Črt Gorup, Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., Štajdohar, M., Umek, L., Žagar, L., Žbontar, J., Žitnik, M., Zupan, B.: Orange: Data mining toolbox in python. Journal of Machine Learning Research **14**, 2349–2353 (2013), http://jmlr.org/papers/v14/demsar13a.html
7. Espejo, P.G., Ventura, S., Herrera, F.: A survey on the application of genetic programming to classification. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **40**(2), 121–144 (2009)
8. Fioravanzo, S., Iacca, G.: Evaluating map-elites on constrained optimization problems. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. pp. 253–254. ACM (2019)
9. Foundation, O.: http://www.openstreetmap.org (2014), http://www.openstreetmap.org
10. Garcia-Almanza, A.L., Alexandrova-Kabadjova, B., Martinez-Jaramillo, S.: Understanding bank failure: A close examination of rules created by genetic programming. In: 2010 IEEE Electronics, Robotics and Automotive Mechanics Conference. pp. 34–39. IEEE (2010)
11. Hagg, A., Asteroth, A., Bäck, T.: Prototype discovery using quality-diversity. In: International Conference on Parallel Problem Solving from Nature. pp. 500–511. Springer (2018)
12. Lavrac, N., Kavsek, B., Flach, P.A., Todorovski, L.: Subgroup discovery with cn2-sd. J. Mach. Learn. Res. **5**, 153–188 (2004)
13. Mouret, J., Clune, J.: Illuminating search spaces by mapping elites. CoRR (2015)
14. Shulman, E., Wolf, L.: Meta decision trees for explainable recommendation systems. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society. pp. 365–371 (2020)
15. Swe, S.M., Sett, K.M.: Approaching Rules Induction CN2 Algorithm in Categorizing of Biodiversity. International Journal of Trend in Scientific Research and Development **3**(4), 1581–1584 (Jun 2019). https://doi.org/10.5281/zenodo.3591374, https://doi.org/10.5281/zenodo.3591374
16. Tokinaga, S., Lu, J., Ikeda, Y.: Neural network rule extraction by using the genetic programming and its applications to explanatory classifications. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **88**(10), 2627–2635 (2005)

17. Urquhart, N., Höhl, S., Hart, E.: An illumination algorithm approach to solving the micro-depot routing problem. Proceedings of the Genetic and Evolutionary Computation Conference (2019). https://doi.org/10.1145/3321707.3321767
18. Vassiliades, V., Chatzilygeroudis, K., Mouret, J.B.: Using centroidal voronoi tessellations to scale up the multi-dimensional archive of phenotypic elites algorithm pp. 1–1 (08 2017)