Artificial Immune System driven evolution in Swarm Chemistry

Nicola Capodieci
Università di Modena e Reggio Emilia
Modena, Italy

Emma Hart
Edinburgh Napier University
Edinburgh, UK

Giacomo Cabri
Università di Modena e Reggio Emilia
Modena, Italy

***Abstract*—Morphogenetic engineering represents an interesting field in which models, frameworks and algorithms can be tested in order to study how self-* properties and emergent behaviours can arise in potentially complex and distributed systems. In this field, the morphogenetic model we will refer to is swarm chemistry, since a well known challenge in this dynamical process concerns discovering mechanisms for providing evolution within coalescing systems of particles. These systems consist in sets of moving particles able to self-organise in order to create shapes or geometrical formations that provide robustness towards external perturbations. We present a novel mechanism for providing evolutionary features in swarm chemistry that takes inspiration from artificial immune system literature, more specifically regarding idiotypic networks. Starting from a restricted set of chemical recipes, we show that the system evolves to new states, using an autonomous method of detecting new shapes and behaviours free from any human interaction.**

I. Introduction

In [1], Doursat et al. established a new field of research called *morphogenetic engineering* that aims to bridge the gap between emergent self-architectural structures that are typical of bio-inspired approaches and the programmability of more traditional systems engineering approaches. The process of filling this gap has evident ties with fostering the development of self-* properties in generic complex systems. On the one hand, morphogenetic engineering provides an ideal test bed for validating design methodologies and implemented algorithms that aim to foster the autonomic features of these system [2]. From another perspective, studying evolution in morphogenetic engineering provides a valuable instrument to better understand the emergent behaviours that characterize the self-organizing structures produced within this very new research field. In this paper we address a part of morphogenetic engineering known as *coalescing*, defined in [1] as "a great number of mobile agents flock and make together dense clusters, whose contours adopt certain shapes". Sayama [3] introduces the concept of *swarm chemistry*: a tuple of kinetic parameters is assigned to a swarm of particles that moves as described by the well known flocking algorithm as firstly introduced by Reynolds [4]. The kinetic parameters (called *recipes*) regulate the sensing radius, velocities and other interaction forces among the components of the swarm. One of the key discoveries of Sayama was that *mixing* swarms that feature different recipes leads to the emergence of robust geometries.

As described in Section II, evolution in swarm chemistry has already been introduced in previous literature, but it still remains a topic that deserves deeper discussion, especially with respect to the evolutionary mechanisms involved and with regard to how to design systems are able to self-evaluate whether the chemical swarm manages to create new shapes or to show new behaviours. The element of novelty of this paper is to present an Artificial Immune System (AIS) inspired algorithm that exploits an *ad-hoc* idiotypic network [5] for controlling the evolution of single particles composing the swarm in an artificial chemistry environment: the emergent property of immune networks of exhibiting cognition [6] proved to be extremely useful for implementing a Self-Organising system that was able to discover new shapes and new behaviours using the same tool described in [3]. Starting from a restricted initial set of known recipes, our algorithm combines and modifies recipes to create shapes that were not present in the initial sets. Furthermore, it is able to maintain diversity by autonomously detecting whether new recipes are too similar to the previously generated recipes. This is achieved by introducing novel methods of describing the *fitness* of a recipe and the *distance* between recipes.

The paper is structured as follows: in Section II a brief overview on the previous attempts to study evolution in swarm chemistry is presented, also the differences with our approach will be detailed in that section. Section III will briefly summarize how swarm chemistry works, while Section IV, will briefly explain the theory behind idiotypic networks. Section V will present all the details of the implemented algorithm of our approach. Experiments details, results and related discussion will be provided in Section VI and Section VII will conclude the paper with final remarks and future work proposals.

II. Related Work

Previous attempts to provide evolutionary features in swarm chemistry or other morphogenetic related systems, fall into four distinct categories. The first category deals with the need to have a human user act as a *judge* in order to provide some sort of fitness measure that can be used to control the magnitude of subsequent mutations involved in the evolutionary process. A less recent example can be found in [7], in which the authors apply a genetic algorithm for designing three dimensional components: after each generation, the eye of the beholder was used to control how these 3D objects mutated over time. Similarly, in the case

of swarm chemistry, the concept of *hyperinteractivity* [8] involves having several users to participate in the evolutionary design process by subjectively selecting and manipulating preferred swarm patterns. Another category deals with excluding human presence by seeking open-ended evolution. In swarm chemistry, this kind of evolutionary algorithm also excludes the need for a fitness function [9] as it discriminates between *active* and *passive* individuals of the swarm. Tuples describing kinetic parameters are transferred from active to passive particles whenever active components collide or fall within the perceptive radius of inactive parts of the swarm. A third category of evaluating the evolutionary properties in coalescing systems can be found in [10], in which the system knows *a priori* the shape it wants to achieve, thus in each time step, the whole swarm manages to measure its fitness as a difference between the currently achieved shape and the objective goal shape. A final category can be classified as *goal-oriented* evolution: let us suppose a set of moving robots that are able to self-assemble and create different formations [11] such that different shapes are suited for performing different tasks (such as overcoming obstacles of the surrounding environment). In this case, the fitness function can be measured according to how well the current shape manages to solve the specified tasks.

The novel approach we present is aimed at focusing on the evolutionary capabilities of a chemical swarm in a decentralized manner, obtained by exploiting the information a single individual of the swarm is able to observe: in this way, we are able to define a fitness function that can be used for controlling the evolution. In our novel algorithm that merges artificial immunology and swarm chemistry, each individual of the swarm is associated with the concept of a virtual *lymphnode* that is able to host a multitude of interconnected *antibodies* able to evolve over the duration of the experiments. According to a fitness function that takes into account only local information, the concentration that characterizes each antibody inside each swarm individual changes over time so to provide the *fittest* immunological response in terms of kinetic parameters to be assigned to the swarm individual. The dynamics of this adaptive process follows a design methodology that we have previously formulated [12], [13] and applied in a robotic swarm foraging scenario [14]. We will show how this immune-inspired approach achieves self-organisation within the swarm, resulting in the creation of new shapes and behaviours, starting from a restricted set of recipes previously found with the hyperinteractive method. Moreover, to further prove the effectiveness of the presented approach we introduce additional methods of analysing the generated recipes, in order to enable the system to self-evaluate its ability to discover new shapes and behaviours without the need of human users.

III. Sayama's Swarm Chemistry

Detailed accounts of swarm chemistry algorithms can be found in the previously cited publications, however in this section we briefly summarise the most relevant aspects, particularly since our algorithm is built from the Java^{TM} implementation of Sayama's swarm chemistry available on line[1]. In the remainder of the article, we use the same terminology adopted by Sayama in his original publications.

In [3], [8], [9] the concept of swarm chemistry is introduced as an artificial chemistry framework in which particles are moving according to a flocking algorithm that is characterised by kinetic parameters assigned to each of the swarm constituent individual (i.e. sensing radius R, normal speed V_n, maximum speed V_m, strength of cohesive force c_1, strength of aligning force c_2, strength of separating force c_3, probability of random steering c_4 and tendency for self-propulsion c_5). Sayama demonstrated that in this kind of coalescing system, a number of interesting morphogenetic properties emerge from the stigmergic interactions among the individuals composing the swarm: for example, when a single tuple of kinetic parameters is assigned to a whole swarm, the homogeneous swarm is able to show dispersal, coherent linear and even oscillatory motion patterns, depending on the parameter set chosen. Much more interesting and yet unpredictable behaviours arise when the swarm is heterogeneous, i.e. when different swarms characterized by different kinetic parameters are confined within the same space. This gives rise to the name *swarm chemistry*, as the process of mixing different *recipes* (as different values of assigned kinetic parameters) together in order to create new robust structures or other notable behaviours. A single recipe r that characterizes a heterogeneous swarm is written in the form:

Recipe r:

$$
\begin{aligned}
&P_1 * (R_0, V_{n_0}, V_{m_0}, c_{1_0}, c_{2_0}, c_{3_0}, c_{4_0}, c_{5_0}) \\
&P_2 * (R_1, V_{n_1}, V_{m_1}, c_{1_1}, c_{2_1}, c_{3_1}, c_{4_1}, c_{5_1}) \\
&\vdots \\
&P_N * (R_N, V_{n_N}, V_{m_N}, c_{1_N}, c_{2_N}, c_{3_N}, c_{4_N}, c_{5_N})
\end{aligned}
$$

Recipe r is composed of N different *types*, with P_i with $i \in [1, ..., N]$ being the number of individuals of the swarm that are currently characterized by the i^{th} type of recipe. The reader is directed to the swarm chemistry website for viewing sample heterogeneous swarms with mixed recipes[1].

IV. Idiotypic networks in artificial immunology

This section provides a basic introduction to the family of algorithms found within the field of artificial immune systems, commonly referred to as *immune* or *idiotypic* networks, in order to provide the reader with all the means for understanding the core methodology that drives the evolutionary capabilities for the presented coalescing system. Artificial Immune Systems (AIS) represent a computational approach for designing intelligent systems and cover a variety of algorithms that have been used in a multitude of different applications [15]. Taking inspiration from biological immunology in order to design

[1]http://bingweb.binghamton.edu/∼sayama/SwarmChemistry/

computational systems, offers great potential for the system engineer [16] due to the fact that the adaptive layer of the biological immune system features a number of characteristics that are often sought by the designer of self-* systems, namely decentralization, scalability, self-non self discrimination and learning capabilities. Immune networks were first proposed by Jerne in [17], where he described an immunological model in which antibodies not only were able to bind with antigens (via epitope-paratope affinity) but also with other antibodies, thus creating a dynamic network in which suppression and stimulation signals are able to vary the concentration of the antibodies over time and the topology of the resultant network. The pattern of concentration of the antibodies characterizes the immunological response — various mathematical models have been proposed to describe how concentration is regulated, using differential equations that calculate the variation of concentration over time as a function of the stimulation and suppression signals that occur within an observed idiotypic network (e.g. [16]). Usually, modelling an idiotypic network for solving a computational problem involves modelling the antibodies as data structures that can be matched against some external variables perceived from the environment and then deploying a corresponding action or combination of actions according to the activated antibodies. Immune networks are shown to provide cognition and memory [5], [6], for instance improving future responses by taking into account whether a previously selected action resulted in a positive or negative feedback, and sharing this information throughout the network in the form of stimulation and/or suppression signals.

V. An AIS-inspired model for evolutionary swarm chemistry

Our approach focuses on individual particles, rather than on the whole swarm. In this way, we are able to evaluate a macroscopic behaviour (such as a shape formation) starting from the microscopic level, i.e. from the individuals. Each individual i of the swarm is associated to a lymphnode L_{c_i} that is able to host a multitude of interconnected antibodies. An antibody Ab_j is represented by a tuple in the form $\langle Action\ A, Expected\ utility\ Ut \rangle$ and is characterized by a concentration level and affinity values regarding all the other antibodies hosted in the same lymphnode (see Fig. 1).

The action field of the antibody tuple is a formal description of a recipe, written in the same form as introduced in Section III; this data structure defines which kinetic parameters should be followed by an individual. The second field of the antibody represents the concept of *expected utility* as a result of a *fitness* function calculated (from the point of view of lymphnode L_{c_i}) according to two observable variables H_o and H_e: we define the first one as *index of local homogeneity* that represents the number of neighbouring individuals that are currently sharing the same kinetic parameters as L_{c_i}. The latter variable is defined as *index of local heterogeneity* and represents the number of different tuples of kinetic parameters currently adopted by the observable neighbourhood. The neighbouring radius is the same as the one specified as the

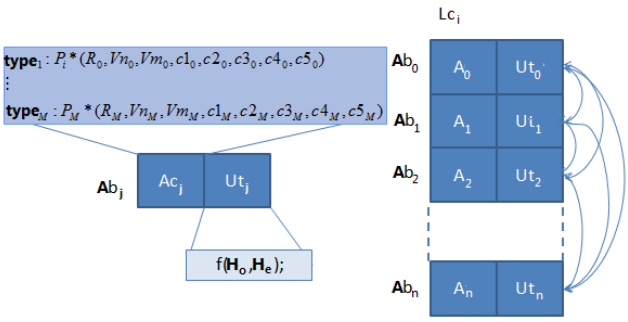

Fig. 1. Antibodies representation inside a single swarm individual modelled as a lymphnode.

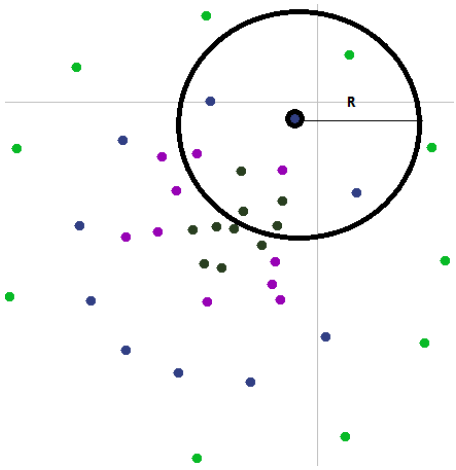

Fig. 2. A black circle is centred in the observed lymphnode-swarm individual. Inside the neighbouring radius, the observed individual is able to perceive two particles that share the same kinetic parameters since different coloured particles indicate individuals that are following different kinetic parameters. At the same time, the same particle can observe four dark green particles, one light green particle and two purple particles. As a result of all of that, the fitness function will be calculated using $H_o = 2$ and $H_e = 3$ as inputs.

first kinetic parameter in the part of the action-recipe that L_{c_i} is currently following (R) and the evaluation of the fitness function occurs periodically after each time interval T_{eval}. The fitness function itself is detailed in the next sub-section, however an example of situation is depicted in Fig. 2, in which we show a screenshot of Sayama's simulation environment with mixed recipes; in that figure we show how H_o and H_e are sensed by a single particle[2].

A. Fitness of a recipe

Intuitively, the purpose of the fitness function is to balance the number of similar individuals (i.e. those particles that share the same kinetic parameters) against the number of individuals with different kinetic parameters observable in the same radius. However, the function should avoid concentrating similar or diverse particles inside a restricted neighbouring radius. In

[2]According to [3], the colour of a particle in the swarm chemistry simulator is a function of parameters c_1, c_2 and c_3

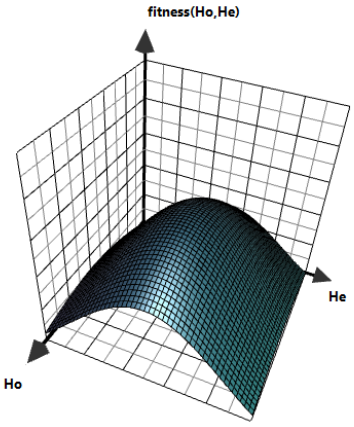

Fig. 3. A graph that shows how fitness values change according to the local point of view of a swarm individual. In this represented case, $N_R = 300$ (maximum population value in Sayama's swarm chemistry framework), $K_0 = 10$

other words, we want a single particle to be physically close to those that shares the same colour, but at the same time we aim at a mix with different coloured particles, and at the same time, avoiding situations in which all the particles are concentrated in a restricted portion of space. We modelled this situation with a pseudo-Gaussian formula whose graph is reported in Fig. 3 and described by eq. 1. H_o and H_e are local homogenous/heterogenous indexes, N_R is the total number of neighbour individuals, K_0 is a constant that is used to move the maximum value of the function according to a specified value of local heterogeneity index.

$$fitness_{(H_o,H_e)} = exp\left(-\left[\left(\frac{H_o - \frac{N_R}{2}}{N_R}\right)^2 + \left(\frac{H_e - \frac{K_0}{2}}{K_0}\right)^2\right]\right) \quad (1)$$

It is important to stress that this fitness function is just one of the many that can be used in this context, however it allows a single particle to evaluate its performance after each time interval without the need for global observers or strong communicative capabilities among the other swarm individuals. Note that this approach differs from that in the original papers in which individuals were unaware of the fact that their neighbouring particles were following different kinetic parameters.

The fitness function plays a key role during the *pre-experimental* phase (detailed later in this section), in connection with initialising the expected utility field of the initial set of antibodies.

B. Action selection

The action field of an antibody tuple is a formal description of a recipe: each recipe can be characterized by more than one *type* (as seen in Section III). However, since a single particle during a single time interval can be characterized by only one tuple of kinetic parameters, we have to define the process of *action selection*. From the point of view of the single particle, selecting an action that features a recipe composed of N types means that the i^{th} tuple of kinetic parameters will have a certain probability p_i to be selected that is dependent on the ratio between the population of the i^{th} type of the recipe and the total population indicated by the recipe stored in the antibody. In eq. 2 we formally describe the process of action selection by indicating how p_i is obtained.

$$p_i = \frac{P_i}{\sum_{j=1}^{N} P_j} \quad with\ i \in [1, N] \quad (2)$$

Equation 2 (which describes the concept of stochastic differentiation) implies that a large swarm operating an action selection based on the same antibody will converge to the original design of the associated recipe.

C. Pre-experimental phase

Since we have now all the necessary elements to model the concept of virtual lymphnode and its constituent antibodies, we can describe how the pre-experimental phase takes place. We start by providing the system with a set of seventeen initial recipes that are described in the swarm chemistry website[1]. These recipes show a fairly complete collection of observable behaviours that are typical of swarm chemistry and they have been discovered by students, researchers and by using the methodologies summarized in Section II. Each recipe is tested for an evaluation time T_{eval} at the end of which, an antibody is created that features the tested recipe in its action field. The expected utility field is calculated for each individual (as described in Section V-A), so that all the obtained fitness values (one for each swarm individual) are averaged into a single value to be stored in the expected utility field of a single antibody. At the end of the pre-experimental phase, we therefore have a set of seventeen antibodies that will be initialized with a concentration level and stored inside each individual to be later used in the evolutionary algorithm. Note that, at the beginning, each individual contains the *same* set of initial antibodies — this is in contrast to common approaches in genetic computing, in which the diversity of the initial population is a fundamental requirement for successfully implementing bio-inspired evolutionary algorithms [18]; in our presented algorithm, the diversity of the antibodies is obtained as a consequence of evaluating the single individuals in a complete decentralized manner. The diversification of the experiences among the swarm individuals will cause their antibodies to undergo different concentration levels and unpredictably different generations of new antibodies (see Section V-E).

Before the initial time step, each individual randomly (with uniform probability distribution) selects one of the seventeen antibodies, so to then apply the process of action selection.

D. Mechanism and immune network dynamics

After each time stamp, concentrations and affinity values that characterize each antibody inside a lymphnode will

change according to the difference between the expected utility value as stored in the initial set of the selected antibody and the obtained fitness. This is known as *evaluation phase* and it enables the single individual to discriminate between situations that lead to positive or negative feedback. Positive feedback is obtained whenever the chosen set of kinetic parameters previously assigned to the swarm individual leads to a fitness value greater than or equal to the one stored in the previously selected antibody. In cases where the obtained fitness value is less than the expected value, negative feedback is applied. Feedback results in a variation of antibody concentration, with the antibody with highest concentration being selected. The value of the concentration of each antibody over time depends on the affinity values amongst all the antibodies: a positive feedback implies an increase of affinity from all unselected antibodies to the last previously selected antibody; vice versa, a negative feedback implies an increase of affinity from the (last) previously selected antibody towards all other unselected antibodies. These dynamics and related formulas (shown in eq. 3) are borrowed (and adapted) from the work of Ishiguro et al. presented in [19], in which an idiotypic network was implemented for controlling a mobile robot.

Being able to evaluate if the selected antibody leads to a positive or negative feedback is useful for calculating the variation of the concentration of the antibodies composing the idiotypic network inside each lymphnode, since higher concentrated antibodies will be selected during the experiments.

$$\begin{aligned}(a)\frac{\Delta c_i}{\Delta t} &= K_p \sum_{j=1}^{Na} a_{j,i} c_i c_j - K_n \sum_{k=1}^{Na} a_{i,k} c_i c_k - K_d c_i \\ (b) a_{i,j} &= \frac{T_{ni} + T_{pj}}{T_{i,j}} \\ (c) c_i &= \frac{1}{1 + exp(0.5 - c_i)}\end{aligned} \tag{3}$$

The differential equation the regulates the concentration over time ($\frac{\Delta c_i}{\Delta t}$ as seen in eq. 3(a)) of the i^{th} of the Na antibodies inside a single lymphnode is composed of three separate terms. The first term is used for summing the stimulation signals, the second term is used for taking into account the suppression signals and the third term is a decay rate that represents the tendency of an antibody to die in absence of any interaction. Each of these terms is regulated by constants (K_p, K_n and K_d) able to balance the weight of these three separate terms over the differential equation. The concentration eq. 3(a) depends on the affinity value ($a_{i,j}$): according to eq. 3(b) a generic affinity between antibody i and j is calculated according to the number of times the selection of antibody i resulted in a negative feedback (T_{ni}) as well as the number of times in which the selection of antibody j provided a positive feedback (T_{pj}). $T_{i,j}$ is total number of times that both these antibodies have been selected. Eq. 3(c) is the squashing function used in order to ensure stability in values of the calculated concentration.

The highest concentrated antibody will be selected as response for the next time interval. Adaptation occurs through the implementation of a mutation operator that enables the lymphnode to create new antibodies as genetic mutations of existing ones. In [19], the process of mutation was described as a continuous process — in contrast, in our case study mutations are triggered by stagnation of the network. This is detected when, after updating affinities and concentrations of the stored antibodies, more than one antibody within a lymphnode shares the same highest level of concentration.

E. Mutation operator

As pointed out in Section V-D, the generation of new antibodies is a process that occurs whenever more than one antibody inside a single lymphnode features the same level of highest concentration. Let us suppose that at the end of an evaluation phase and after the update of the concentration values, M antibodies feature the same highest level of concentration. A new recipe is created by averaging all the last selected kinetic parameters inside the previously selected different M antibodies into a single tuple of kinetic parameters — as a result, the newly generated recipe will feature just one type. This average is weighted by the number of times each of the M antibodies has been previously selected, so that if an antibody has never been selected before, it will not participate in the mutation operation. The newly generated antibody is then matched against all the other antibodies stored in the same lymphnode and erased if it features the same tuple of kinetic parameters. If the new antibody survives this post-generation control, it will be initialised with the same level of initial concentration as in the pre-experimental phase and marked to be selected for the next time interval.

The algorithm enables mutation and generation of new recipes according to two levels, a *macroscopic* level and a *microscopic* level. Macroscopic mutations emerge during the first time interval of the experiment, since we are mixing particles that selected different actions hence have different kinetic parameters. From the perspective of the entire swarm, this results in a single recipe that describes the different choices of each individual. Microscopic mutations occur within the process detailed in this section that relates to how a single particle creates new tuples of kinetic parameters.

F. The algorithm

Algorithm 1 shows the mechanism behind the proposed approach as an iterative process that repeats itself with a frequency described by the parameter T_{eval}. After the pre-experimental phase (see Section V-C), each individual lymphnode randomly chooses an antibody (with uniform probability). After the action selection process related to this selected antibody, the entire swarm is free to move for an evaluation time T_{eval}: this constitutes the *testing phase*. At the end of the testing phase, an evaluation phase as detailed in Section V-D occurs for every swarm individual: after assessing if the previous tested recipe caused positive or negative feedback, affinity and concentration values are updated. The highest concentrated antibody will be selected: in case of network

stagnation (i.e. more than one antibody features the same highest concentration level) a mutation operator is performed, as detailed in Section V-E. Once the process of selection of the next antibody is complete, the individual starts over from the testing phase.

Algorithm 1: AIS-inspired evolutionary swarm chemistry

```
Data: Initial population of antibodies (AB) after
      pre-experiments is initialised
Executed by: each lymphnode L_ci;
Uniform random selection among AB ;
From selected antibody Ab_j → select action A_j;
TEST: while (t < T_eval) do
    test the selected action;
end
performance = evaluate(H_o,H_e);
updated affinities and concentrations;
if stagnation condition is true then
    apply Mutation Operator;
    add newly generated antibodies to list of antibodies;
    select newly generated antibody;
end
else
    Select highest concentrated antibody;
end
Action selection related last selected antibody;
goto: TEST;
```

Algorithm 1 summarizes the most important steps followed by each swarm individual during the duration of the experiment. Some additional mechanisms that relate to how to globally evaluate the performance of the whole system are presented in the next section.

VI. Experiments and discussions

Experiments were performed by implementing the described algorithm on the top of the Java^{TM} implementation of Sayama's swarm chemistry framework, which is available as open source[1]. The system used for simulation features a 2.2 GHz processor, 800 MHz FSB, 4 GB Memory, Java version 7u51 on 64 bit machine. Table I shows the parameters involved in the simulations, as well as the equation that they relate to.

TABLE I
Table of parameters and respective values used in the simulations.

Parameter	Value	Equation
# of Particles	300	-
K_p	0.5	eq. 3
K_n	0.5	eq. 3
K_d	0.0125	eq. 3
K_0	10	eq. 1
T_{eval}	4 sec	-

Testing the proposed algorithm is a two-fold process: on the one hand, we will show in the form of a time line what happens during the evolutionary behaviour of the swarm. On the other hand, we will see how it is possible to engineer a system able to autonomously take into account of whether the swarm has evolved into new interesting features. This latter aspect will be possible thanks to the introduction of the concept of *distance between recipes* (see Section VI-B).

A. Evolutionary behaviour

Ten different runs with different seeds were used during the experiments. As visible in Algorithm 1, the first operation performed by each swarm individual is to randomly select one of the seventeen antibodies created after the pre-experimental phase. During the first time interval after the pre-experimental phase (T_{ape}), from a swarm-oriented point of view, the behaviours of the separate initial recipes are essentially still retained: parts of the swarm aggregate within a small area, while other parts are strongly separated. Many particles also fluctuate among the sub-swarms in a periodic motion. We indicate with (t_0) the first time interval in which the immune network starts to self-organize, as fitness values for the individuals are evaluated at the end of T_{ape}. Some examples of evolutionary behaviours (obtained from three different seeds) are shown in Fig. 4: a typical swarm formation after the pre-experimental phase is shown (only one is shown as it depends on uniformly random choices), followed by the swarm formation at t_0, $0.25T_{exp}$, $0.75T_{exp}$ and T_{exp}, with the latter one being the total duration of experiments in terms of multiples of T_{eval}. The duration of a single experiment (T_{exp}) depends on the time that is necessary for reaching a condition of convergence: this condition implies that at a certain point in time, most of the swarm does not change its recipes during the subsequent time intervals.

As visible in Fig. 4, after T_{ape}, the swarm converges towards a restricted number of recipes, and later reforms into a formation with a large number of different recipes in the latter time intervals. In the intermediate time intervals, notable behaviours are easy to spot and we can identify both known and new behaviours. The list of known behaviours that we can observe from Fig. 4 (as already stored in the initial set of antibodies in the form of recipes) includes a swarm the splits into similar sub-swarms (**j**), gets divided by an "invisible wall" (this expression comes from previous Sayama's work, and it is visible in **f**), features particles that float around an highly oscillatory nucleus (looks like the initial recipe called *linear oscillator*[1], and similar configurations are visible in **a**, **e** and **i**). The swarm may also organize in concentric circles (**g**, **k** and **l**).

New behaviours are represented by the fact that the newly created recipes appear, after a visual inspection, to be qualitatively more complex: most of the known behaviours are observed singularly at the level of single recipe, while due to the proposed algorithm we are able to mix them and find many of their features in a single (evolved) swarm. Specifically with respect to the concentric circular formations that are created as a result of an evolutionary process, constituent circles features different distances to each other and different

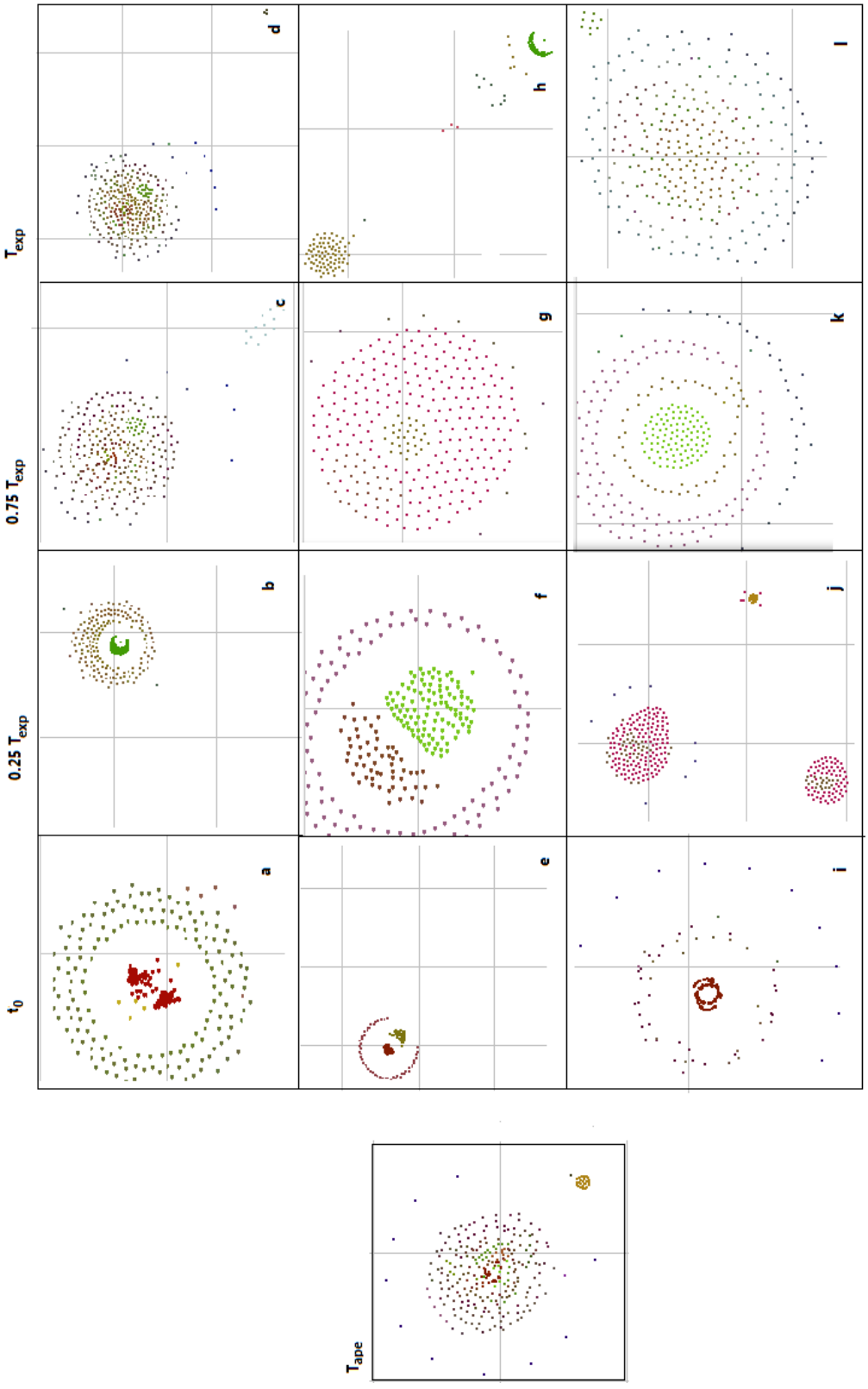

Fig. 4. Timeline describing the some evolutionary behaviours observed throughout the experiments

values of thickness. Also, in **e** and **i** we can see how semi-circular structures are also created. A more difficult behaviour to spot on a still image is one we have named *tail* behaviour, in which a small sub-section of the swarm is attracted to a larger part of the swarm as the small one follows the big one in a thin and long structure (the tail, as most visible in the bottom-right corner of **c**). In **h**, we can see a strongly separated swarm that features two very diverse sub-structures; also in this picture and together with **b**, what we call *the umbrella shape* is created and it is clearly recognizable by its green colour. A video of a typical swarm evolution can be seen as an on line resource[3].

B. Qualitative evaluation

The text so far describes that the evaluation of new behaviours and shapes are recognised through visual inspection. As stated in the introduction, we want to provide the system as a whole with the means to self-recognize whenever new shapes or behaviours have been found, in order to avoid the presence of a human being in the evolutionary process of recipes discovery. As an implication of that, the system should be able to discriminate whenever a newly created recipe is *different* enough from the ones it already knows. Therefore, in this section we introduce the concept of *distance* between recipes. This has some relation to the immunological concept of *self-non self discrimination*: initially, we consider the collection of the seventeen initial recipes generated from after the pre-experimental phase as *self*, and whenever a new recipe is created through our immune network approach, this new recipe is matched with all the entries of the self-set and it is added to this set whenever the minimum value among all the calculated distance values is above a certain threshold T_h. Calculating the distance between two recipes is not a trivial task, since two recipes may be composed of different types and for each type, different populations will follow different kinetic parameters. In order to solve this problem, we introduce two dimensions for characterizing a single observed recipe.

To define the first dimension we need to fix a recipe that acts as a reference point: for this purpose we indicate as RZ (recipe zero) a recipe that is composed of one type that features a population of one individual with each of its kinetic parameters set to the minimum value (i.e. 0). Then, referring to a generic single recipe r_1, we introduce the *weighted average tuple* d_{r_1} that we are able to obtain with the equations in eq. 4.

$$\begin{aligned}
(a) w_i &= \frac{P_i}{P_{MAX}} \; with \; i \in [1, \ldots, N] \\
(b) x_{i,Par_k} &= w_i \frac{Par_k}{MAX_{Par_k}} \; with \; Par_k \in \{R, V_n, \ldots, c_5\} \\
(c) d_{r_1} &= \langle \frac{\sum_{i=1}^{N} x_{i,R}}{\sum_{i=1}^{N} w_i}, \ldots, \frac{\sum_{i=1}^{N} x_{i,c_5}}{\sum_{i=1}^{N} w_i} \rangle
\end{aligned} \tag{4}$$

According to the equations in eq. 4, d_{r_1} is a tuple containing a weighted average of each kinetic parameter (indicated with Par_k), normalized over the maximum value they can assume in the swarm chemistry framework (MAX_{Par_k}). The weight w_i of the i^{th} of the N types of the recipe is calculated over the ratio between the corresponding population of the i^{th} type (P_i, as we are following the notation used in Section III) over the maximum value of the population of a chemical swarm ($P_{MAX} = 300$). Our first dimension is therefore represented by an eight-dimensional Euclidean distance between d_{r_1} and d_{RZ} that we call $X(r_1)$: the purpose of this dimension is to summarize the total swarm within a single tuple in a first attempt to provide a global view of the swarm behaviour, even if the swarm is characterized by a recipe that features $N > 1$ types. However, by doing so we lose the important information regarding the *fragmentation* of the recipes: in other words, two recipes should also be compared in terms of how the total population is divided among the different types. For this reason, we introduce a second dimension for each recipe representing a fragmentation index ($frag_I$), shown in equation 5.

$$frag_I = \frac{max(P_1, \ldots, P_N)}{P_{MAX} - N} \tag{5}$$

A single recipe is now characterized by two coordinates relating dimension X and $frag_I$, so that the final distance between two generic recipes (r_1 and r_2) is therefore their Euclidean distance calculated over this two coordinates and we indicate this distance as $D(r_1, r_2)$. The threshold value (T_h) used to discriminate between self and non-self, is calculated (after creating the initial self) as $T_h = min[D(r_i, r_j)], \forall r_i, r_j \in (self), r_i \neq r_j$. In other words, we create the initial set from the seventeen initial recipes as initial self, then for each of these recipes the system calculates the distance between all the other recipes, with the minimum obtained value being the threshold value that discriminates between self and non-self. As an implication of this, when a newly created recipe is matched against the self repository, it will successfully be a part of the self if its distance between all the self elements are greater or equal than T_h: in case there is a self element whose distance with this new recipe is less than the threshold value, this new recipe is assumed to be too similar to the ones that the system already knows, and thus will be discarded.

As a proof of concept of the introduced distance between recipes the reader can consider this: in Fig. 5 we can see the two recipes (**a**,**b**) from the initial self-set that provides the minimum distance value (T_h): **a** and **b** show the behaviours of the recipes in absence of any perturbation and through visual inspection they do not hold strong similarities as **a** is composed of three types, while **b** has just two. Moreover, the oscillatory red nucleus of **a** oscillates around a perfectly circular trajectory, while in **b** a similar nucleus moves back and forth a straight line, since dashed blue lines represent these movements. However when we interactively add perturbations

[3]https://vimeo.com/92732849

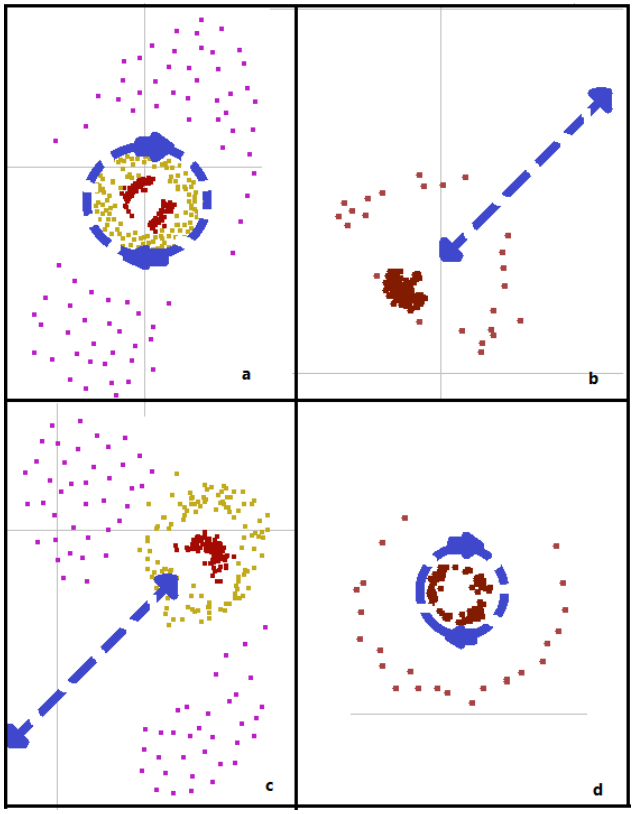

Fig. 5. Representation of the *closest* two recipes of the initial self-set according to the distance measure we introduced (**a**, **b**). Oscillatory behaviours under perturbations show similarities (**c**, **d**).

in the swarms[4], we can see that the responsive behaviours of these two swarms (in term of oscillatory behaviours in **c** and **d**) shows that these two swarms are somehow related: although **c** is a perturbed version of **a**, we can see that the oscillatory nucleus in **c** now resembles the one in **b** while the circular motion of **d** (the perturbed version of **b**) is similar to the one in **a**.

As experiments are repeated over ten different runs, in each run, all the recipes generated until the stopping criteria were stored and compared to the self-set according to the distance measure defined in this section. Some examples recipes that the system autonomously labelled as *new* and added to self are depicted in Fig. 6. Proportions of recipes added to self or erased are shown in the pie chart in Fig. 7.

VII. Conclusion and Future Work

In this paper we introduced a novel approach based on the application of artificial immune systems in the field of morphogenetic engineering; more specifically, an idiotypic network was designed in order to foster the evolutionary ability of chemical swarms so to study the emergence of new shapes

[4]See Sayama's swarm chemistry related publications for details about the dynamics of these interactive perturbations

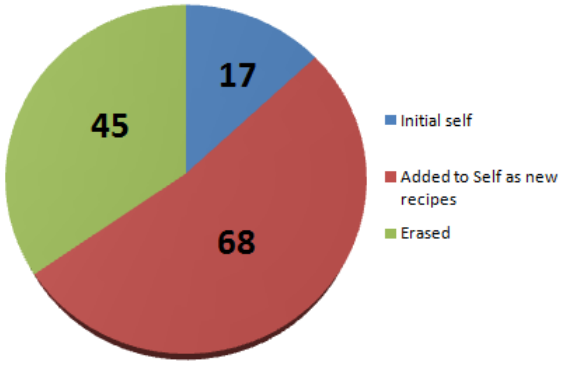

Fig. 7. A pie chart depicting the situation at the end of the self-creation process after ten runs. All the 68 recipes were autonomously recognized as new.

and behaviours on the point of view of the single particle. The design methodology [13] used for this purpose has been previously and successfully applied to a different case study in swarm robotics [14]. In this case study, the algorithm was implemented on the top of an existing implementation of an artificial chemistry framework and involves mutations at the level of mixing pre-existing recipes and in the generation of new tuples of kinetic parameters to be assigned to particles thanks to a newly introduced genetic operator.

In addition to that, a novel measure of distance among recipes has been introduced, so to have a system able to autonomously label whenever newly generated recipes actually show new behaviours compared to known recipes. The proposed algorithm was tested in a number of runs and evaluated in terms of number of behaviours that the system is able to classify as *new*, with results visible in Section VI. We therefore believe that the presented work represents an interesting addition to the already published methodologies for evolutionary swarm chemistry. Being swarm chemistry a morphological model that features both Self-Repair and Self-Organizing behaviours, with our approach we have also investigated how to add Self-Evaluating properties in this case study; by generalizing this approach we believe to have further contributed to Self-* systems related literature.

As for future work, investigating alternative fitness functions is crucial in order to understand more about the newly generated behaviours. For instance, it could be interesting to test a fitness function for an individual that depends on its trajectory over time rather than its vicinity with other particles. Also, a more in depth analysis of what happens in the swarm individual transitions between two recipes could provide interesting points to discuss.

Another future work is related to the exploitation of the proposed approach in the field of context-dependent applications, in order to explore the definition of recipes that are not only related to space but also to information.

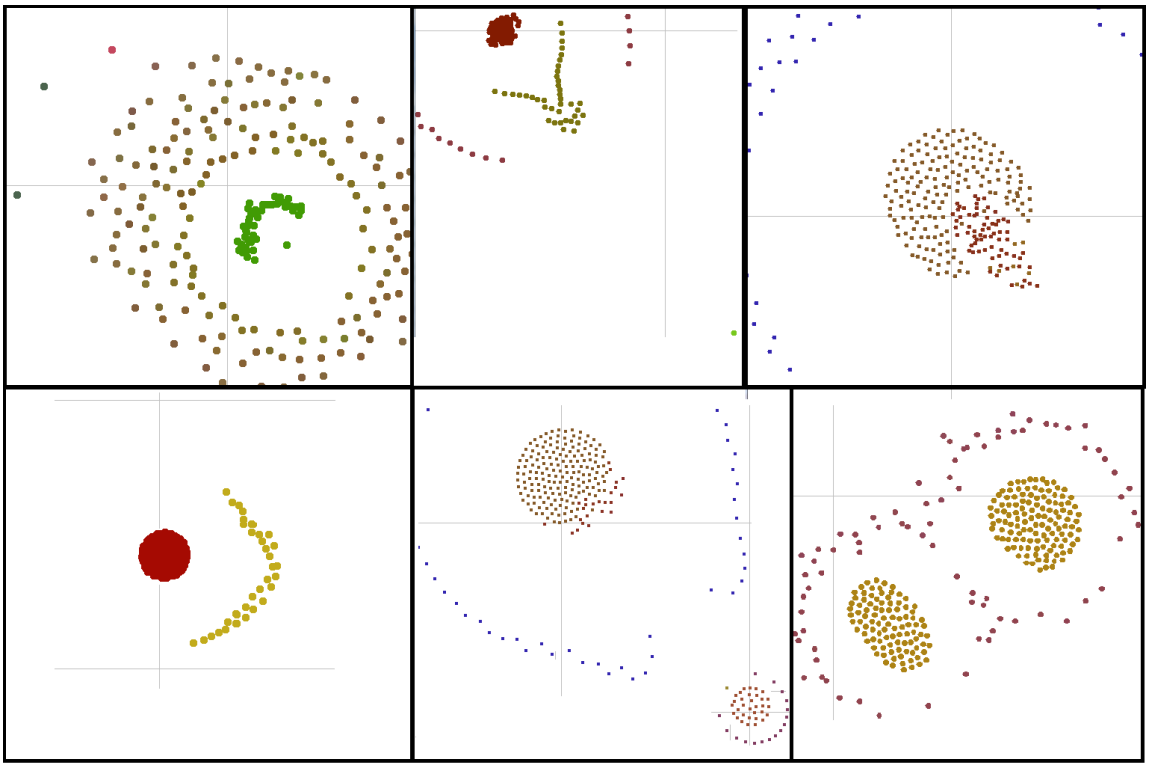

Fig. 6. A collection of newly generated recipes that the system autonomously labelled as new.

ACKNOWLEDGMENT

The work is partially supported by the ASCENS project (EU FP7-FET, Contract No. 257414), and by the "AMBIT - Algorithms and Models for Building context-dependent Information delivery Tools" project funded by Fondazione Cassa di Risparmio di Modena.

REFERENCES

[1] R. Doursat, H. Sayama, and O. Michel, *Morphogenetic Engineering: Toward Programmable Complex Systems*. Springer, 2012.

[2] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer.*, vol. 36, no. 1, pp. 41–50, 2003.

[3] H. Sayama, "Decentralized control and interactive design methods for large-scale heterogeneous self-organizing swarms," in *Advances in Artificial Life*. Springer, 2007, pp. 675–684.

[4] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4. ACM, 1987, pp. 25–34.

[5] I. R. Cohen, "Discrimination and dialogue in the immune system," in *Seminars in Immunology*, vol. 12, no. 3. Elsevier, 2000, pp. 215–219.

[6] ——, *Tending Adam's Garden: evolving the cognitive immune self*. Academic Press, 2000.

[7] P. Husbands, G. Jermy, M. McIlhagga, and R. Ives, "Two applications of genetic algorithms to component design," in *Evolutionary Computing*. Springer, 1996, pp. 50–61.

[8] B. J. Bush and H. Sayama, "Hyperinteractive evolutionary computation," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 3, pp. 424–433, 2011.

[9] H. Sayama, "Seeking open-ended evolution in swarm chemistry," in *Artificial Life (ALIFE), 2011 IEEE Symposium on*. IEEE, 2011, pp. 186–193.

[10] L. Bai and D. E. Breen, "Chemotaxis-inspired cellular primitives for self-organizing shape formation," in *Morphogenetic Engineering*. Springer, 2012, pp. 209–237.

[11] A. L. Christensen, R. O'Grady, and M. Dorigo, "Parallel task execution, morphology control and scalability in a swarm of self-assembling robots," in *Proceedings of the 9th Conference on Autonomous Robots and Competitions. IEEE Press, Piscataway, NJ*, 2009.

[12] N. Capodieci, E. Hart, and G. Cabri, "Designing self-aware adaptive systems: from autonomic computing to cognitive immune networks," *In Proceedings of the 3rd Workshop on Challenges for Achieving Self-Awareness in Autonomic Systems, Philadelphia, USA*, 2013.

[13] ——, "Artificial immune system in the context of autonomic computing: integrating design paradigms," *Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2014*, no. -, p. to appear, 2014.

[14] ——, "An immune network approach for self-adaptive ensembles of autonomic components: a case study in swarm robotics," in *Advances in Artificial Life, ECAL*, vol. 12, 2013, pp. 864–871.

[15] E. Hart and J. Timmis, "Application areas of ais: The past, the present and the future," *Applied soft computing*, vol. 8, no. 1, pp. 191–201, 2008.

[16] L. N. De Castro and J. Timmis, *Artificial immune systems: a new computational intelligence approach*. Springer, 2002.

[17] N. K. Jerne, "Towards the network theory of the immune system," *Ann. Immunol.(Inst. Pasteur)*, vol. 125, pp. 373–389, 1974.

[18] D. Floreano and C. Mattiussi, *Bio-inspired artificial intelligence: theories, methods, and technologies*. MIT press, 2008.

[19] A. Ishiguro, R. Watanabe, and Y. Uchikawa, "An immunological approach to dynamic behavior control for autonomous mobile robots," in *Intelligent Robots and Systems 95. Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ*. IEEE, 1995.