A Graph-based Developmental Swarm Representation & Algorithm

Sebastian von Mammen[1], David Phillips[1], Timothy Davison[1], and Christian Jacob[1,2]

[1] Dept. of Computer Science, University of Calgary, Canada
[2] Dept. of Biochemistry and Molecular Biology, University of Calgary, Canada

Abstract. Modelling natural processes requires the implementation of an expressive representation of the involved entities and their interactions. We present *swarm graph grammars* (SGGs) as a bio-inspired modelling framework that integrates aspects of formal grammars, graph-based representation and multi-agent simulation. In SGGs, the substitution of subgraphs that represent locally defined agent interactions drive the computational process of the simulation. The generative character of formal grammars is translated into an agent's *metabolic* interactions, i.e. creating or removing agents from the system. Utilizing graphs to describe interactions and relationships between pairs or sets of agents offers an easily accessible way of modelling biological phenomena. Property graphs emerge through the application of local interaction rules; we use these graphs to capture various aspects of the interaction dynamics at any given step of a simulation.

1 Introduction

We are interested in modelling complex biological systems at various levels of scale, i.e. from the biomolecular level [33] to cells [13] to systems [15], etc. Different levels of resolution often require different computational techniques, such as differential equation solvers to compute physics or fluid dynamics, or engines that execute high-level agent behaviours that implement rich interaction policies and complex strategies [39]. Independent of the specific computational approaches that drive the simulation processes, they all rely on state changes, the principle of digital computation. Furthermore, a system's state determines the introduced changes, probabilistically or deterministically. This idea is emphasized in numerous computational representations such as Markov chains [1] or cellular automata [38]. The state of a system is generally understood as the states of all its subsystems including their interrelations. Consequently, states and relations are interchangeable terms that provide the condition for change, or the antecedent for a consequent in a simple *If-then rule*. A set of probabilistic rules (like in Markov chain systems) works well to represent the activities of decentralized, self-organizing swarm agents [3–5, 15, 14], including swarm-based developmental systems [25, 24].

Rule-based swarm systems seem to be a good fit to capture biological models. However, there are several hurdles that make it hard to deploy swarm models in fields outside of computer science. (1) The predicates and actions that drive the simulations—e.g. the detection of a chemical signal or the deposition of a particle—depend on the modelling domains and are usually re-implemented for different experiments. Still, many of these operations can be abstracted, parametrically adjusted and reused in different contexts. The integration of these operations into a rule-based formalism also makes it possible to utilize functionality from various computational engines such as physics engines or general differential equation solvers within one modelling framework. (2) Depending on the degree of specificity of a rule's condition and its associated actions, a theoretically simple interaction can result in an over-complicated representation. A graphical description of the predicates and the associated actions can amend this issue. (3) As swarm simulations often exhibit complex behaviours, little details—for example the order of execution and the discretization steps in a simulation—can greatly influence the outcome. Therefore, we think it is crucial to design models based on a unified algorithmic scheme.

We have devised *swarm graph grammars* (SGGs) to alleviate some of the challenges discussed above. SGGs provide a graphical, rule-based description language to specify swarm agents and a generalized algorithmic framework for the simulation of complex systems. Fundamental operations such as creation or deletion of programmatic objects, as provided by formal grammars, are part of the SGG syntax. Through SGGs we can capture (metabolic) functions at multiple biological scales, i.e. the processes of secretion and diffusion [37], or consumption/removal and production/construction [20], respectively. As a consequence of the graph-based syntax, SGGs capture the simulation in a global graph at each computational step. Thereby, the continuous re-shaping of an interaction topology of a dynamic system is traced and interdependencies that emerge over the course of a simulation are graphically represented.

The remainder of this paper is organized as follows. In Section 2, immediately relevant work in the respective areas of research is presented. Section 3 details swarm graph grammars (SGGs) and their constituents, i.e. swarm individuals, graph grammatical rules, and a general SGG algorithm. Section 4 shows how the SGG formalism is applied in a step by step manner to retrace a simple boid simulation, wasp nest construction, and directed cell growth and proliferation. We conclude with a summary and an outlook on possible future work.

2 Related Work

Cellular automata (CAs) can be considered the first computational developmental models [28]. CAs revolve around state-based interactions of individuals given a fixed interaction topology. However, in the emerging discipline of computational developmental systems, the focus shifted towards constructive expressiveness and thus overshadowed the idea of individual-based modelling. In this

section, we briefly review the emergence of CDMs and demonstrate their reunion with agent-based modelling.

2.1 Complex CDMs

Giavitto et al. summarize several approaches to computational developmental models [10]. The most simple ones are considered to be *dynamical systems* with sets of state variables determining their global states. *Structured dynamical systems* are more complex; they are dynamic systems that can be divided into subsystems. Finally, there are *dynamical systems with dynamical structures*, abbreviated as $(DS)^2$-systems, for instance a "developing multi-cellular organism" [12]. In addition, Giavitto et al. describe developmental models as tuples of topology and formalism. L-systems [22], for instance, describe how individual elements of sequences are substituted in parallel. *Group-based data fields* (GBF) [34], on the other hand, operate on sets of units that are connected with a homogeneous, fixed topology not unlike cellular automata [28]. *Map L-systems* [2], similar to *random boolean networks* (RBNs) [16], promote combinatorial topologies on the interacting, or growing, data structures. There are also formalisms that explicitly integrate the topology of the modelled systems, such as *membrane computing* (MC), or *P systems* [29]. P systems draw their inspiration from membrane structures of cells, neural cells and tissues. In a more generalized fashion, *graph grammars* [9] are a means to integrate topological information into any kind of developmental model. Examples are *multiscale tree graphs* (MTGs) and the *modèle géneral de simulation* (MGS) that represent changes of *topological collections* of units by *transformation paths* on a symbolic notation [11].

2.2 Graph-based CDMs

Kniemeyer et al. have developed *relational growth grammars* (RGGs) which promise, like MGS, to be a universally applicable representation of CDMs [19]. They use RGGs as extensions of parametric L-systems with object-oriented, rule-based, procedural features. In fact, modelling CDMs by graph grammars, like in RGGs, allows for the expression of all developmental data structures commonly used in the computational sciences: multisets, strings, axial trees, and relational structures (edge-labeled directed graphs). Graph grammar-based CDMs can therefore be considered as a universal modelling language, able to simulate standard L-systems, artificial chemistries and ecological systems alike. Kniemeyer et al. successfully applied the RGG model to grow multi-scale models of plants integrating their structure and function [18], and, recently, to grow architectural models [17]. They also suggested that RGGs could support *agent-based* modelling—by interpreting nodes as agents, edges as inter-agent relations, and by driving their interactions through sub-graph substitutions [21].

Almost 20 years before Kniemeyer presented RGGs, Culik et al. had extended L-systems with the means to describe plants through graph structures and their growth through graph grammatical substitutions, which were later on referred to as *graph L-systems* [6]. Shortly afterwards, Nagl investigated the relationship

between graph grammars and graph L-systems, concluding that graph grammars can be reduced to graph L-systems and vice versa [27]: identical graphs can be achieved by either sequential graph grammar productions or by parallel subgraph substitutions as realized in graph L-systems. About another decade later, Lindenmayer argued that relying on maps instead of graphs bears many advantages, e.g. a clear method for mapping between the abstract representation and the natural, growing structures and better performance due to the avoidance of transformations of the representations [23].

Recently, Tomita et al. have presented *graph rewriting automata* [36], in which lattice-based CAs evolve into complex networks through the application of production rules that change local connectivities. Sayama et al. went one step further and considered the local states of a CA to inform the development of *generative network automata* (GNA) [32].

2.3 Swarm-based CDMs

Developmental systems can be simulated by means of agent-based, decentralized models that incorporate diffusion of molecular signals paired with particular protein or cell behaviours [31]. A generic formalism for agent-based models was provided by Denzinger et al. [7, 8] in which an agent is represented as a quadruple $Ag = (Sit, Act, Dat, f_{Ag})$. An agent Ag can find itself in any of the situations expressed in Sit. It can perform the actions described by the set Act. Its internal data areas, i.e. local variables or memory cells, are determined by the set of possible values Dat. Based on the perceived situation and its internal data values, the agent determines the next action through a decision function $f_{Ag} : Sit \times Dat \rightarrow Act$. This representation is very expressive and follows the descriptive methodology of many natural sciences in which the principle of local cause and effect leads to associated emergent phenomena of interest.

Based on these ideas, we have introduced *swarm grammars* (SGs) that merged L-systems with an agent-based modelling approach [24]. In swarm grammars, decentralized swarm agents, or individuals, have the ability to perceive and act in accordance with Denzinger et al.'s agent definition. In particular, SG individuals can react to their local environment, differentiate, reproduce, and create structures by depositing construction elements. Albeit the fact that SGs merge several instrumental biological concepts of developmental, non-linear interaction systems, they do not provide a unified, easy-to-use representation and algorithm that allows for systematic deployment in other scientific disciplines (as discussed in Section 1).

3 Swarm Graph Grammars

We present *swarm graph grammars* as a unified modelling and simulation framework for swarm-based systems that addresses the challenges outlined in Section 1, and provides a unified, graphical, rule-based modelling language for swarm individuals and a generalized simulation algorithm. The graphical description renders model dynamics more tangible and translates local interactions into global,

continuously changing interaction networks. We believe that investigations into the development of these networks, in turn, could reveal quantifiable measures about *emergent* global phenomena. We address Lindenmayer's concerns about the inefficiency of graph-based CDMs by a minimalist subgraph matching procedure that only considers star networks of depth 1 around the corresponding, active reference agent.

3.1 Representation

An SGG agent's behaviour is described by a set of rules (Figure 1). Each rule tests a set of predicates (solid edges on the left-hand side) and executes a set of actions (dashed edges on the right-hand side) in respect to the acting agent itself (reference node) or other agents. Nodes represent individual agents or sets of agents. In Figure 1, the acting agent is displayed as an orange node with a black border. Other agents or agent groups are depicted as grey nodes. The application of the rule is associated with a frequency and a probability. Sets of predicates can attempt to identify an arbitrary number of agents. The relative location, i.e. the two-dimensional coordinates, of the node on the left-hand side of the rule is matched with its appearance on the right-hand side of the rule. If a node does not reappear on the right-hand side, it implies that its corresponding agent has been removed. If a node appears at a location that is unoccupied on the left-hand side, a new node is created. Figure 1 shows an example rule: It is applied with a probability of $p = 0.3$ at every fourth time step ($\Delta t = 4$). One (arbitrarily chosen) node that fulfills *predicateX* and *predicateY* is affected by *actionJ* and *actionK*. Also note that a new node is created and is initialized in this rule for which no reference had existed before. In case there are at least 6 nodes that fulfill *predicateZ*, they will all be removed.

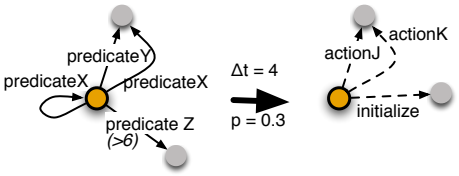

Fig. 1. An SGG rule that queries the reference node itself, other individuals and sets of interaction candidates, to interact with them, delete some and to initialize a new node.

3.2 Algorithm

A swarm graph grammar $SGG = (\mathcal{I}, \Xi, \mathcal{G}_{predicate}, \mathcal{G}_{action}, P)$ is a quintuple, where $\mathcal{I}$ describes a set of individuals relying on rules and properties as explained in the previous section. At the beginning of the simulation, a set Ξ of axioms, in the form of initialization algorithms, is executed by (1) selecting and expressing individuals from $\mathcal{I}$, and (2) by assigning initial states to the newly created individuals. For a homogeneous swarm of nest-constructing wasps[3], for instance, $\mathcal{I}$ only has to comprise a single agent description. Having created a sufficient number of wasp agents, the axioms would assign contextual information such as an initial location to the individuals. In the main loop of the swarm

[3] See Section 4.2 for details.

graph grammar algorithm (Algorithm 1) two graphs $G_{predicate} \in \mathcal{G}_{predicate}$ and $G_{action} \in \mathcal{G}_{action}$ are subsequently created that merge the triggered predicates and corresponding actions of the individuals' local rules. $\mathcal{G}_{predicate}$ represents the set of possible graphs of individuals interconnected through predicates. $\mathcal{G}_{action}$ hosts all possible action graphs. Chains of relations among sets of swarm individuals create semantic topologies for global graph structures that describe the situational context or activity in the SGG system. Executing the actions of G_{action} yields the next simulation state after a policy P is applied to resolve possibly arising computational conflicts[4]. Thus, the alternating update of the graph instances $G_{predicate}$ and G_{action} based on the swarm individuals' behaviours drives the SGG simulation (Figure 2).

Algorithm 1 Swarm Graph Grammar: Main Loop

Require: $G_{predicate}$, optional: P
Ensure: alternating computation of $G_{predicate}$ and G_{action}
  repeat
    compute predicative graph $G_{predicate}$
    compute action graph G_{action} based on $G_{predicate}$
    apply order policy P to G_{action}
    execute ordered actions of G_{action}
  until simulation is terminated

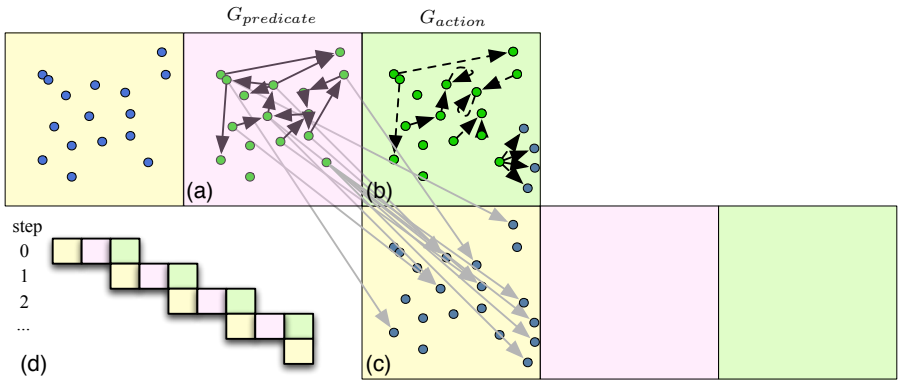

Fig. 2. Subsequent computation of (a) $G_{predicate}$ and (b) G_{action} yield (c) the next simulation state. The grey arrows from (a) to (c) relate nodes to their contextual impact. (d) The simulation process is shown as a computation pipeline.

[4] The implementation of an efficient conflict policy P is often difficult and its execution can be computationally expensive.

4 Swarm Graph Grammars in Action

In this section we present three computational models realized with the SGG framework. We retrace (1) a simple boids simulation [30], (2) the stigmergic construction behaviour of the Chartergus wasp [35], and (3) cell proliferation induced by a set of growth factors.

4.1 Boids

In order to specify a standard boid flocking simulation [30], we use a swarm graph grammar $SGG_{boid} = (\mathcal{I}_{boid}, \Xi_{boid}, \mathcal{G}^{boid}_{predicate}, \mathcal{G}^{boid}_{action}, P_{boid})$. The sole individual $i_{boid} \in \mathcal{I}_{boid}$ contains several weights for flocking urges, parameters to determine a field of perception, as well as boundaries for the maximal flight acceleration max_{accel} and velocity max_{vel}. Ξ_{boid} generates a homogeneous set of swarm individuals that are initialized with a random position $\overrightarrow{p}$ and velocity $\overrightarrow{v}$. As no interaction conflicts arise, the policy P is empty.

Boids rely on two behavioural rules shown in Figure 3. The *movement* rule continuously updates a swarm individual's position in accordance with its velocity. The *acceleration* rule, substitutes the predicate $sees(u, v)$ with the action $accelerate(u, v)$.
The predicate considers the reference node's location, orientation and perceptional field to select a set of interaction partners in accordance with their respective locations. The action also considers the difference between u's and v's states, including their locations and velocities, and accelerates u accordingly. For example, u accelerates towards v's location and it aligns its flight direction. In the example displayed in Figure 4, the boid agents form a cluster over time which is also reflected by increasingly connected interaction graphs.

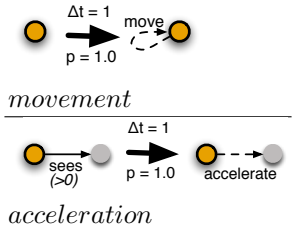

Fig. 3. Two rules to describe a boid agent's interaction behaviour.

4.2 Stigmergic Construction

Theraulaz et al. have translated the nest construction processes of Chartergus wasps into individual behavioural rules [35]. The rules in Figure 5 closely retrace this behaviour[5]. The predicates *around*, *below* and *occupied* test the immediate surroundings of the wasp to trigger comb construction in the remaining rules. Hereby, previously deposited combs of two different types ($Comb1$, $Comb2$, or $Comb*$ for both) trigger the next *placement* actions. In addition, a *movement* rule as seen in Figure 3 moves an individual unconditionally to a random location in the simulation space. Figure 6 shows the development in agent space and correlates the activating (red) and the constructed combs (green). The rule deployment is shown in a series of interaction graphs $G_{interaction} = G_{predicate} \cup G_{action}$.

[5] The lattice-based matrix representation provided in [35] was translated into predicates that test the corresponding spatial relationships.

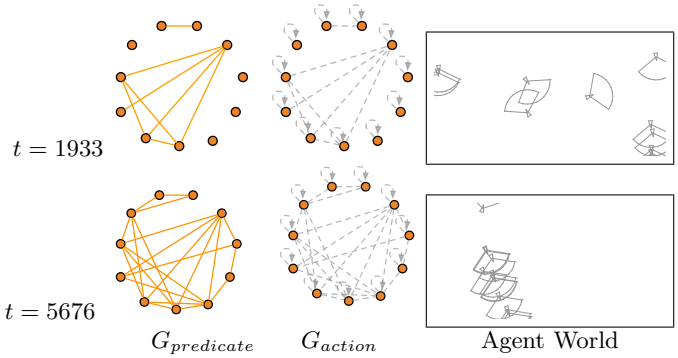

Fig. 4. Two sets of graphs $G_{predicate}$, G_{action} and a visualization of the agent space show a clustering process in a SGG-driven boid simulation. The boid renderings—triangles oriented towards their velocity with a conic field of perception—partially overlap due to their strong alignment urge.

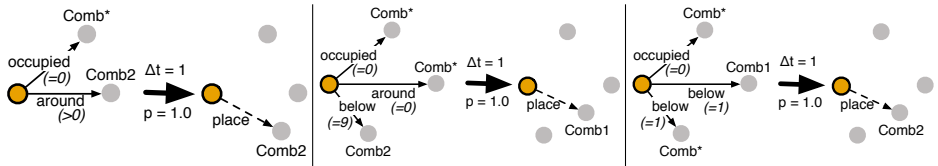

Fig. 5. SGG rules that retrace the construction behaviour by the Chartergus wasp as described in [35].

4.3 Swarm Development

Signalling factors determine the rate of cell proliferation which influence specific morphological developments [26]. The rules in Figure 7 configure cells which *grow* until they reach maturity (predicates *not mature* and *mature*). Mature cells that are *close to* a *Growth Factor* increase their internal *mitogen* concentration which in turn instigates proliferation (modelled as *reset* of the acting cell and *initialization* of a second cell).

Figure 8 shows screenshots of the simulation. Tissue cells within the vicinity of a signalling molecule start proliferating. Collision resolution through an embedded physics engine allows the cells to assemble[6]. The emerging protuberance is slanted to the right in accordance to the initial distribution of signalling molecules. However, it is surprisingly symmetrical still, which could result from a lack of simulated cell polarization.

[6] In the given experiment we rely on the Bullet physics engine, http://bulletphysics.org

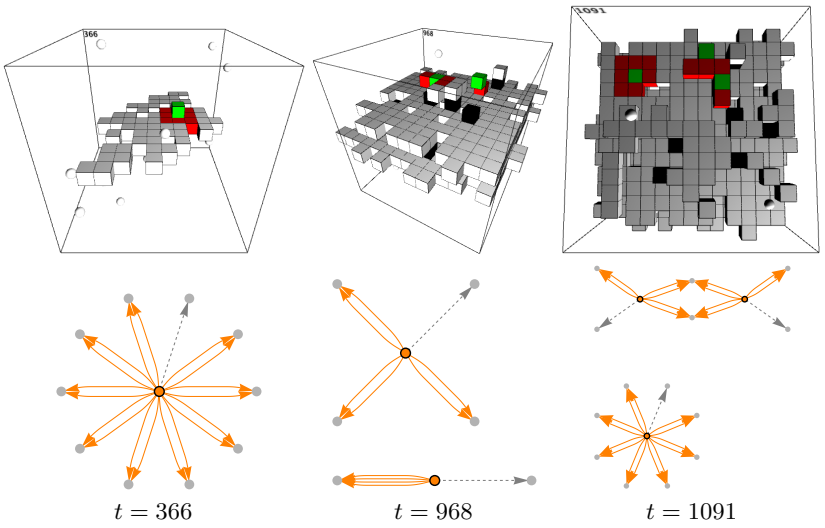

Fig. 6. Agent space and the corresponding interaction graphs of a wasp-inspired construction process (grey dashed arrows indicate actions, orange ones predicates). At $t = 366$ a floor template is constructed (rule (c) in Fig. 5). At $t = 968$ the construction of a new floor is started (rule (d) in Fig. 5). At $t = 1091$ two floor extensions are performed by different wasp agents triggered by the same subset of combs.

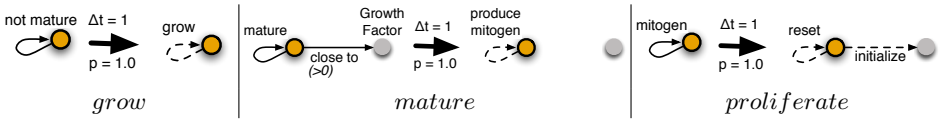

Fig. 7. Three rules to describe a simple developmental process model.

5 Summary and Future Work

Swarm graph grammars are a modelling and simulation framework that provides a universal graph-based representation for swarm-based developmental systems. Besides metabolic operations, i.e. the creation or removal of agents, the semantics of agent relations are not part of the framework. The agents' abilities have to be implemented in the form of predicates and actions. The agents' rule sets (behaviours) drive the simulation processes and they are immediately reflected in the interaction graph of a simulation. As examples, we used SGGs to simulate boid flocking, stigmergic wasp nest construction, and growth and proliferation in cellular morphological processes.

We are currently working on several aspects to improve and harness the utilization of swarm graph grammars. The application of the framework has led to many refinements in respect to the formalism and the algorithm. However,

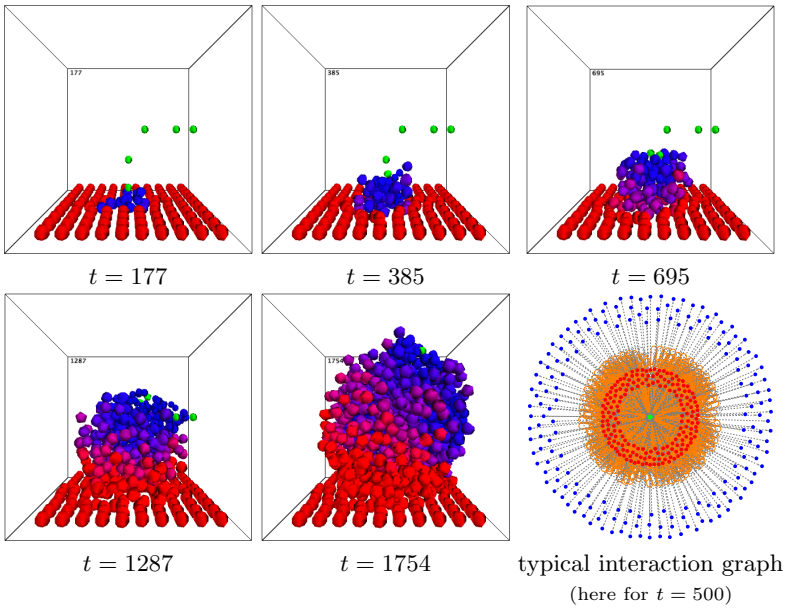

Fig. 8. The proliferation of mature cells (blue: not mature; red: mature) is dependent on the proximity to growth factors (green). At any time of the simulation, large numbers of agents are informed by growth factors leading to typically dense but homogeneous interaction graphs.

in order to render modelling with SGGs accessible, especially to non computer scientists, we need to collect feedback from interdisciplinary modellers about the shortcomings of the representation, e.g. regarding its visualization, terminology and logic. In this paper, we have touched upon matching local agent rules with a simulation's emerging interaction graphs. We deem this a very promising approach to analyze emergent phenomena in simulations on the one hand, and to create complex interaction processes with dynamic interaction topologies on the other hand. Accordingly, systematic investigations have to be started. We are also working on a slight modification of the SGG framework so that nodes can encapsulate children and thereby computational or spatial hierarchies can be built. This would allow for hierarchical modelling as in P systems [29].

Acknowledgements

Support for this research was provided by the Undergraduate Medical Eduction program of the University of Calgary. We would like to thank Jörg Denzinger for his invaluable advice on multi-agent systems and Heather Jamniczky for her feedback on biological developmental systems.

References

1. Allen, L.J.S.: An Introduction to Stochastic Processes with Applications to Biology. Pearson Education, Upper Saddle River, NJ (2003)
2. de Boer, M.J.M., de Does, M.: The relationship between cell division pattern and global shape of young fern gametophytes. i. a model study. Botanical Gazette 151(4), 423–434 (1990)
3. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Santa Fe Institute Studies in the Sciences of Complexity, Oxford University Press, New York (1999)
4. Burleigh, I., Suen, G., Jacob, C.: Dna in action! a 3d swarm-based model of a gene regulatory system. In: ACAL 2003, First Australian Conference on Artificial Life. Canberra, Australia (2003)
5. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-Organization in Biological Systems. Princeton Studies in Complexity, Princeton University Press, Princeton (2003)
6. Culik, K., Lindenmayer, A.: Parallel graph generating and graph recurrence systems for multicellular development. International Journal of General Systems 3(1), 53–66 (1976)
7. Denzinger, J., Kordt, M.: Evolutionary on-line learning of cooperative behavior with situation-action-pairs. In: ICMAS. pp. 103–110. IEEE Computer Society (2000)
8. Denzinger, J., Winder, C.: Combining coaching and learning to create cooperative character behavior. In: CIG. IEEE (2005)
9. Ehrig, H., Kreowski, H.J., Montanari, U., Rosenberg, G. (eds.): Handbook of Graph Grammars and Computing by Fraph Transformation, Concurrency, Parallelism, and Distribution, vol. 3. World Scientific Publishing, Singapore (1999)
10. Giavitto, J.L., Godin, C., Michel, O., Prusinkiewicz, P.: Modelling and Simulation of biological processes in the context of genomics, chap. Computational Models for Integrative and Developmental Biology, pp. 12–17. Hermes (July 2002)
11. Giavitto, J.L., Michel, O.: Data structure as topological spaces. Unconventional Models of Computation pp. 137–150 (2002)
12. Giavitto, J.L., Michel, O.: Modeling the topological organization of cellular processes. Biosystems 70(2), 149–163 (2003)
13. Jacob, C., Burleigh, I.: Biomolecular swarms: An agent-based model of the lactose operon. Natural Computing 3(4), 361–376 (December 2004)
14. Jacob, C., Hushlak, G., Boyd, J., Nuytten, P., Sayles, M., Pilat, M.: Swarmart: Interactive art from swarm intelligence. Leonardo 40(3) (2007)
15. Jacob, C., Steil, S., Bergmann, K.: The swarming body: Simulating the decentralized defenses of immunity. In: Artificial Immune Systems, ICARIS 2006, 5th International Conference. Springer, Oeiras, Portugal (September 2006)
16. Kauffman, S.: At Home in the Universe: The Search for the Laws of Self-Organization and Complexity. Oxford University Press (1995)
17. Kniemeyer, O., Barczik, G., Hemmerling, R., Kurth, W.: Relational Growth Grammars—A Parallel Graph Transformation Approach with Applications in Biology and Architecture. In: Applications of Graph Transformations with Industrial Relevance. pp. 152–167. Springer-Verlag, Berlin, Heidelberg (2008)
18. Kniemeyer, O., Buck-Sorlin, G., Kurth, W.: Groimp as a platform for functional-structural modelling of plants. In: Vos, J., Marcelis, L.F.M., deVisser, P.H.B., Struik, P.C., Evers, J.B. (eds.) Functional-Structural Plant Modelling in Crop Production. pp. 43–52. Springer (March 2006)

19. Kniemeyer, O., Buck-Sorlin, G.H., Kurth, W.: A graph grammar approach to artificial life. Artificial Life 10(4), 413–431 (2004)
20. Kumar, S., Bentley, P. (eds.): On Growth, Form and Computers. Elsevier Academic Press, London (2003)
21. Kurth, W., Buck-Sorlin, G., Kniemeyer, O.: Relationale wachstumsgrammatiken: Ein formalismus zur spezifikation multiskalierter struktur-funktions-modelle von pflanzen. In: Modellierung pflanzlicher Systeme aus historischer und aktueller Sicht. Landwirtschaft, vol. 7, pp. 36–45. Landesamtes für Verbraucherschutz, Landwirtschaft und Flurneuordnung Brandenburg (2006)
22. Lindenmayer, A.: Developmental systems without cellular interactions, their languages and grammars. Journal of Theoretical Biology 30(3), 455–484 (1971)
23. Lindenmayer, A.: An introduction to parallel map generating systems. Graph-Grammars and Their Application to Computer Science pp. 27–40 (1987)
24. von Mammen, S., Jacob, C.: The evolution of swarm grammars: Growing trees, crafting art and bottom-up design. IEEE Computational Intelligence Magazine (August 2009)
25. von Mammen, S., Jacob, C., Kókai, G.: Evolving swarms that build 3d structures. In: CEC 2005, IEEE Congress on Evolutionary Computation. pp. 1434–1441. IEEE Press, Edinburgh, UK (2005)
26. Megason, S.G., McMahon, A.P.: A mitogen gradient of dorsal midline wnts organizes growth in the cns. Development 129, 2087–2098 (May 2002)
27. Nagl, M.: On the relation between graph grammars and graph l-systems. Fundamentals of Computation Theory pp. 142–151 (1977)
28. von Neumann, J., Burks, A.W.: Theory of self-reproducing automata. University of Illinois Press, Urbana and London (1966)
29. Paun, G., Rozenberg, G.: A guide to membrane computing. Theoretical Computer Science 287(1), 73–100 (9 2002/9/25)
30. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
31. Salazar-Ciudad, I.: Tooth Morphogenesis in vivo, in vitro, and in silico. Current Topics in Developmental Biology 81, 342 (2008)
32. Sayama, H., Laramee, C.: Generative network automata: A generalized framework for modeling adaptive network dynamics using graph rewritings. Adaptive Networks pp. 311–332 (2009)
33. Schlick, T.: Molecular Modeling and Simulation: an interdisciplinary guide, Interdisciplinary Applied Mathematics, vol. 21. Springer-Verlag, New York (2002)
34. Spicher, A., Michel, O., Giavitto, J.L.: A topological framework for the specification and the simulation of discrete dynamical systems. Cellular Automata pp. 238–247 (2004)
35. Theraulaz, G., Bonabeau, E.: Modelling the collective building of complex architectures in social insects with lattice swarms. Journal of Theoretical Biology 177(4), 381–400 (1995)
36. Tomita, K., Kurokawa, H., Murata, S.: Graph-rewriting automata as a natural extension of cellular automata. Adaptive Networks pp. 291–309 (2009)
37. Walker, D.C., Southgate, J.: The virtual cell–a candidate co-ordinator for 'middle-out' modelling of biological systems. Briefings in Bioinformatics 10(4), 450–461 (7 2009)
38. Wolfram, S.: A new kind of science. Wolfram Media Inc., Champaign, Ilinois, US, United States (2002)
39. Wooldridge, M.: An Introduction to MultiAgent Systems. John Wiley and Sons, Chichester, England (February 2002)