International Scholarly Research Network
ISRN Artificial Intelligence
Volume 2012, Article ID 365791, 13 pages
doi:10.5402/2012/365791

Research Article

Optimization of Swarm-Based Simulations

Sebastian von Mammen,[1] Abbas Sarraf Shirazi,[1] Vladimir Sarpe,[1] and Christian Jacob[1,2]

[1] *Department of Computer Science, Faculty of Science, University of Calgary, Calgary, AB, Canada T2N 1N4*
[2] *Department of Biochemistry & Molecular Biology, Faculty of Medicine, University of Calgary, Calgary, AB, Canada T2N 1N4*

Correspondence should be addressed to Sebastian von Mammen, s.vonmammen@ucalgary.ca

Received 14 March 2012; Accepted 8 April 2012

Academic Editors: F. Camastra, K. W. Chau, and K. Rasheed

In computational swarms, large numbers of reactive agents are simulated. The swarm individuals may coordinate their movements in a "search space" to create efficient routes, to occupy niches, or to find the highest peaks. From a more general perspective though, swarms are a means of representation and computation to bridge the gap between local, individual interactions, and global, emergent phenomena. Computational swarms bear great advantages over other numeric methods, for instance, regarding their extensibility, potential for real-time interaction, dynamic interaction topologies, close translation between natural science theory and the computational model, and the integration of multiscale and multiphysics aspects. However, the more comprehensive a swarm-based model becomes, the more demanding its configuration and the more costly its computation become. In this paper, we present an approach to effectively configure and efficiently compute swarm-based simulations by means of heuristic, population-based optimization techniques. We emphasize the commonalities of several of our recent studies that shed light on top-down model optimization and bottom-up abstraction techniques, culminating in a postulation of a general concept of self-organized optimization in swarm-based simulations.

1. Introduction

Agent-based modelling techniques have prepared the stage for the systematic exploration of complex systems. The interconnection of multiple simple, state-based units, as propagated in cellular automata [1] or random boolean networks [2], yields complex, a priori unpredictable but iteratively computable system behaviours. Discretization and confined interaction spaces have rendered a systematic and comprehensive investigation possible that has provided far-reaching conceptual insights—most prominently the identification of complexity classes and the provision of tools for the classification and analysis of complex systems [3, 4].

Taking the alternative route and trying to consider and integrate even minute details unearthed by natural scientists and amalgamating them into one comprehensive computational model are a daunting task. Yet, steps in this direction have been successfully taken. Material scientists have paved the road in the field of multi-scale model integration in order to gain insights into the properties and behaviours of compound materials [5]. Biomedical researchers have recently been taking similar approaches that target numerous scales of human physiology—from the level of gene expression upto a human population [6]. The integration of model data different from traditional equation-based systems is also moving forward. Recent trends in developmental simulations, for instance, integrate high-level agent behaviours, such as morphogenesis or proliferation, and physical environmental constraints [7–10]. Although these simulations are typically confined to lattice spaces, often even to two spatial dimensions only, they show considerable promise in retracing natural phenomena of growth and physiological development.

Unfortunately, one inevitably faces a tradeoff between real world phenomena and the intricacies of the corresponding models, between the number of interdependent variables and computational viability—in terms of computational efficiency and of effectiveness regarding the expected results. Agent-based models scale particularly poorly with increasing degrees of interaction and increasing numbers of simulated agents. Due to their numerous advantages, exactly these two aspects are emphasized in swarm-based models. These

large-scale multiagent models typically support dynamic interaction topologies, allow the agents to interact spatially, and they target the transition between local interactions and emergent global effects. The great variability in swarms not only demands for special diligence to maintain computational efficiency, for instance, by reducing the search space for interacting individuals based on preceding simulation states [11]. It also exalts the hardship of formulating and parameterizing the agents' behaviours—even the execution order of location update and velocity integration in simple flocking simulations yields fundamentally different global results [12]. These seemingly two distinct problems can both be tackled by optimizing the behaviours of swarm individuals.

In this paper, we present selected works that show how swarms can be optimized to retrace global effects on the one hand and how they can be optimized to maintain computational efficiency on the other hand. In particular, the remainder of this article is structured as follows. In Section 2 we give a brief overview of select topics around the optimization of swarms (as opposed to using swarms for the purpose of optimization). Section 3 demonstrates how swarms can be adapted to meet specific expectations. In Section 4 we present an approach how swarm simulations could re-organize themselves during runtime to maintain computational efficiency. We conclude with a summary of this article and an integrative outlook on swarm optimization in Section 5.

2. Related Work

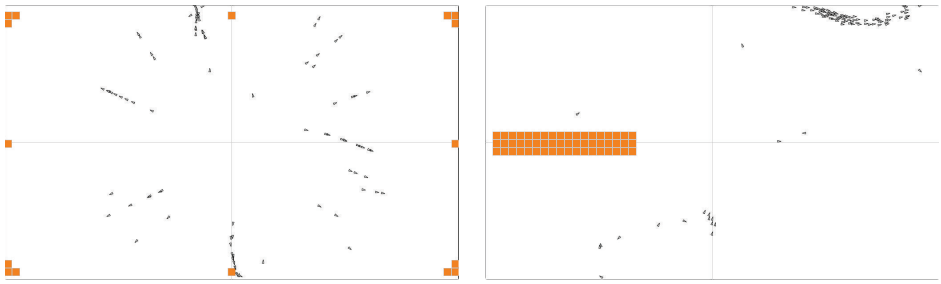
The work presented in this article is inspired and motivated by several disciplines of computer science and their applications. Reynolds raised a lot of excitement in the computer graphics community when he demonstrated the simulation of flocking bird-oids, or boids, at the SIGGRAPH conference in 1987 [13]. Simple acceleration urges steered the boids in accordance with their local neighbourhoods through three-dimensionally rendered virtual worlds. The principles of large numbers of particles attracting and repelling one another in spatial simulations have also received considerable attention by physicists [14–16]. In many occasions, Bonabeau, Camazine, and their colleagues built computational swarm models to retrace the biological behaviours of social insects [17, 18]. Dorigo, Kennedy, and their colleagues were forerunners to apply computational swarms for the purpose of optimization [19, 20].

2.1. Evolution of Constructive Swarms. Some of the mentioned scientists emphasized the applications of computational swarms for visualization or optimization, others focussed their efforts on the design of accurate biological models. Bonabeau et al. for instance, designed agent-based models to examine the nature of the cooperation of social insects. In models of nest construction, agents deposit particles triggered by environmental stimuli. Their behaviour was expressed in sets of rules that test the individuals' neighbourhood situations. Randomly chosen behavioural rules do not yield interesting structures. However, the researchers found rulesets that recreated the shapes of the different wasp genera's nests: Epipona, Parabolybia, Stelopolybia, Vespa, and Chatergus. Marcin Pilat later added rule sets for the wasp families Agelaia, Parachatergus, and Vespula [21]. Motivated by the constructive character of these simulations, some of the authors of this article merged L-systems, formal production systems to generate plant-like geometric structures [22], with the interaction dynamics of swarms (swarm grammars, [23]). Similar to the work in which Henry Kwong and Christian Jacob interactively genetically bred novel parameter sets for boid flock formations [24], swarm grammars were also bred interactively and in immersive breeding grounds in three-dimensional space [25, 26].

2.2. Bottom-Up and Cross-Scale Modelling. Evolutionary breeding techniques have been used to optimize a vast range of computational models—from random boolean networks [27] and cellular automata [28] to L-systems [29] and membrane computing models [30]. Despite their algorithmic and formal universality, the underlying modelling approaches are designed to reflect special properties of the target systems; random boolean networks emphasize the interdependencies of genes; cellular automata and L-systems focus on fixed neighbourhood structures of differentiating cells, whereas membrane computing models; or p-systems; focus on the processes that occur between distinct tissues. Computational swarms find applications across scales—from molecular artificial chemistries to social science simulations—because of their inherently flexible interaction topologies and the focus on the relationship between local interactions and global effects. Therefore, Minar and his colleagues emphasized their multiscalar properties and promoted a hierarchical design approach to swarm models [31].

2.3. Learning Hierarchies. First steps toward the design of emergent multi-scale models—where interactions on one level recursively determine the behaviour of the next higher levels, as opposed to chaining up differential equation systems that operate at different levels—were naturally taken in the domain of artificial chemistries. Rasmussen et al. designed a computational model in which increasingly complex structures emerge exhibiting novel properties—from monomers to polymers to micelles [32]. Although these experiments clearly retrace the formation of patterns at several levels of scale, Dorin and McCormack claim that such phenomena are not surprising given the model's simplicity. Dorin and McCormack argue that it takes considerably more effort to determine the novelties at higher levels in the hierarchy [33].

Dessalles and Phan foresaw a system in which detectors would identify emergent patterns in simulations and subsume the activity of the respective lower level objects [34]. Similarly, Denzinger and Hamdan introduced a modelling agent that observes the behaviours of other agents and maps them to predefined stereotypes [35]. Periodic reevaluations of the agents' behaviours provided the opportunity to adjust the mappings in accordance with the dynamics of the system.



(a) (b)

Figure 1: (a) A flock has learned to swarm to the edges of the simulation space. (b) The flight in formation of a broad stripe maximizes the flock's fitness when hitting the rectangular tiles.

Not only might the local interaction patterns change over time, but high-level phenomena might also influence the underlying layers. Lavelle et al. use the term immergence, or downward causation, to describe the impact of high-level organizations on entities at lower scales [36]. They postulate that explicit functions must be defined to bridge between micro- and macrolevels.

Table 1: Genotype vectors of the boid flocks shown in Figures 1 and 2 (rounded to two decimal places).

Phenotype	α	$d_{\min}$	r	c_{coh}	c_{sep}	c_{ali}	c_{ran}	$v_{\max}$	$a_{\max}$
Figure 1(a)	0.74	90.28	56.16	4.23	1.62	5.0	0.55	6.51	20.02
Figure 1(b)	1.29	100.0	33.70	0.40	3.96	4.53	4.16	8.32	13.17
Figure 2	3.14	100.0	70.84	0.07	3.25	1.12	3.13	8.91	13.45

3. Guiding Emergence

Part of the fascination and the scientific value of computational simulation lies in the prediction of emergent phenomena. The driving computation may be based on various representations, for example, mathematical equations, logical facts, or rule-based interactions. Numeric, iterative simulations can also be used to infer plausible underlying models for a given phenomenon, expressed by means of the utilized representation. Swarm-based simulations are of particular interest as they are typically set up to bridge the gap between local interactions and global, emergent properties and processes (Abduction refers to the corresponding logic-based approach to infer the underlying parts of a model, whereas the field of inverse and ill-posed problems represents the mathematical, analytical analogue.) In this section, we present several approaches to optimize the local behaviours of swarm individuals in order to retrace predefined emergent phenomena. Hereby, we rely on evolutionary computation techniques and we distinguish between fixed predefined target criteria and those that change over time.

3.1. Guiding along 2D Surfaces. Inspired by observations of their natural counterparts, computational swarm models are often represented in two or three spatial dimensions. As the individuals' interactions depend on and impact the corresponding, spatially reflected interaction topologies, swarms lend themselves well for studying emergent phenomena that are graphically representable.

In [37], we showed how a virtual boid flock [38] can be bred so that its individuals maximize the time spent in predefined two-dimensional areas while flocking. In these experiments (Figures 1 and 2), each swarm individual, or boid, is depicted as a triangle that is oriented towards its velocity. It identifies its neighbours inside of a forward-projected conic field of perception that is determined by a radius r and an angle α. To some extent, a boid accelerates randomly; however, its neighbours have a major impact on its trajectory. In particular, a boid follows an urge to align with its neighbours, to flock toward their geometric centre (cohesion urge), and to accelerate away from neighbours that are too close. This separation urge is triggered whenever a neighbour is closer than a given minimal distance. For the given experiment, the alignment, cohesion, and separation vectors are normalized by dividing through the number of neighbours, whereas the random vector is normalized to a unitvector. An individual's acceleration is computed by the weighted sum of these vectors. As a result, the genotype of a boid comprises the parameters for the field of perception (r and α) and the minimum distance $d_{\min}$, as well as the weight coefficients $c_{\text{coh}}, c_{\text{sep}}, c_{\text{ali}}$, and c_{ran} and limit values for both acceleration and velocity, $a_{\max}$ and $v_{\max}$, respectively.

Figures 1 and 2 show boids that were optimized by means of an evolutionary algorithm to flock in the tiled areas (genotypes listed in Table 1). In Figure 1(a) the flock breaks up into several clusters to reach the corners of the simulation space. In a second experiment, the flock formation shown in Figure 1(b) achieves a high fitness value due to the great similarity between its shape and the tiled target area. Another specimen that was discovered in the second evolutionary setting is presented in Figure 2. It solved the given, nonsymmetrical task utilizing the constraints of the simulation environment, great dispersion, but a great degree of connectivity among the boids. In Figure 2(a) the individuals spread radially from the origin. When repelled from the edges, the flock breaks into four parts (Figure 2(b)). To the left and to the right, new clusters form and head back to the world centre (Figure 2(c)), which makes at least one

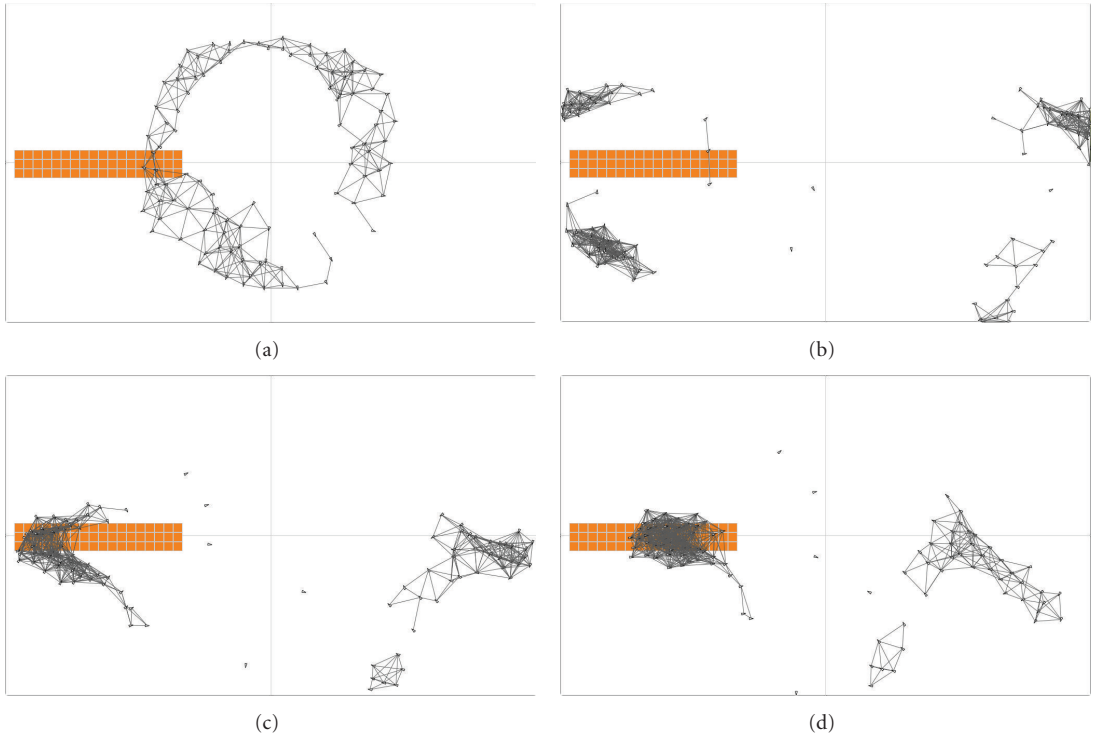


Figure 2: An evolved swarm relies on interactions with the environment in order to hit a nonsymmetrical tiled area.

of the clusters pass across the tiles to the left centre of the simulation space (Figures 2(c) and 2(d)).

In order to breed viable boid parameters for homogeneous flocks, we used a standard Genetic Algorithm (GA) which implemented: (1) rank-based selection: 70% for the best 20%, 20% for genotypes between 20% and 50% of the ranks, and 10% probability for the remainder of the parent population; (2) recombination for half the offspring with multipoint crossover, normally distributed across the genotype; (3) mutation of previous genotypes for the remaining offspring with a mutation probability $p = 0.2$ on single genes. We computed the phenotype fitnesses based on (1). It sums the collisions of boids on all tiles, up to a maximum number of collisions per tile, over the course of a simulation. m denotes the number of swarm agents, n the number of tiles, t_{sim} the simulation time, and the function $c()$ yields the number of collisions between swarm individuals and an individual tile n_{ind} at time step t. In order to promote a smooth distribution of agents across the given tiles, the fitness evaluation function considers at most $c_{\max}$ agents on one tile. The final sum is normalized by the number of simulation steps and the number of swarm agents:

$$f_{2D} = \frac{1}{t_{\text{sim}} m} \sum_{n_{\text{ind}}=0}^{n} \sum_{t=0}^{t_{\text{sim}}} \min(c(n_{\text{ind}}, t), c_{\max}), \quad \text{with } c_{\max} = \frac{m}{n}. \tag{1}$$

The genotypes of the three presented cases are detailed in Table 2. The first one, depicted in Figure 1(a), yields a high degree of scattered clusters due to the high cohesion and alignment weights and the narrow perception angle. The third genotype (Figure 2) is a descendant of the second one (Figure 1(a)). Its cohesion and alignment weights dropped significantly while its perception radius increased to the maximally possible value. $d_{\min}$ is greater than the actual perception radius in all three cases which implies that the separation urge was consistently triggered by all perceived neighbours.

Table 2: Flocking genotypes of the constructive swarms shown in Figures 3(a), 3(b), 3(c), and 3(d), respectively (rounded to two decimal places).

Phenotype	c_{coh}	c_{sep}	c_{ali}	c_{ran}	c_{foc}	c_{gro}
Figures 3(a) and 3(b)	0.18	0.06	0.30	0.00	0.14	0.17
Figures 3(c) and 3(d)	0.16	0.43	0.16	0.00	0.23	0.12

3.2. Guiding through 3D Volumes. In [39, 40], we presented an approach to guid swarm dynamics very similar to the one in Section 3.1. The model was inspired by work on nest construction in social insects [17, 18]. In this model, in addition to following the flocking parameters outlined in Section 3.1, environmental stimuli prompted the individuals to place or remove cubic building blocks in virtual three-dimensional space (gravitation was not simulated, intersecting building blocks not allowed). The individuals' construction behaviour was expressed as *if-then* rules. The rules' antecedents would test the existence of up to five building blocks that were positioned relative to the acting individual. The consequence of each of twenty allowed rules could trigger the creation or destruction of a building

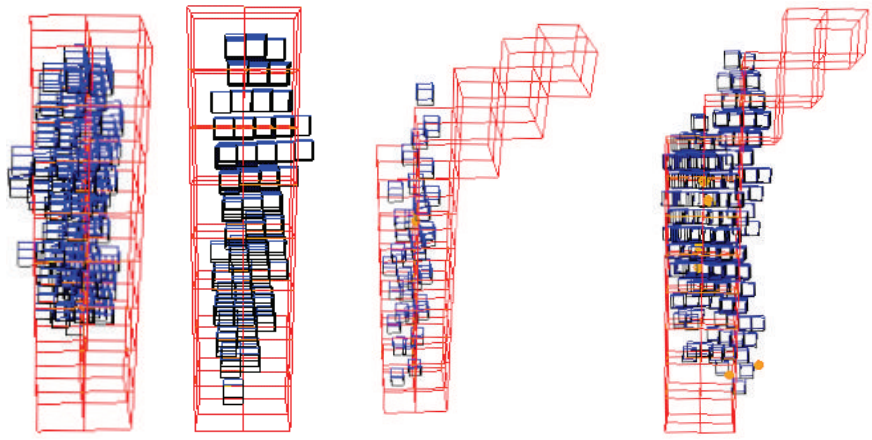


Figure 3: Swarm constructions (inner aggregations) are guided by predefined 3D structures (outer grids).

block at specified relative coordinates, or it could set or reset the acting individual's point of focus coordinates. If set, the individual would be accelerated towards the point of focus alongside of the basic boid urges of alignment, separation, cohesion, and some random acceleration. In addition, we also introduced an acceleration urge toward the ground that would increase with an individual's height. c_{foc} and c_{gro} denote the weight coefficients for these two additional urges, respectively.

Again, we used a standard GA to breed swarms that were guided by geometrical constraints. This time, the fitness of a swarm was determined by the ratio of building blocks built inside and outside of a predefined three-dimensional structure composed of a set of cubes. An initial seed cube marked the site a swarm's construction efforts would be measured against. Figure 3 shows the predefined structures and the swarm-based constructions of two different experiments. Instead of multipoint crossover operators, recombination is performed for 40% of the offspring based on a randomly generated two-point crossover mask that preserves pairs of dependent rules with a greater probability. The number of rules of the offspring is limited to the smaller number of rules of the parents. The parents for all the offspring were chosen by means of fitness proportionate selection. In addition the ten best individuals were always considered as parents (kBest with $k = 10$). Mutation is performed per boid gene with a probability of $m_{\text{boid}} = 0.2$, whereas the conditions, the action, and the action parameter (a relative position) are considered for mutation independently with a probability $m_{\text{rule}} = 0.1$. In the evolutionary experiments, we emphasized the coordination of construction and fixed some of the flocking parameters. In particular, $d_{\min} = r = 2.0$, $\alpha = 2.0$, $v_{\max} = 0.5$, and $a_{\max} = 0.3$. Please note that for these experiments a different simulation environment, VIGO [41], was used which resulted in a spatial scaling factor much smaller than that in Section 3.1.

The construction rule sets of the two independently bred swarms depicted in Figure 3 were dominated by unconditional and conditional rules for cube creation. Each of the swarms also set and reset the individuals' focusses (3 unconditional construction rules in Figures 3(a) and 3(b), 4 in Figures 3(c) and 3(d), and 2 conditional ones in both specimens). In the swarm depicted in Figures 3(a)–3(b), the individuals also unconditionally removed construction elements in a relative location. Further information about these rule sets can be found in [40].

3.3. Tracing and Learning Flock Dynamics. The speciality of a swarm is its inherently dynamic interaction topology and the resulting feedback on its global behaviour. In [42], we analyzed previously discovered boid flock specimens [24] based on their interaction topologies over time. We also presented an approach to finding new flock configurations whose interaction topologies evolved in accordance with predefined functions that reflect naturally occurring phenomena such as biological switches and clocks or timers. In particular, we showed that a step function can be approximated by a flock's average neighbourhood degree $\overline{n}$, if its individuals slowly drift away from one another, and that an oscillating neighbourhood degree can be established by a pulsating flock. Here, we want to share the latter example, as its characteristic sequence of phase transitions is especially interesting in the context of complex simulation research.

As the initial configuration of a complex system may heavily impact the results of a numeric experiment, we encoded the initial configurations (position, velocity, and acceleration) of individuals as part of a swarm's genotype, similar to an epigenetic factor. In order to provide a spatial point of reference, we allow the swarm to urge toward the world centre, $\mathbf{o} = (0,0,0)^T$ (weighted by c_{foc}). This time, we simply configured a Genetic Algorithm with fitness proportionate selection, incremental mutation, and multipoint crossover on all numeric values. To enforce the approximation of a predefined target function, we computed the following fitness value: $f_{\text{oscillation}} = 1/(\sum_{t=1}^{40} |\overline{n}(t) - x(t)|)$. Over a period of 40 time steps, the fitness diminishes

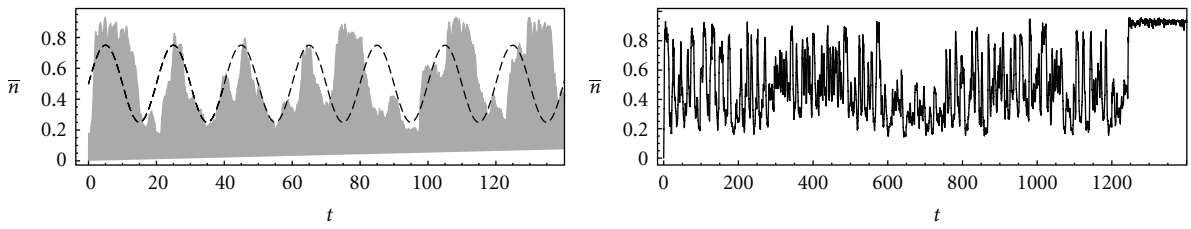


Figure 4: (a) The average neighbourhood of a flock $\overline{n}$ approximates a sine function that it learned until $t = 40$. (b) At $t = 1244$, the flock forms a tight cluster and remains in an equilibrium with $\overline{n} \in [0.35; 0.45]$.

Table 3: Evolved swarm parameters that result in the neighborhood evolution shown in Figure 4. The corresponding flocks oscillate though repeated contraction and expansion (Figure 5).

Phenotype	α	$d_{\min}$	r	c_{coh}	c_{sep}	c_{ali}	c_{ran}	c_{foc}	$a_{\max}$	$v_{\max}$
Figures 5 and 6	2.64	4.12	7.86	0.95	0.53	0.76	0.76	0.36	12.15	7.16

Table 4: Average parameters of two classes of swarm-based predator-prey models that were found using Particle Swarm Optimization (rounded to two decimal places).

Phenotype	α	β	γ	δ	$\lvert p_{\text{init}} \rvert$	$\lvert P_{\text{init}} \rvert$	steps
Figure 7(a)	0.38	0.13	0.64	0.18	432.63	317.13	132.63
Figure 7(b)	0.76	0.30	0.69	0.20	436.44	330.64	119.26

proportional to the absolute difference between the target function $x(t)$ and its approximation $\overline{n}(t)$.

Figure 4 shows the neighbourhood function $\overline{n}(t)$ as exhibited by the evolved swarm configuration listed in Table 3. The oscillation happens as the flock repeatedly expands (Figure 5) and contracts (Figure 6). Leaps from a plateau to a local maximum, as seen at $t = 100$, occur when formerly separated flocks rejoin. Eventually, at $t = 1244$, the oscillation ends (Figure 4(b)); this is when the agents form a tight cluster and start orbiting around the world centre. In order to facilitate the identification of flocking patterns, we activated motion blurring in the renderings.

3.4. Parameter Optimization in a Heterogeneous Predator-Prey Model. As a test bed for learning the behavioural parameters of heterogeneous swarms, we chose a classic predator-prey model, in which the populations of prey p and predator individuals P depend on one another [43, 44]. The Lotka-Volterra differential equations (DEs) describe the dynamics of a predator-prey ecosystem ((2) and (3)). In our corresponding, two-dimensional swarm-based model, both prey and predators wander about randomly. Prey dies when encountering a predator. It also dies of other causes with probability β at each step of the simulation, or it reproduces with probability α. Predators prosper from nutritional encounters with prey individuals and reproduce on the spot with a probability γ. Their deaths occur with probability δ. The populations of predator and prey individuals, p_{init} and P_{init}, are initially set to magnitudes between 10 and 500:

$$\frac{dp}{dt} = p(\alpha - \beta P), \tag{2}$$

$$\frac{dP}{dt} = -P(\gamma p - \delta P). \tag{3}$$

We reverse-engineered the parameters for the swarm model relying on several algorithms. First, we discretized the continuous results of (2) and (3) by means of an online time-series segmentation algorithm [45]. We then measured the similarity value between the time series produced by the swarm-based model and the segmented differential equation results using a generic dynamic time warping algorithm [46, 47]. This measure served as the fitness value to search for adequate solutions based on particle swarm optimization (PSO) [48].

Different from the experiments presented in Section 3.3, the swarm individuals in this predator-prey model cannot alternate their velocities. Therefore, in order to approximate a given plot with a fixed time scale, we optimized for qualitative similarity between the swarm simulation and the DE system. We accomplished this by adding the number of simulated steps to the swarm configuration. A single scalar factor suffices to match the evolved and the expected graphs.

In order to foster robust solutions, we ran each simulation three times for a given set of parameters and considered the average performance as the particular swarm's fitness value. Twenty optimization experiments yielded two prototypical swarm configurations (Table 4). Their average evolution over the course of one simulation is depicted in Figure 7. While the overall PSO experiments have converged on two different solutions, each of them is still close to the DE-based results. The second class of solutions, Figure 7(b), qualitatively matches the DE model better as the population of prey individuals recovers at the end of the simulation. We give credit for this development to the greater reproduction rate α of prey individuals as seen in Table 4. The shift between the swarm-based approximations and the DE-based target graphs in Figure 7 is the result of a relatively generous error threshold for the similarity measures.

4. Abstract and Scale

In the previous section, we demonstrated the optimization of swarm behaviours in respect to statically measurable

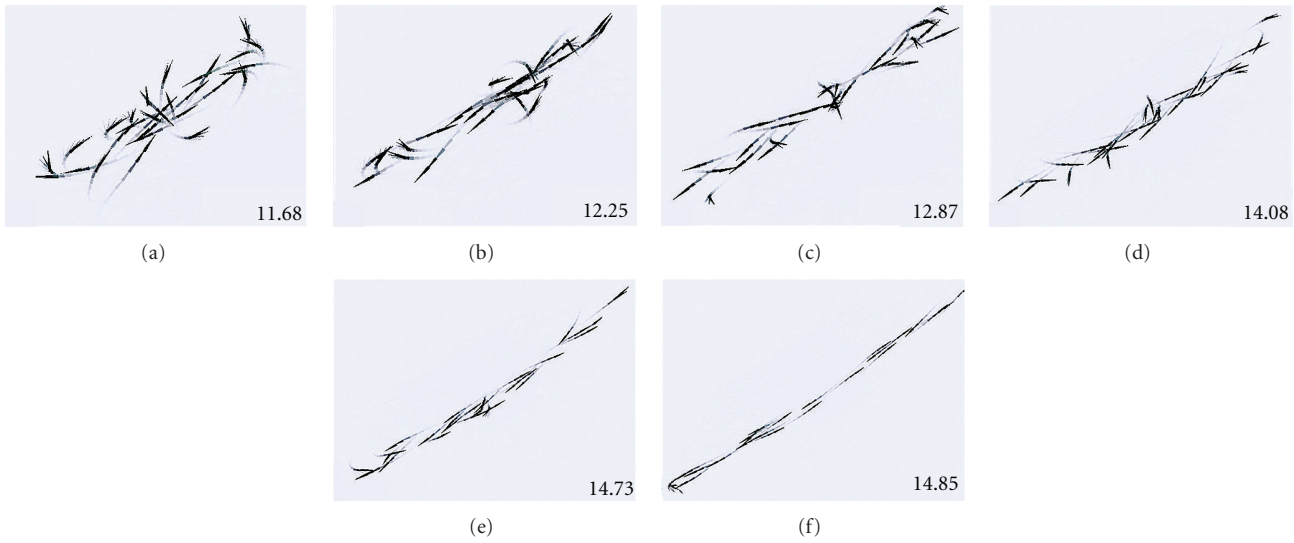


Figure 5: The series of images shows how a swarm in a knot formation expands to two sides. Eventually, two flocks emerge and head into opposing directions.

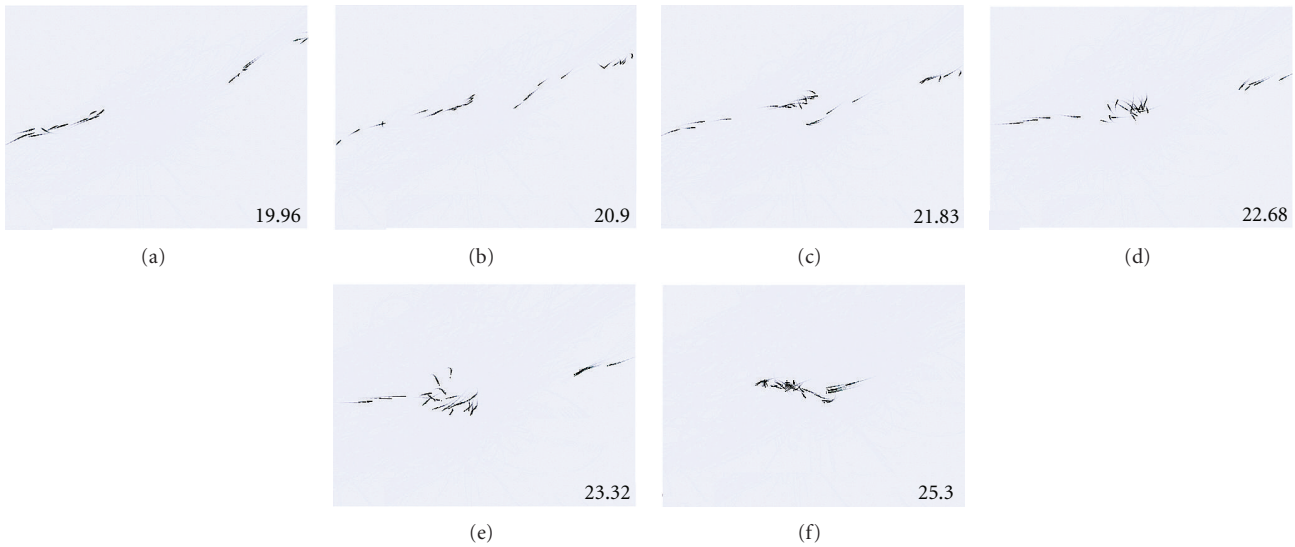


Figure 6: (a-b) Two flocks head toward the world centre from opposing directions. (c) They avoid each other at first. But soon they closely interact again. (The images were adjusted to fit both flocks, the zoom was slightly increased once for capturing (d–f)).

outcomes, dynamics over time, and heterogeneous system compositions. While the resulting systems may suffice to retrace and explore certain isolated phenomena, the extensibility of swarms, their intrinsic potential to interface with newly introduced elements and to yield high-level emergent properties renders scalability of swarms another great challenge.

The flexibility of swarm-based modelling comes at a cost. Without further optimization, the identification of interaction partners of n swarm individuals alone yields a computational complexity of $O(n^2)$. Typically, the interaction scope of large numbers of units may, therefore, be drastically reduced. The interaction in spatial environments is often limited to the local, discrete neighbourhoods relying on discrete computational modelling approaches such as cellular automata or cellular potts [49]. However, the ability of the models to continuously change their interaction topologies among the agents is crucial to capture the systems' dynamics responsible, for instance, for emergent transportation [16]. Of course, this confinement does not only apply to spatial interactions but to the number of dimensions of interactions in general, to the number of individual interaction rules, and to the number of simulated individuals. A system of automated abstraction, which learns the local patterns and subsumes them as high-level agents, offers a perspective for a truly scalable computational approach. Instead of learning

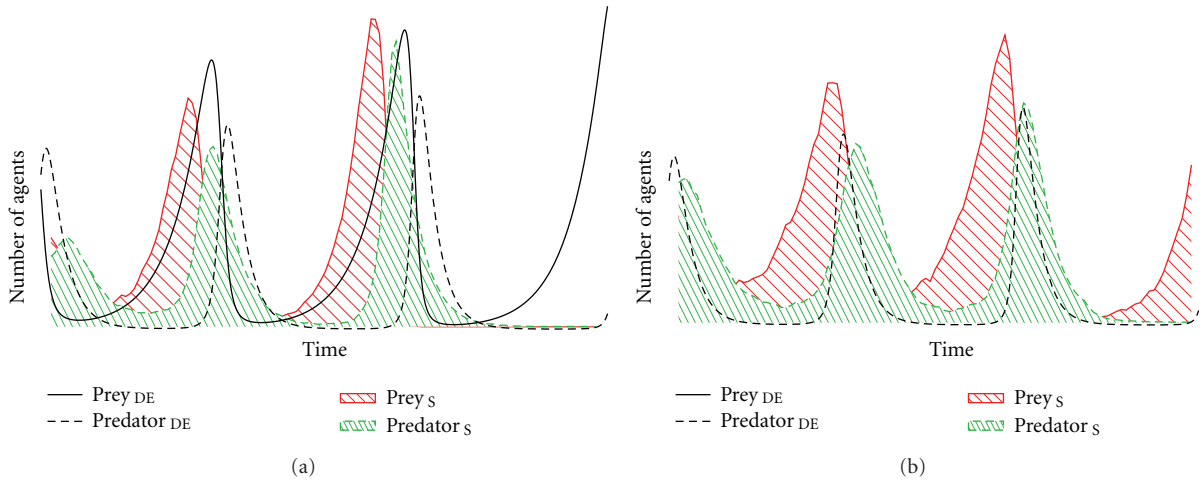


FIGURE 7: (a) and (b) show the population dynamics of two prototypical swarm configurations compared to the results of the Lotka-Volterra DE model of a predator-prey system. The results of both modelling approaches had to be scaled to match (see *steps* in Table 4), yielding these qualitative diagrams.

behaviours motivated by predefined patterns as exercised in the previous section, we now demonstrate how emergent patterns that occur among (properly trained) agents can be learned, rephrased as higher level behaviours, and utilized to reduce the number of simulated agents.

4.1. Towards Self-Organizing Hierarchies. Early on when we started our investigations, we already had a rather clear picture of our envisioned abstraction framework. It should automatically, and in a decentralized fashion, create abstractions in a simulation, whenever possible, and abolish them, whenever necessary. As we imagined it to be primarily deployed in swarm systems, it was obvious to us that the abstractions should be discovered and managed by special swarm individuals that are immersed into arbitrary swarm simulations. We termed this concept self-organized middle-out abstraction approach, or SOMO [50]—"middle-out" referring to the idea that it would create higher level representations (bottom-up) but also break them down again (top-down).

However, in order to ensure the validity of our conceptual foundation, we narrowed down the scope of our first set of experiments [51]. Therein, we identified correlated nodes in gene regulation networks (modelled by a set of simple differential equations), approximated their behaviours as groups by means of artificial neural networks (ANNs), and subsumed the lower level nodes by high-level agents or meta-agents. High correlation values between concentrations would consistently yield higher level agents, whereas drops in the correlation values of previously grouped nodes resulted in the removal of the respective, outdated abstraction. Figures 8(a), 8(b), 8(e) and 8(f) show the results of this *greedy* approach when applied to two different MAPK pathway models, one resulting in a sigmoidal concentration of the MAPK-PP protein [52], the other one in a periodic expression pattern [53]. The relationship between inaccurate emulation by the meta-agents and the number of meta-agents in the system is obvious when comparing Figures 8(a) and 8(e). The occurrence of dips in the otherwise smooth approximative graph triggers the removal of abstractions. In the periodic model, changes occur too frequently to be accommodated by the meta-agents which resulted in a high frequency of their creation and removal (Figure 8(d)).

Although the overall performance of the greedy abstraction approach was far from satisfactory, it successfully reduced the number of agents in the system. In our second set of experiments, we attempted to amend the particularly short lifespans of the abstractions seen in the periodic MAPK model in Figure 8(d). So we promoted a dynamic management of the learned meta-agent hierarchies [54]. Whenever a meta-agent became obsolete, it would restore the subsumed, previously active abstraction hierarchy. Figures 8(c)–8(f) depict the results of this *hierarchical* approach. The stepwise restoration of lower-level abstractions is clearly identifiable in Figure 8(e). At about $t = 2250$ one meta-agent, which was trained by means of standard Genetic Programming (GP), is removed and its two underlying meta-agents are reintroduced into the simulation. Before this point in time, the learning process consistently built greater abstractions. The predictions by the meta-agents exhibited greater inaccuracy than in the greedy case. In addition, the divergence between target graph and approximative results (Figure 8(c)) does not coincide well with the creation and removal of meta-agents (Figure 8(e)); it is surprising that yet another hierarchical level is added shortly after time step $t = 2000$, even though the preceding emulated concentration strongly deviated from the target function.

4.2. Immersive Decentralized Abstraction. We believe that the optimization and further the situation-dependent choice of apt parameter set for the efficient abstraction and hierarchy management necessitate in-depth studies on top of a fully fledged SOMO prototype. Therefore, for our

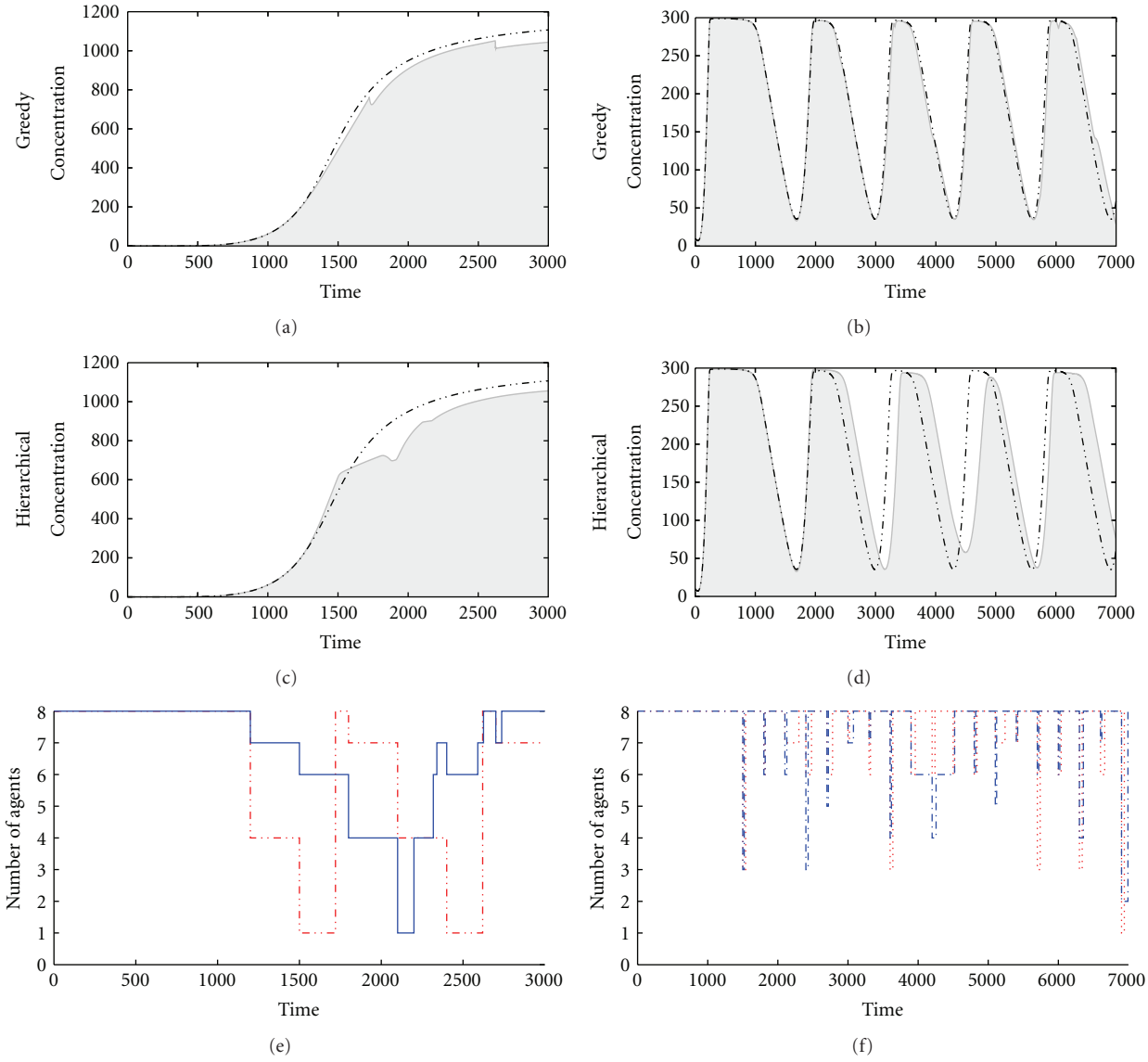


Figure 8: Simulations based on a nonperiodic and a periodic MAPK pathway model are shown on the left-hand and the right-hand side, respectively. Comparisons between the differential equation system and a greedy (a-b) and a hierarchical (c-d) abstraction approach are shown. The results of the DE model are indicted by dashed lines, the agent-based dynamics are depicted as shaded areas. The numbers of agents deployed by the abstraction approaches are compared in (e) and (f) (individual legends are provided in these two diagrams).

next experiments, instead of fine-tuning the parameters to optimize the ratio between agent reduction and accurate emulation, we searched for a better learning example—one that allows for the deployment of self-organizing, abstracting swarm individuals in the context of a swarm simulation. As previous results had suggested (Section 4.1), linear instead of periodic system dynamics promised the best results for a prototype SOMO implementation. Hence, we adjusted the SOMO system and designed swarm individuals that could be immersed into a swarm simulation of the physiological process of blood coagulation.

In addition to the swarm individuals of the model, or *model agents*, we designed an *observer agent*. It observes model agents and logs their interactions in an *interaction history* that serves as a database for pattern recognition. An entry in an interaction history contains, for instance, a reference to the acting agent A, the executed action *act* with time stamp t along with the set of interaction partners $\mathcal{A}$. In our prototype, the observer applies a k-means clustering algorithm [55] to find a cluster of overlapping interaction partners as soon as the interaction history contains a sufficiently large set of logged entries. Once a cluster is identified, the observer infers a generalized group behaviour from the logged interaction data. It learns the information that remains fixed across the set of relevant rules and it identifies boundaries, periodicities, and probabilities of reoccurring variable actions. All the logged interactions that led to the rules of the newly phrased group behaviour are

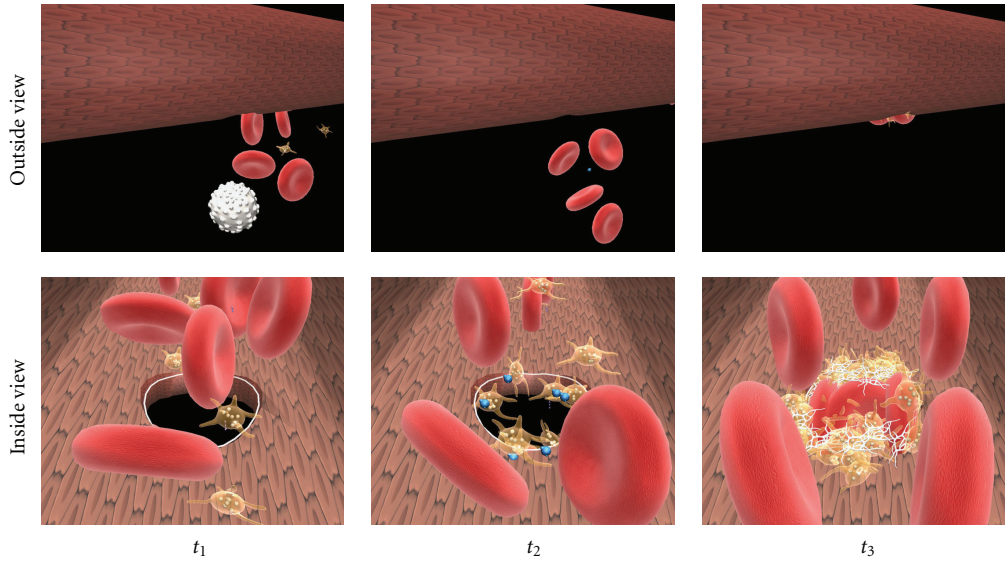


Figure 9: A swarm-based blood coagulation simulation shown from two perspectives at three consecutive time steps $t_1, ..., t_3$.

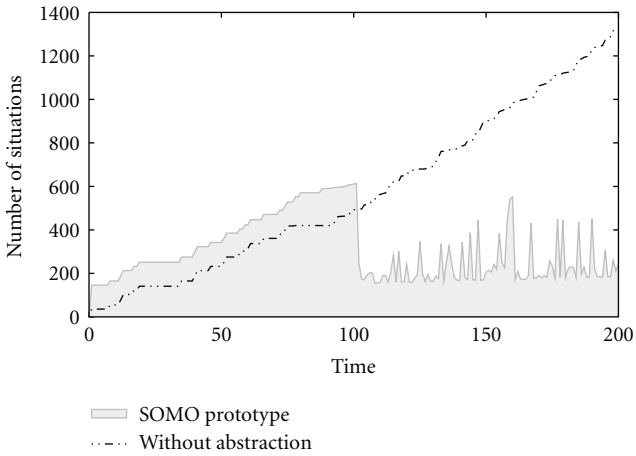


Figure 10: Number of agent situation evaluations over the course of a blood coagulation simulation, with and without SOMO abstraction.

removed from the lower-level individuals and the observer starts performing on their behalf. Initially, the observer has an unbiased confidence in a newly learned abstraction. Periodically testing the behaviour of the subsumed agents in the current situation lets the observer adjust this confidence—it grows, if the predictions were correct, otherwise it drops. The observer fully restores the subsumed agents, should the confidence drop below a certain threshold. The behavioural subsumption by the observer reduces the otherwise necessary tests for triggering actions and the search for the respective interaction partners. Of course, in a deployment scenario, this performance gain would be measured against the computational overhead for observing model agents, abstracting, validating, and possibly removing group behaviours.

We immersed the prototype SOMO observer in a swarm-based blood coagulation simulation in which bioagents aggregate at a wound site and form a clot (Figure 9). After $t = 100$, the observer identifies $k = 30$ clusters in its interaction history and the centroid of the largest cluster is considered to be the learned group behaviour for which $[t_{\min}, t_{\max}]$ and p_{exec} are inferred. At intervals of $\Delta = 10$, the observer updates its confidence values; abstractions with confidence values below $\tau = 30\%$ are revoked. In this environment, our prototype successfully identified and abstracted behaviours such as random movement, executed with probability $p_{\text{exec}} = 100\%$ and $t \in [0, \infty]$, and state changes induced by collision ($p_{\text{exec}} = 77\%$ and $t \in [90, 95]$). Due to the model's simplicity, the number of calculated situations over the course of a simulation increases linearly with the number of incoming bio-agents (introduced by the blood stream). Our prototype managed to keep this number constant (Figure 10). Its overhead is shown in the additionally computed situations that occur just before the abstraction starts ($t < 100$). The peaks in our proposed method indicate the intervals at which some model agents are allowed to execute their actions.

5. Summary and Future Work

Swarm-based models and simulations bridge the gap between the level of local interactions and global system behaviours. Instead of programming a swarm system, one has to program its individuals, and in such a way that the whole swarm can accomplish its task. A computational swarm might, for instance, be designed to retrace and predict natural phenomena, to optimize mathematically phrased problems, or to support creative design decisions. In this article, we presented several experiments that elucidate how the behaviour of swarm individuals can be programmed.

First, in Section 3, we focussed on the interplay of globally defined constraints and the inferred behaviours of locally interacting swarm individuals. Due to the spatial properties of basic boid swarms, we formulated tasks geometrically to (1) evolve flocking swarms in 2D and (2) constructive swarms in 3D (by means of Genetic Algorithms). (3) We introduced a quantitative measure to capture the neighbourhood dynamics of boid flocks that allowed us to genetically breed boid individuals that would, in a group, approximate a predefined neighbourhood density function. (4) A heterogeneous swarm model of predator and prey concluded our explorations of guiding emergence; here, a system of differential equations specified the system dynamics, and the parameters of the two types of swarm individuals were learned (by means of Particle Swarm Optimization).

In Section 4, we then presented several stages toward an inherently scalable approach to swarm simulation, the self-organized middle-out abstraction framework, or SOMO. Here, meta-agents subsume the behaviours of lower-level individuals based on reoccurring interaction patterns in order to reduce the number of required computation steps. Meta-agents organize themselves in hierarchies that are dynamically built up and broken down, depending on the demands of the ongoing simulation and the predictive power of the learned abstractions. In our experiments, we first (5) greedily subsumed low-level agents by meta-agents in an easily verifiable differential equation model of the MAPK signalling pathway (mitogen-activated protein kinase). (6) We introduced a dynamic management of hierarchies, so that, upon the identification of an obsolete abstraction a preceding abstraction is restored instead of resetting all the learned accomplishments all at once. Finally, we (7) equipped special swarm individuals, so-called observer agents, with a behaviour to build and manage abstraction hierarchies based on interaction histories of groups of monitored individuals.

While examples (1) to (4) emphasize the top-down learning, breeding, or optimization of the behaviour of swarm individuals, instances (5) to (7) attempt the opposite; the SOMO approach learns and utilizes patterns that emerge from local interactions bottom-up, only breaking them down again should it become necessary. As much as these perspectives might differ, we believe that they might serve as forerunners of an algorithmic framework for integrative, large-scale and multi-scale modelling and simulation. In the last paragraph of this article, we want to outline how this could work, at the same time implying a suggested direction of future work in this field.

The more specialized the interaction patterns a SOMO observer is looking for, the more efficiently it will identify and abstract them. A set of differently configured SOMO observers spread across the simulation space could evolve based on their success to abstract in their respective niches—one may assume that activity is strongly heterogeneous across the interaction dimensions of most large-scale simulations. At this point, the unsupervised online learning process of SOMO would be two-tear, considering the accuracy of the generated abstraction hierarchies and the configuration of the observer agents. Additional top-down constraints could be introduced by a second observer type that reconfigures individuals in order to reproduce specific process patterns. Such a top-down observer could substantially change the original model, so its influence should be strictly constrained. The conditional introduction and removal of top-down observers, depending, for instance, on the emergence of certain high-level behaviours learned by the currently implemented bottom-up SOMO observers, would enable an external modeller to embed expected milestones into a bottom-up computed multi-scale simulation and ensure the seamless computational integration of its scales.

References

[1] J. von Neumann and A. W. Burks, *Theory of Self-Reproducing Automata*, University of Illinois Press, London, UK, 1966.

[2] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *Journal of Theoretical Biology*, vol. 22, no. 3, pp. 437–467, 1969.

[3] S. A. Kauffman, *The Origins of Order*, Oxford University, New York, NY, USA, 1993.

[4] S. Wolfram, *A New Kind of Science*, Wolfram Media, Champaign, Ill, USA, 2002.

[5] M. F. Horstemeyer, "Multiscale modeling: a review," *Practical Aspects of Computational Chemistry*, pp. 87–135, 2010.

[6] T. Eissing, L. Kuepfer, C. Becker et al., "A computational systems biology software platform for multiscale modeling and simulation: integrating whole-body physiology, disease biology, and molecular reaction networks," *Frontiers in Physiology*, vol. 2, no. 4, pp. 1–10, 2011.

[7] J. Beal, "Functional blueprints: an approach to modularity in grown systems," in *Proceedings of the ANTS 2010: 7th International Conference on Swarm Intelligence*, pp. 179–190, Springer, 2010.

[8] J. Werfel, "Biologically realistic primitives for engineered morphogenesis," in *Proceedings of the ANTS 2010: 7th International Conference on Swarm Intelligence*, pp. 131–141, Springer, 2010.

[9] I. Salazar-Ciudad and J. Jernvall, "A computational model of teeth and the developmental origins of morphological variation," *Nature*, vol. 464, no. 7288, pp. 583–586, 2010.

[10] S. von Mammen, D. Phillips, T. Davison, H. Jamniczky, B. Hallgrmsson, and C. Jacob, "Swarm-based computational development," in *Morphogenetic Engineering*, Springer.

[11] J. Klein, "Breve: a 3D environment for the simulation of decentralized systems and artificial life," in *Proceedings of the 8th International Conference on Artificial life*, pp. 329–334, MIT Press, 2002.

[12] C. Huepe and M. Aldana, "New tools for characterizing swarming systems: a comparison of minimal models," *Physica A*, vol. 387, no. 12, pp. 2809–2822, 2008.

[13] C. W. Reynolds, "Flocks, herds, and schools: a distributed behavioral model," in *Proceedings of the SIGGRAPH '87 Conference*, vol. 4, pp. 25–34, Anaheim, Calif, USA, 1987.

[14] I. Derényi and T. Vicsek, "Cooperative transport of brownian particles," *Physical Review Letters*, vol. 75, no. 3, pp. 374–377, 1995.

[15] A. Czirók and T. Vicsek, "Collective behavior of interacting self-propelled particles," *Physica A*, vol. 281, no. 1, pp. 17–29, 2000.

[16] T. Vicsek and A. Zafiris, "Collective motion," *Reviews of Modern Physics*. In press.

[17] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Santa Fe Institute Studies in the Sciences of Complexity Oxford University Press, New York, NY, USA, 1999.

[18] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*, Princeton Studies in Complexity, Princeton University Press, Princeton, NJ, USA, 2003.

[19] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, The Morgan Kaufmann Series in Evolutionary Computation, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[20] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 26, no. 1, pp. 29–41, 1996.

[21] M. Pilat, "Wasp-inspired construction algortihms," Tech. Rep., University of Calgary, 2004.

[22] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*, Springer, 1996.

[23] C. J. Jacob, G. Hushlak, J. E. Boyd, P. Nuytten, M. Sayles, and M. Pilat, "SwarmArt: interactive art from swarm intelligence," *Leonardo*, vol. 40, no. 3, pp. 248–254, 2007.

[24] H. Kwong and C. Jacob, "Evolutionary exploration of dynamic swarm behaviour," in *Proceedings of the Evolutionary Computation*, IEEE Press, Canberra, Australia, 2003.

[25] S. Von Mammen and C. Jacob, "Genetic swarm grammar programming: ecological breeding like a gardener," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 851–858, Singapore, September 2007.

[26] S. Von Mammen and C. Jacob, "The evolution of swarm grammars- growing trees, crafting art, and bottom-up design," *IEEE Computational Intelligence Magazine*, vol. 4, no. 3, pp. 10–19, 2009.

[27] A. Esmaeili and C. Jacob, "A multi-objective differential evolutionary approach toward more stable gene regulatory networks," *BioSystems*, vol. 98, no. 3, pp. 127–136, 2009.

[28] D. Andre, F. H. Bennett III, and J. R. Koza, "Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem," in *Proceedings of the 1st Annual Conference on Genetic Programming*, pp. 3–11, MIT Press, Stanford University, Palo Alto, Calif, USA, 1996.

[29] C. Jacob, *Illustrating Evolutionary Computation with Mathematica*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[30] V. Sarpe, A. Esmaeili, I. Yazdanbod, T. Kubik, M. Richter, and C. Jacob, "Parametric evolution of a bacterial signalling system formalized by membrane computing," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, IEEE Press, Barcelona, Spain, July 2010.

[31] N. Minar, R. Burkhart, C. Langton, and M. Askenazi, "The swarm simulation system: a toolkit for building multi-agent simulations," Tech. Rep., Santa Fe Institute, Santa Fe, NM, USA, 1996.

[32] S. Rasmussen, N. A. Baas, B. Mayer, M. Nilsson, and M. W. Olesen, "Ansatz for dynamical hierarchies," *Artificial Life*, vol. 7, no. 4, pp. 329–353, 2001.

[33] A. Dorin and J. McCormack, "Self-assembling dynamical hierarchies," in *Proceedings of the 8th International Conference on Artificial Life*, p. 423, 2003.

[34] J. L. Dessalles and D. Phan, "Emergence in multi-agent systems: cognitive hierarchy, detection, and complexity reduction part I: methodological issues," *Artificial Economics*, vol. 564, pp. 147–160, 2006.

[35] J. Denzinger and J. Hamdan, "Improving observation-based modeling of other agents using tentative stereotyping and compactification through kd-tree structuring," *Web Intelligence and Agent Systems*, vol. 4, no. 3, pp. 255–270, 2006.

[36] C. Lavelle, H. Berry, G. Beslon et al., "From Molecules to organisms: towards multiscale integrated models of biological systems," *Theoretical Biology Insights*, vol. 1, pp. 13–22, 2008.

[37] S. von Mammen and C. Jacob, "Swarming for games: immersion in complex systems," in *Proceedings of the Applications of Evolutionary Computing Part II*, Lecture Notes in Computer Science, pp. 293–302, Springer, Tübingen, Germany, 2009.

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

[39] S. Von Mammen, C. Jacob, and G. Kókai, "Evolving swarms that build 3D structures," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 1434–1441, IEEE Press, Edinburgh, UK, September 2005.

[40] S. von Mammen, *Evolving Artificial Constructive Swarms—Experimental Models and Methodologies*, VDM, Saarbrucken, Germany, 2008.

[41] I. Burleigh, "Vigo::3d: a framework for simulating and visualizing of threedimensional scenes," October 2008, http://vigo.sourceforge.net/docs/.

[42] S. von Mammen and C. Jacob, "The spatiality of swarms—quantitative analysis of dynamic interaction networks," in *Proceedings of the Artificial Life XI*, pp. 662–669, MIT Press, Winchester, UK, 2008.

[43] A. J. Lotka, "Contribution to the theory of periodic reactions," *Journal of Physical Chemistry*, vol. 14, no. 3, pp. 271–274, 1910.

[44] V. Volterra, "Variazioni e fluttuazioni del numero d'individui in specie animali conviventi," *Atti della R. Accademia Nazionale dei Lincei. Memorie della Classe di Scienze Fisiche*, vol. 2, pp. 31–113, 1926.

[45] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Proceedings of the 1st IEEE International Conference on Data Mining (ICDM '01)*, pp. 289–296, IEEE Computer Society, Washington, DC, USA, December 2001.

[46] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[47] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proceedings of the AAAI-94 Worshop on Knowledge Discovery in Databases (KDD '94)*, pp. 359–370, 1994.

[48] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.

[49] F. Azuajeu, "Computational discrete models of tissue growth and regeneration," *Briefings in Bioinformatics*, vol. 12, no. 1, Article ID bbq017, pp. 64–77, 2011.

[50] S. von Mammen, J.-P. Steghöfer, J. Denzinger, and C. Jacob, "Self-organized middle-out abstraction," in *Self-Organizing Systems*, C. Bettstetter and C. Gershenson, Eds., Lecture Notes in Computer Science, pp. 26–31, Springer, Karslruhe, Germany, 6557 edition, 2011.

[51] A. S. Shirazi, S. von Mammen, and C. Jacob, "Adaptive modularization of the MAPK signaling pathway using the multiagent paradigm," in *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature: Part II (PPSN '10)*, vol. 6239, pp. 401–410, IEEE Press, Department of Computer Science, Faculty of Science, University of Calgary, Canada, September 2010.

[52] C. Y. F. Huang and J. E. Ferrell, "Ultrasensitivity in the mitogen-activated protein kinase cascade," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 93, no. 19, pp. 10078–10083, 1996.

[53] B. N. Kholodenko, "Negative feedback and ultrasensitivity can bring about oscillations in the mitogen-activated protein kinase cascades," *European Journal of Biochemistry*, vol. 267, no. 6, pp. 1583–1588, 2000.

[54] A. S. Shirazi, S. von Mammen, and C. Jacob, "Hierarchical self-organized learning in agent-based modeling of the MAPK signaling pathway," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '11)*, pp. 2245–2251, June 2011.

[55] J. B. MacQueen, "Some methods for classification and analysis of MultiVariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, L. M. Le Cam and J. Neyman, Eds., vol. 1, pp. 281–297, University of California Press, 1967.