Evolutionary Synthesis of Collective Behavior

David Kriesel
Department of Computer Science I
Römerstr. 164
D-53117 Bonn, Germany
Tel.: +49-177-4234223
Fax: +49-228-7342321
eMail: mail@dkriesel.com

Abstract

In the present position paper, I explore biologically-inspired computational processes that allow complex high-level collective behaviors to arise from low-level artificial agents (*swarmers*) – automatically. In contrast to similar projects, I seek elimination of technical constraints that narrow the free development of biology-analogous behavioral patterns. The result of such swarm evolutions is a fascinating variety of biological, yet completely transparent, analyzable behavior. Results include the spontaneous evolution of an exploration strategy that recently has been mathematically proven to be the optimal one under the conditions given.

The work (which is part of my diploma thesis [11]) originally contributes to the field of synthetic biology and the goal was to make evolution milestones in biological swarm collaboration visible. However, I feel that high-level behavior generation techniques can be migrated to the field of collaborative security and suggest approaches to do so.

1 Introduction: Collective Behavioral Patterns in Nature and Synthesis

Since the beginning of life on earth, nature has developed a vast variety of collective behaviors. These behaviors include, for instance, thousands and thousands of fireflies flashing synchronously in summertime, bathing entire grasslands in sallow light. They include as well complex structures built by social insects that appear technically mature and allow for storage or even production of food, sophisticated ventilation mechanisms, breeding and defense against predators. Some of those structures are built to such perfection and precision that we have just started to learn how they actually work [4, 21].

Collective behaviors observed in biology inspired a variety of research work in recent times. Pursuing this research, it surprisingly turned out that the complexity of the observed collective behavior is in stark contrast to the relative simplicity of participating individuals. However, the appearance of complex phenomena that arise from local interactions of simple parts with each other and the environment is not limited to the fauna. Works such as [6, 14] observe similar phenomena in the fields of physics and chemistry.

Enabled by continuously growing processing power of up-to-date computers, collective behaviors could be synthesized. From decentralized control of robot and agent swarms [8, 19], over routing of traffic in communication networks to idealized synthesized collective behavior as a metaheuristic to solve complex optimization problems [3, 18] the possibilities seem to be unlimited.

The ever further increasing processing power finally made possible the synthesis of collective behavior in an evolutionary way [12, 22]. Here, mechanisms of natural evolution are used to synthesize collective behavior, which opens new possibilities for research. However, the development of natural behaviors via synthetic evolutions is usually significantly narrowed due to certain technical constraints such as static sensor and actuator parameters in swarmers or too bottom-of-the-line communication methods among them.

In this work, I seek for elimination of some of those constraints[1], trying to evolve complex collective behavioral patterns and make the process of behavioral evolution visible. Particularly, I create behavioral patterns analog to the ones found in nature in a swarm of insect-like swarmers.

[1] I, for example, enable evolution of parts of sensors, actuators, morphology as well as different forms of direct and indirect communication and make use of extensive simulations obeying physical laws.

2 Ingredients for Evolutionary Synthesis of Collective Behavior

To evolve such collective behavior, I make use of the following methods and algorithms:

Parametrizable swarmer blueprint. I defined a blueprint swarmer, whose controllers, sensors[2], actuators[3], and certain internal values are heavily defined by a set of parameters (*genome*). A swarmer's controller maps sensor perceptions to actuator outputs that enable a swarmer to make use of given capabilities I won't discuss in detail here. As controllers, I use recurrent neural networks with freely evolvable topology, a data processing structure that is inspired by biological sets of neurons [7, 10]. Advantages of the distributed, parallel information processing by lots of simple neurons include the lesion tolerance and learning aptitude through simple operators defined on the neural network. Disadvantages include the difficult analysis of what a neural network does, once trained. There exists a variety of published evolutionary operator packets, simulation frameworks, benchmark tests and theoretical work concerning the evolution of neural networks, see for instance [20, 23].

Artificial evolution. Swarmer genomes should not be crafted, but *evolved*. Evolutionary optimization (short in this paper: *evolution*) is a metaheuristic, which is inspired from natural evolution and widely used for generating adequate solutions for complex nonlinear optimization problems [16]. This brief introduction of evolutionary optimization is not intended to follow a particular school or kind of evolutionary optimization strategies, but to explain on an abstract level, what evolutionary optimization is. In evolving, one has to create an initial random population of genomes (each of which represents a solution candidate for the optimization problem to solve), *evaluate* all of them, and finally let the *fittest* ones act as parent genomes that are crossed and mutated in order to create the next generation of the genome population. This process is repeated until a given amount of time has passed or certain evaluation criteria are fulfilled.

Evaluation by physics simulations. A genome is evaluated by spawning swarmers from it and placing them in a simulated environment. Such environments are of two dimensions, space-continuous and time-discrete, and simulated using a multi agent simulation framework built upon an open physics simulations core [5]. A simulated environment always contains analogons of food, that are necessary for a swarmer to live and reproduce, and more or less difficult to collect (depending on the particular *experiment*). The more reliable a swarmer manages to grow and keep up a population of his species in the environment, and the larger it grows, the higher is the *fitness* of its genome. The fitness function was the average number of swarmers continuously living during an evaluation simulation. Since a swarmer's lifetime is much shorter than an evaluation simulation lasts, swarmers have to maintain their population by reproduction and – depending on the experiments – more or less complex foraging strategies.

Partly automated behavior analysis. Once evolved, swarmer genomes are analyzed in a partly automated way by a test suite, which allows for unveiling capabilities (like pheromone or food dispersal or certain kind of sensor perception ...) that are important for the evolved behavior. Casually spoken, if a capability is important, the performance of the swarm would drop significantly in case the capability is missing. Thereupon, I designed a family of tests to measure *importance* of several swarmer capabilities. Based on the importance knowledge gained, I further analyze evolved swarm behavior.

I am aware of the fact that, instead of fitness evaluations as described above, other swarm evolution systems [2, 17] recently published use a fitness-function-less artificial ecology approach in order to not restrict the behavioral evolution by defining fitness functions. Furthermore, less pre-specified agents can be used for the same reason. I agree with the voices saying that this approach might be more free in terms of free behavioral evolution alone, but in our case, it has two strong disadvantages:

1. It can't be easily mapped to other problem domains (for example, a fitness function can be easily derived of most network epidemics problems, and most of such problems incorporate heavily predefined agents).

2. Evolution of complex behavioral patterns known from biology may also be narrowed if the agents are designed in a too simple and bottom-of-the-line way: It may be just too difficult to evolve behaviors sophisticated enough to solve given problems.

[2] Swarmers can percept other swarmers that are nearby, as well as objects in the environment, like food or pheromone analogons, etc.

[3] Swarmers incorporate many nature-inspired capabilities like moving, steering, dispersal of pheromones or food into the environment, eating food that exists in the environment, communication, reproduction, etc. Discussing them in detail would go beyond the scope of an position paper.

Consequently, I deliberately chose the fitness-function-approach (with the above very generic fitness function) as well as partly predefined agents with a very rich set of nature-inspired capabilities including direct and indirect communication (which however are most configurable through their genome). Nevertheless, my more traditional approach is able to contribute competitive behavioral complexity, as we will see in the experimental results.

In order to evolve manifold behaviors concerning a single experiment, several separate evolutions were performed in each experiment. My evolution framework is capable of thousands of simulation steps a second on a single CPU with possibly thousands of static objects in a simulated world, as well as (in this work) 15 to 50 swarmers. Evaluations were carried out on a network cluster of more than 100 CPUs in parallel, so eliciting many different behaviors each experiment was possible.[4].

3 Example of Experiments and Results

As stated above, swarmers are evolved in environments that vary throughout different experiments. To get an idea of the complexity end efficiency of behaviors evolved, I now want to examine one representative behavioral pattern evolution in one single experiment that actually has been theoretically proven to solve the experiment problem in an optimal way. As the experiment goal was the observation of evolution milestones, first basic skill were evolved, then complex ones. I will explain the experiment results in this order.

In the experiment, swarmers were initially close to a large accumulation of "food" items, which makes them learn quickly how to recognize this food, eat it, and remain close to food they see (fig. 1). Regularly after a fixed number of simulation steps, the next large food accumulation appears in the simulated world – however, it is placed out of the swarm's sight. To further increase a genome's fitness in evolution after learning the basic skills above, those far away food sources obviously have to be exploited.

As a result, the swarmers not only learned the very basic skills mentioned above, but – with the basic ones as groundwork – also several higher level skills widely known from biology.

[4]It is to note in this context, that significantly higher speeds and therefore more complex experiments will be possible in future: The number of PCs was fluctuating during runtime and most of the evolution time the calculation was slowed down to keep the calculating PCs available for daily work. Other changes to the whole framework, like replacing the internal neural networks implementation with the recently finished SNIPE [9] will provide us with significant speed-ups, too. Other plans include using features of the evolutionary computation toolkit ECJ [15].

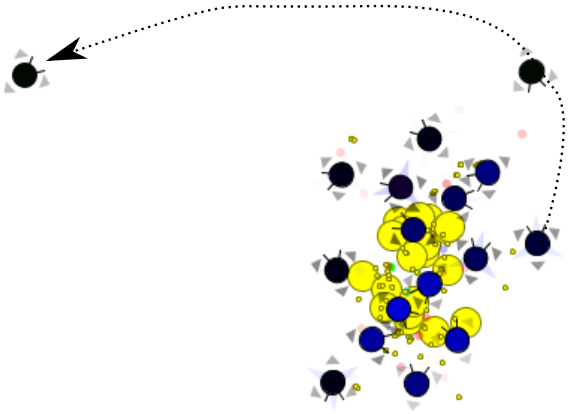

Figure 1: Swarmers (dark, with antennae) aggregating at a food accumulation (yellow items) with two scouts leaving it. The scout's approximate trajectory is marked with a dotted arrow. The sallow red spots mark pheromones dispensed by swarmers. The dark blue filling color of some swarmers as well as little arrows around them mark ways of communication.

Ration existing food: Too much reproduction would result in too quick eating, so that the swarm would starve to death before a new food source appears. Consequently, a swarmer's food rationing strategy is not to reproduce while it senses signals of other swarmers nearby.

Optimal search strategies: Time by time, scouts leave the swarm, searching for new food accumulations. When scouts search for new food accumulations, they do so in a spiral trajectory (fig. 1). This spiral search strategy actually is, as Langetepe has recently proven [13] based on work of Alpern and Gal [1], the optimal searching strategy in 2D spaces given only local environment knowledge. To the best of my knowledge, this is the first time a search strategy known as an optimal one has spontaneously emerged out of such a nature-inspired evolution.

Communication: Both food rationing and scouting are controlled by signals swarmers learned to exchange locally.

Through more than one hundred evolutions, I elicited many different behavioral patterns: For example, after another evolution, a dense group of intensively communicating swarmers moved in a coordinated way towards and around food sources (fig. 2). Pheromones are dispensed around food sources and serve as landmarks.

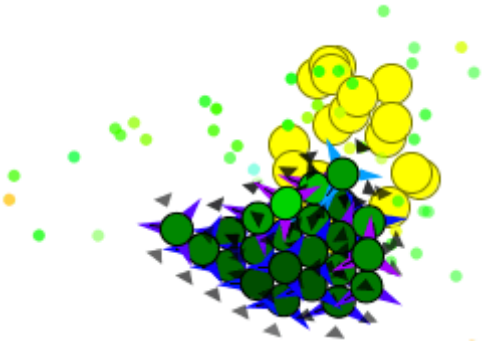

Figure 2: Swarmers moving in a dense group.

4 Multi-Agent Simulations in Computer Security

The results shown above indicate how sophisticated (spiral search) and resource-efficient (rationing) such evolved collaborative behavior can be. In this section, I consequently want to suggest how mechanisms, that elicit life-like behavior as described above in synthetical biology, can be applied in multi-agent security simulations that seek for evolving attack or defense strategies in collaborative security. Possible scenarios include:

Defense strategy evolution: Instead of evolving food foraging strategies in a 2D space obeying physical laws in simulation, one could evolve similar sophisticated "foraging" strategies for viruses or vulnerabilities in communication networks. In doing so, behavior of *network maintenance agents* (that may be even more predefined than the ones above) could be evolved. Such agents are computers, that patrol the network, fight network epidemics and activate resource demanding anti-virus routines or alternatively perform scans for security vulnerabilities like weak passwords or missing security patches. On the one hand, one wants want to use limited number of those agents (as they use expensive resources), but on the other hand, sometimes it may be purposeful to let them multiply fast (when epidemics are taking place, for example). Instead of evolving swarmers, that collaboratively forage for clusters of "food", I suggest to evolve collaborative strategies to forage for *contaminated machines*. Contaminations may appear, and even spread quickly, and here the motivation is to "devour" them as efficiently as possible with respect to network and machine capacities.

Attack strategy evolution: Alternatively, the evolvable agents can model the attacking agents, namely viruses. In this case, strategies to spread as much as possible, without drawing too much attention, have to be evolved. Like network maintenance agents, a virus needs "food" (machines, or free resources on a machine that are not used). Both sophisticated strategies and resource rationing are useful here, too, because spawning too many virus instances will be contra productive (as they cause network monitors to notice them, and launch a counter attack).

To perform such evolutions, the simulation domain obviously would have to be changed from 2D spaces to network-like domains or even abstractions of those. This results in programming effort, but also could decrease the computational effort of the evolutions significantly, thus narrowing the number of computers that is needed to evolve. In both cases, importance tests would gain knowledge of how to further increase efficiency of results, for security tests considered less important can be spared.

5 Conclusions and Outlook

In conclusion, I managed to synthesize the process of complex behavioral evolution, consequently enabling deep behavioral analysis and evolution of sophisticated behavioral patterns hardly possible so far.

There are primarily two research branches I want to further investigate. First, from the synthetic biology point of view I want to generalize from collectives of insect-like beings to collections of cells, thus evolve multicellular organisms. Second, from the algorithmic point of view I want to pursue more theoretical approaches for better analysis of collective and collaborative behavior.

Acknowledgements

I would like to thank Dr. Yaniv Altshuler for advice, discussion and remarks, which were of great value for me.

References

[1] ALPERN, S., AND GAL, S. *The theory of search games and rendezvous*. Kluwer Academic Publishers, 2003.

[2] CHANNON, A. Unbounded evolutionary dynamics in a system of agents that actively process and transform their environment. *Genetic Programming and Evolvable Machines 7*, 3 (2006), 253–281.

[3] DORIGO, M., AND STÜTZLE, T. *Ant Colony Optimization*. MIT Press, 2004.

[4] GARNIER, S., GAUTRAIS, J., AND THERAULAZ, G. The biological principles of swarm intelligence. *Swarm Intelligence 1*, 1 (2007), 3–31.

[5] GLASS, K. Phys2d – the 2d game physics engine in java. Online. http://www.cokeandcode.com/phys2d/.

[6] HAKEN, H., AND WUNDERLIN, A. *Synergetik: eine Einführung: Nichtgleichgewichts-Phasenübergänge und Selbstorganisation in Physik, Chemie und Biologie*. Springer, 1983.

[7] HAYKIN, S. *Neural networks: a comprehensive foundation*. Prentice Hall PTR Upper Saddle River, NJ, USA, 1994.

[8] KRIEGER, M., BILLETER, J., AND KELLER, L. Ant-like task allocation and recruitment in cooperative robots. *Nature (London) 406*, 6799 (2000), 992–995.

[9] KRIESEL, D. Snipe – scalable and generalized neural information processing engine. Online. http://www.dkriesel.com/en/science/snipe.

[10] KRIESEL, D. *A Brief Introduction to Neural Networks*. 2007. available at http://www.dkriesel.com.

[11] KRIESEL, D. Distributed evolution of swarms. Master's thesis, Univ. Bonn, 2009.

[12] KRIESEL, D., CHEUNG, E., SITTI, M., AND LIPSON, H. Beanbag Robotics: Robotic Swarms with 1-DoF Units. In *ANTS Conference* (2008), M. Dorigo et al., Ed., vol. 5217 of *LNCS*, Springer, pp. 267–274.

[13] LANGETEPE, E. On the optimality of spiral search. In *SODA 2010: Proc. 21st Annu. ACM-SIAM Symp. Disc. Algor.* (2010), pp. 1–12.

[14] LAUGHLIN, R., AND REUTER, H. *Abschied von der Weltformel: Die Neuerfindung der Physik*. Piper, 2009.

[15] LUKE, S. ECJ Evolutionary Computation System. Online, 2002. http://cs.gmu.edu/ eclab/projects/ecj/.

[16] MITCHELL, M. *An Introduction to Genetic Algorithms*. Bradford Books, 1996.

[17] PICHLER, P., AND CANAMERO, L. Evolving Morphological and Behavioral Diversity Without Predefined Behavior Primitives. *Artificial Life 11* (2008), 474.

[18] POLI, R., KENNEDY, J., AND BLACKWELL, T. Particle swarm optimization, an Overview. *Swarm Intelligence 1*, 1 (2007), 33–57.

[19] REYNOLDS, C. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), ACM Press New York, NY, USA, pp. 25–34.

[20] STANLEY, K., AND MIIKKULAINEN, R. Evolving neural networks through augmenting topologies. *Evolutionary Computation 10*, 2 (2002), 99–127.

[21] THERAULAZ, G., GAUTRAIS, J., CAMAZINE, S., AND DENEUBOURG, J. The formation of spatial patterns in social insects: from simple behaviours to complex structures. *Philos Transact A Math Phys Eng Sci 361*, 1807 (2003), 1263–82.

[22] TRIANNI, V., AND DORIGO, M. Self-organisation and communication in groups of simulated and physical robots. *Biological Cybernetics 95*, 3 (2006), 213–231.

[23] YAO, X. Evolving artificial neural networks. *Proceedings of the IEEE 87*, 9 (1999), 1423–1447.