Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds.

EMERGENCE BY STRATEGY: FLOCKING BOIDS AND THEIR FITNESS IN RELATION TO MODEL COMPLEXITY

Michael Wagner
Wentong Cai
Michael Harold Lees

School of Computer Engineering
Parallel and Distributed Computing Centre
Nanyang Technological University
50 Nanyang Avenue, SINGAPORE 639798

ABSTRACT

In this paper we aim to examine emergent properties of agent-based models by using evolutionary algorithms. Taking the model of flocking boids as an example, we study and try to understand how the pressure of natural selection towards strategic behavior can result in emergent behavior. Furthermore we investigate how an increase of complexity in the model affects those properties and discover some counter-intuitive behavior in the process.

1 INTRODUCTION

Multi-agent based simulation (ABM) that is designed under a bottom-up approach defines behavior on a low or *microscopic* level, supplying every agent with a certain set of rules to follow. The high level, or *macroscopic* behavior in a system results from an accumulation of low level interactions between the individual agents. The striking feature of macroscopic behavior is its vastly increased complexity in comparison to the microscopic properties that cause them. Emergent behavior resulting from those interactions is usually hard to predict, difficult to understand and - most of all - very challenging to create on purpose.

This stems from the nonlinear relationship between micro- and macroscopic properties. Slight changes in the environment or agent rules may lead to completely different outcomes in the simulation. A good example of this complexity is the Game of Life by Conway, where little differences in the initialization decide whether a population sustains itself, dies out or grows forever. Another one of the most well known examples are the flocking boids, first described and developed by Reynolds (1987), where agents resemble flocking animals such as fish or birds.

Understanding the powerful phenomena of emergence is very desirable as it can provide valuable insight about the real life counterparts of the processes represented in the simulation.

Stonedahl developed an evolutionary approach for finding emergent behavior in multi-agent based environments (Stonedahl and Wilensky 2010). In this case the emergent properties to search for (e.g., converging to flocks or flocking in vee-shapes) are described beforehand, by using their geometric properties. These properties, which mostly describe relationships between movement vectors, are quantifiable. As such they can be used to specify error measures, allowing to detect occurrences of this behavior and determine a metric distance to the ideal form. This approach was successful in repeatably creating convergent and volatile flocking of boids.

Our first goal is to extend the work done by Stonedahl to investigate how variations in model complexity may affect the evolution of model parameters. Increasing the complexity of a model can be helpful by offering a larger set of possible directions to reach a goal. The negative trade-off here is an also increased

size of the search space. We use the boid model but gradually extend its complexity by allowing to have more than one species of boids. The fundamental difference to Stonedahl's approach is that we do not directly specify the emergent properties to evolve, but rather look for strategic behavior in order to achieve an objective. Accomplishing this objective in consequence may cause the boids to show emergent behavior. Secondly we want to analyze the sensitivity of model parameters to the objective. Model parameters often vary in their effects on emergent properties, so the question arises as to which parameters are the most influential ones. This may also provide insights about the emergent behavior itself. Eventually we want to examine whether the increase of model detail also leads to new forms of emergent behavior as it might increase the possible strategies available to the boids.

In the following we first describe the boid model and how it was extended. Furthermore the framework consisting of a simulation engine and an evolutionary algorithm is elaborated, followed by a description of how the models are to be evolved. To conclude this paper we show the experiment results and characterize them where model complexity, sensitivity analysis and new emergent behavior is concerned.

2 RELATED WORK

One of the first to discuss the use of evolutionary means for the purpose of tuning model parameters were Calvez and Hutzler (2005). The authors examine a model of ant foraging and describe the necessary steps to use it within an EA. Most importantly it provides valuable insight about the difficulties arising when simulation runs are used for evaluating individuals in EAs. The biggest drawback here is the enormous amount of time it can take to run evaluations. Furthermore stochastic noise is an important factor that can severely influence the outcome and quality of parameter search. Overcoming this noise usually demands multiple repetitions of evaluation runs for creating mean values, which in turn extends the run time further. Noise is also one of the issues Jin and Branke (2005) discuss in their survey of the usage of EAs in uncertain environments, which also includes simulations. Moreover it addresses various important issues, namely how to configure EAs to fit different situations related to noise.

Oboshi et al. (2003) provide insight about how they evolved escaping behavior in fish flocks using a genetic algorithm. Similar to Stonedahl they aim to directly evolve emergent behavior but concentrate solely on evasion behavior. They make use of a time oriented fitness which tries to maximize the time span an individual manages to escape its predator.

A different approach has been taken by Spector et al. (2003). This approach works inversely to the kind of parameter evolution described above. Spector evolves the agents individually all while the simulation is running, instead of using an EA to externally evolve model parameters. In this model every boid possesses its own genome. Their metabolisms evolve around a pool of energy that needs to be kept as high as possible, which essentially is the fitness function. The evolution takes place as boids die from starvation and are reborn as children of the fitter individuals, inheriting their genome.

Similarly Ikegami and Nishimura (1997) created an evolutionary-influenced predator-prey model where the scenario was turned into a game. Prey would be constantly gaining points for staying alive while predators are losing points for being hungry. Now the only possibility for a predator to gain points is hunting and encountering prey individuals from their rear. If the encounter happens with prey and predator facing each other, both will loose points. Individuals with the higher scores qualify for reproduction, which happens for an individual by spawning an offspring in the near proximity. This offspring assumes the same angle of heading as the parent and inherits its genome with slight random changes. Like Spector's model, they make use of evolving the individuals within the simulation, rather than using the simulation itself as a fitness measure for evaluating model parameters.

While our work concentrates more on the parameters of models rather than single agents Spector's work still provides valuable insight on evolving model properties.

The work reported in this paper was mostly inspired by Stonedahl's boid model. However our approach is different in so far that we are not planning to evolve emergent properties. The goal here is to evolve strategically efficient behavior of the boids, which consequently may result in emergent behavior.

3 EXTENDED BOID MODEL

In the following we will describe the boid model we used and how it was extended.

3.1 Boid Model

The common boid model contains a group of agents, called 'boids', which move around the world and tend to gather in flocks, depending on their rule parameters. Boids are using several rules to determine their direction of movement around the simulated world. The MASON boid implementation we use supplies four rules to steer the behavior.

Avoidance influences the distance the boids are keeping to each other as well as how fast they do so. Cohesion acts in the opposite direction and controls how near and strong the boids are sticking to each other. Consistency controls how fast the boids align their direction to the general direction of the flock while momentum depicts the impulse and sensor range respectively. Each of them is associated to a parameter which acts as a multiplier and determines how strong the respective rule is influencing the behavior. Additionally the parameter *neighborhood* determines the boid sensor range. These five parameters are subject to evolution by the EA. With these rules alone, boids can show distinct behavioral patterns like flocking, oscillating and floating. In the following the equations for computing the movement vectors are shown.

At first, the auxiliary variable Δd for the avoidance rule is calculated:

For nearest seven boids $b \in neighborhood$:

$$\Delta d(b) = \Delta x^2(b) * \Delta y^2(b) \tag{1}$$

where Δx and Δy denote the distance between the current and the neighboring boid b.

The actual values for the new movement vector, x_{new} and y_{new}, are computed differently for each rule:

avoidance

$$\begin{aligned} x_{new} &= \sum_{b \in neighborhood} \frac{\Delta x(b)}{\Delta d^2(b)+1} \\ y_{new} &= \sum_{b \in neighborhood} \frac{\Delta y(b)}{\Delta d^2(b)+1} \end{aligned} \tag{2}$$

cohesion

$$\begin{aligned} x_{new} &= \sum_{b \in neighborhood} \Delta x(b) \\ y_{new} &= \sum_{b \in neighborhood} \Delta y(b) \end{aligned} \tag{3}$$

consistency

$$\begin{aligned} x_{new} &= \sum_{b \in neighborhood} m_x(b) \\ y_{new} &= \sum_{b \in neighborhood} m_y(b) \end{aligned} \tag{4}$$

where m_x and m_y are parts of the momentum vector of neighboring boid b.

The momentum is given by the last movement vector, which was computed in the previous time step of the simulation.

The mean is computed by dividing the values by the number of neighboring boids. All computed mean values *avoidance*, *cohesion*, *consistency* and *momentum* are multiplied with their respective model parameters *avo*, *coh*, *con*, *mom* and eventually summed up to result in the new movement vector for the individual boid.

$$x_{boid} = x_{avoid} \times avo + x_{coh} \times coh + x_{conv} \times con + x_{mom} \times mom$$
$$y_{boid} = y_{avoid} \times avo + y_{coh} \times coh + y_{conv} \times con + y_{mom} \times mom \quad (5)$$

3.2 Prey-Predator Model

In the following we discuss how the boid model was changed in order to allow for more meaningful and strategic behavior. In our work the boid model was extended by introducing two more agent types: predators and food.

Prey-Boids Boids now act as prey, hunted by the predators while simultaneously consuming the food to stay alive. In contrast to the original implementation , boids take only the nearest seven neighboring prey-boids into account when determining their own movement.

This behavior is related to observations on starlings (Young, Scardovi, Cavagna, Giardina, and Leonard 2013) which results in more realistic behavior and furthermore reduces simulation complexity as there are less individuals to process for every agent. Nearby predators as well as food sources are separately evaluated. However, finding the seven nearest neighbors is constrained by the neighborhood parameter of the model which determines the sensor range of boids. If there are less than seven neighbors to be found within range, then only those will be used for calculation of movement.

In addition, a simple metabolism for the prey has been introduced, providing them with an individual pool of energy and a base energy level. Moving and, to a greater extent, fleeing drain energy while feeding on food sources replenishes it. Simply put, a prey boid has two states: hungry and full, as shown in Figure 1. The likelihood of getting hungry is proportional to the difference between current energy and base energy level. Upon getting hungry the prey is seeking food to refill their energy pool. After feasting on food, a fixed amount of energy is restored. Eating is repeated until the energy pool is replenished enough to not get hungry again immediately. At all times, prey will try to evade predator boids, which may lead to conflicting situations when predators are occurring near food sources.

To adapt the prey to their new environment we equipped them with two more rules: *searchFood* and *flee*. The fleeing rule technically equals avoidance, except that it only takes predators into account instead of other prey. Akin to avoidance it is also influenced by the same model parameter, creating a trade-off between flocking easier and escaping faster. The searching rule was derived from cohesion to draw boids to food sources. But in contrast to cohesion it has its own multiplier which is not influenced by the evolutionary process. This modifier, coined *hungerDrive*, results from the ratio between base energy reserve and current energy reserve. Thus the modifier ranges from 0 to 1. The more energy is drained from a boid's pool the stronger its hunger drive becomes.

Predators They comprise the upper end of the food chain in this scenario. Predators are determined to hunt and devour the prey. The states and transitions between being hungry and full are essentially the same as in the prey implementation. Similarly predators can accelerate when hunting to catch fleeing boids. When not hungry, they move towards the general direction of any prey they can detect in their surrounding, guided by the rule *stray*. This rule is derived from the cohesion rule of the prey, but not influenced by any multiplier.

When not hungry and moving around, predators move with lower speed than the prey, to save energy. This results in short periods of following boids until they are out of sight. When no boids are in sight, predators move randomly around the world. Once a predator is hungry and detects prey in his surroundings it will internally create a list of all neighboring boids. Those who have more neighbors than others, meaning they are in a flock, will be removed since isolated targets are preferred. The resulting list contains all boids that have the same minimal number of neighboring boids. Predators will randomly choose one member of the list to hunt down, since they are all difficult to focus on.

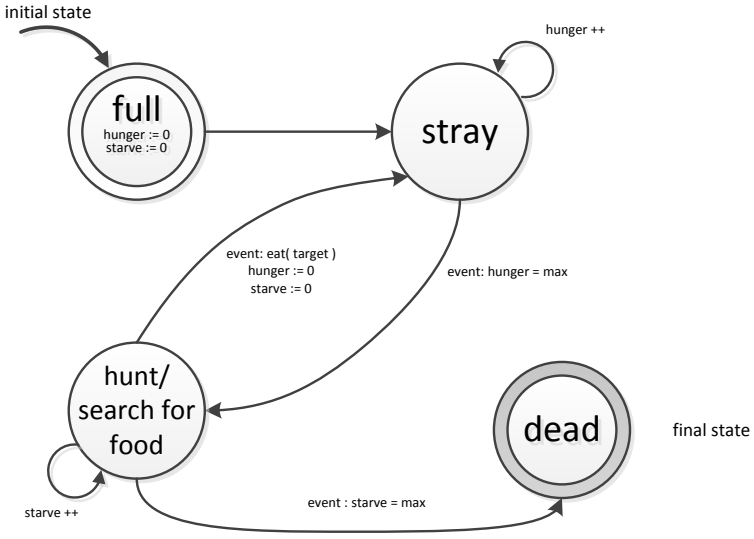


Figure 1: Emotion states of prey and predators.

To sum up, predators are influenced by the rules *stray*, *hunt* and *momentum*. Only hunt has a multiplier and none of the rules get changed in any way by the evolutionary algorithm. The experiments are solely concentrating on evolving the prey.

Food Sources The food sources are simply passive entities situated in the world. They possess a certain amount of food which is decreased by one every time a boid feasts on them. Furthermore if one food source is depleted it vanishes and re-spawns at a different place to simulate over-farming of that area. After being depleted in the respawn location, it respawns in the original place, considered as regrown. This adds an incentive for the boids to keep being flexible as a swarm and attributes for the danger of over-farming.

3.3 Multi-Species Prey-Predator Model

The Multi-Species Prey-Predator Model introduces different species of prey boids; everyone of them now belongs to one autonomous acting species. This is done to add complexity and emulate more advanced societies, where different species are flocking together or cooperating, as seen in bird flocks and hierarchies similar to beehives containing drones, workers and queens. Every group of boids has its own parameter set consisting of the aforementioned five parameters, increasing the number of total parameters up to 15. Keep in mind that all boids belonging to one group will be given the same parameter values.

Alliances By default, boids of one group refrain from flocking with boids from another group. They are ignoring members of other species when applying their rules. The only exception here is avoidance which is used to prevent collision. Regarding their separated parameter sets, they can be considered completely independent groups. To create interaction between them we introduced three further, also evolvable, parameters. Each one of them comprises of a Boolean value which defines whether the specified groups are allowed, *allied*, to flock or not: group 1 and 2, group 1 and 3, group 2 and 3. After adding alliance parameters to the chromosome we now have a total of 18 parameters to evolve.

4 EVOLVING BOID MODEL

4.1 Evolutionary Algorithm And Framework

This biologically inspired class of algorithms is very effective in exploring large solution spaces and has already been employed in various scientific fields such as task assignment (Ho, Lin, Liauh, and Ho 2008), power systems (Das, Venayagamoorthy, and Aliyu 2008) and evolving of simulation model parameters (Johansson, Helbing, and Shukla 2007).

By default Evolution Strategies (ES) are designed to work with populations of real valued vectors. This avoids unnecessary complexity by converting the parameters into binary strings which are more commonly used by Genetic Algorithms (GA). Furthermore Beyer (Beyer 2001) offers a comprehensive theoretical groundwork for the use of ES.

An individual of the ES will be a five-dimensional real-valued vector, comprised of the parameters: avoidance, cohesion, consistency, momentum and neighborhood. We use the common $(\mu + \lambda)$ - type ES where μ indicates the number of individuals which remain in the population to mate and generate the children. Furthermore λ denotes how many children will be generated from the μ parents. It follows that $(\mu + \lambda)$ is also the population size. The EA employs a basic one-point crossover which cuts two parent chromosomes in half and exchanges sub-strings between them. This suffices as recombination factor since ES are devised to use mutation as main evolutionary driving force. By the nature of design ES can also function without any recombination at all.

As stated above, Mutation is the main driving force behind evolution strategies. In this particular case it takes place by adding a random Gaussian distributed value to each parameter of the individual. This operator is always executed on a newly generated child. We also specified a standard deviation of the random value of 0.7 which had been determined by a series of experiments. All parameters are set to have their values range from 0 to 20.

To simulate the boid model and evolve its parameters we combined the Java-based tools MASON and ECJ, both developed by Sean Luke et al. (Luke, Balan, and Panait 2003), (Luke, Panait, Balan, and Et 2007) to create a framework that allows to be used for both tasks; simulation as well as evolution. MASON, a framework specifically designed for multi-agent based simulation, already provides a two dimensional boid model which we used and extended over the course of the experiments. Initially it only consisted of a space in which a certain number of boids are moving around, all using the same rules and same parameters. ECJ is an evolutionary research system that offers a variation of evolutionary algorithms including genetic algorithms and evolution strategies.

ECJ, even though mostly configured by using parameter files, is equipped with a couple of Java-interfaces to customize the EA in great detail. One of those allows to write user defined classes for implementing own fitness functions. And in our case, these fitness functions act as the interface between ECJ and MASON. A fitness function in ECJ will be executed when an individual is to be evaluated. This marks the interface between Simulation and EA. We devised the fitness function to evaluate individuals by starting an instance of the MASON model and setting model parameters to the respective values, according to what is encoded in the individual. An additional monitor class, able to access the data of all single agents, records values necessary for determining the fitness and delivers it back to the fitness function class which in turn can be read by ECJ for further processing.

4.2 Evolving Prey-Predator Model

We have established a basic food chain, starting with food sources and ending with predators. The goal is to find out the parameter configuration that maximizes the prey survival rate. We constrained the problem to the time span within the first 5000 simulation time steps. This interval was chosen as a reasonable middle ground to have possibly low run time costs for the EA. At the same time it is considered long enough for the boids to develop a distinct behavior and for the predators to be able to seriously endanger the prey. All experiments were done using two food sources, seven predators and 75 prey boids in the simulation.

In order to increase their chances of survival, boids would have to find a way of evading predators as well as getting enough food to avoid starvation. Parameters to evolve are only the ones influencing prey boids, as described in Section 3.1. All five parameters: avoidance, cohesion, consistency, momentum and neighborhood are floating point variables, ranging from 0 to 20. They are randomly initialized and evolved by the ES for 50 generations. The fitness of each individual is measured by the number of boids that survive until the simulation reaches the 5000 step time limit. Due to how the predators hunt and acquire their target, we believe that flocking behavior should be one of the most profitable strategies for boids. But this only applies as long as the flocking does not hinder the mobility of prey, impeding their ability to find food sources. Theoretically speaking, a single boid which is very agile might also have decent chances to survive if it can outmaneuver predators. But this seems unlikely if the whole population is following the same strategy, as the boid density in our simulated world is high enough for predators to regularly encounter potential targets. The fitness function counts the surviving prey boids after the simulation ran for 5000 time steps. This also attributes to the simplicity of the model.

During the EA runs the fitness is converging to an average of 70 surviving boids which is about 93% of the population as shown in figures 4a and 4b. As expected, the boids showed flocking behavior, increasing their chance of survival by aggravating the predators abilities to lock on a specific target. The ES parameters were set to reasonable values, meaning to find a middle ground between shorter run time and higher quality. For purposes of fine tuning we tested several different configurations. For the evolution we tested different configurations of the EA, concerning the size of population as well as the number of evaluation samples per individual, which is used to decrease stochastic noise in the results. Population sizes of 25, 50 and 100 delivered similar results, with the latter ones having a more smoothed out fitness curve over generations, due to the larger sample size. This improvement however comes at a high cost of run time. The results in terms of fitness and convergence only changed in negligible dimensions. Ultimately the majority of experiments in this series was done using a population of 25 individuals for the EA.

4.3 Evolving Multi-Species Prey-Predator Model

In contrast to what has been described in the last sub-section we now divide the prey boids in up to three different species, resembling inter-species communities occurring in nature (Hunt, Harrisom, Hamner, and Obst 1988). Upon creating the simulation, each boid is randomly assigned to one group. The probability of assigning a boid is equal for each species, so that all of them have approximately the same number of members. This extension of our boid model is supposed to enable a wider variation of possible behavioral patterns. We want to examine how this increase of detail affects evolving the boid survival capabilities. There are potentially more ways to tackle the problem of finding food and escaping predators. Again, as in the previous chapter, all experiments used 2 food sources, 7 predators and 75 prey boids in the simulation. The same applies to the fitness function, which still measures the number of surviving boids at the end of simulation.

Each group has its own set of parameters, independent from the other groups' settings. This triples the number of parameters and in addition to that, three more parameters are added to enable or disable so called alliances. These three values are represented by Boolean variables in the simulation and decide whether groups 1 and 2, groups 1 and 3 or groups 2 and 3 are permitted to flock with each other. Incorporating Boolean values into the individual is done by adding three more real valued variables to the vector and constraining their range from 0 to 1. Upon loading the vector into the boid model, the value is decided by whether it is greater or smaller than 0.5. To sum up, the parameter vector is increased to eighteen dimensions as shown in Figure 2. All of these are subject to evolution. The parameter ranges are set at 0 to 20 again to keep the experiments consistent. Prey boids now have the opportunity to cooperate with other species or to stay isolated.

Differentiations between species of boids and in particular the possibility to cooperate or isolate themselves should enable the EA to find better or alternative solutions for the survival problem. Moving in smaller flocks may be an attractive option for more agile species which would benefit from keeping

Figure 2: Vector representation for an individual in the ES.

their mobility. On the other side, bigger flocks consisting of multiple races may offer more protection for members of inflexible and slower moving species.

5 EXPERIMENTS AND RESULTS

5.1 MODEL COMPLEXITY

The Prey-Predator Model showed flocking behavior of the prey as it was predicted to increase the boids survival chances. After starting the simulation the boids begin to flock in several smaller groups. Depending on predator behavior and the course of the simulation, the groups start to merge until eventually very few or even only one big flock is left. The mean count of surviving boids after applying the EA is close to 70 prey individuals. According to an initial population of 75 individuals this is a survival rate of 93%. We can safely say that this is very close to the global optimum since a certain number of casualties is unavoidable, regarding to the model configuration. This way we created flocking behavior as a result of strategically efficient behavior, rather than by directly searching for it, as it was the case in Stonedahl's work.

The Multi-Species Prey-Predator Model comes with greater detail as it allows up to three individually evolve-able species of prey and is therefore supposed to offer a much wider range of possible solutions to the survival problem.

However, as Figures 4a and 4b show, the fitness decreases to an average between 52 and 50 surviving boids after increasing the number of species to three. The survival rate rapidly declines from 93% with one race to 68% with three boid species.

The experiments with two prey species appear to be very unstable. While reaching a fitness of 68 in the first run, which is close to 90% survival rate, it is unable to achieve the same success in the second run. The three species model however is seemingly unable to reach any fitness higher than 52 to 54 survivors, and less fluctuating in its results that the two-species model. This behavior indicates the increasing difficulty for the EA when it comes to handling more complex models. Taking into account that all EA runs start off at roughly the same fitness (see figures 4a and 4b), the evolution of 2 species still has the potential to reach very good results but it is considerably harder to not drift into local optima.

Neither one of the experiments conducted with two and three species was capable of reaching the success rate of the single species experiment.

This is rather counter-intuitive, especially since the single species is basically a subset of the multi species, precisely it denotes the special case where all species are allied and have the same parameters. The question now rises: why is the EA unable to find at least the solution from the first experiment again, let alone an even better one? Several potential reasons may explain that behavior. Doubling or tripling the number of parameters and adding the alliance parameters extend the search space considerably. On the other side, one of the strengths of EAs is their ability to handle gigantic search spaces. A more plausible explanation could be the introduction of fundamentally more stochastic noise by the greater detail, which leads to premature convergence.

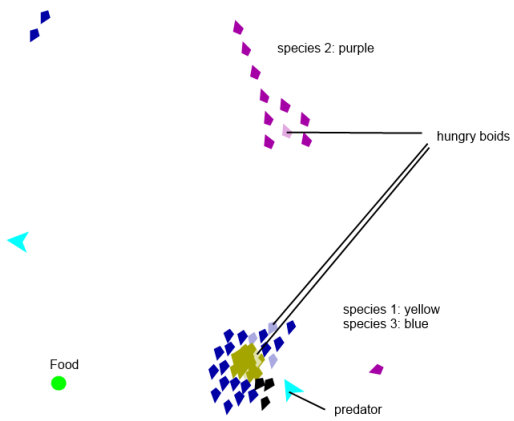
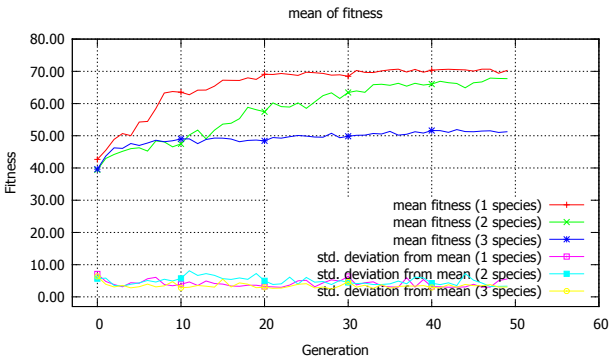
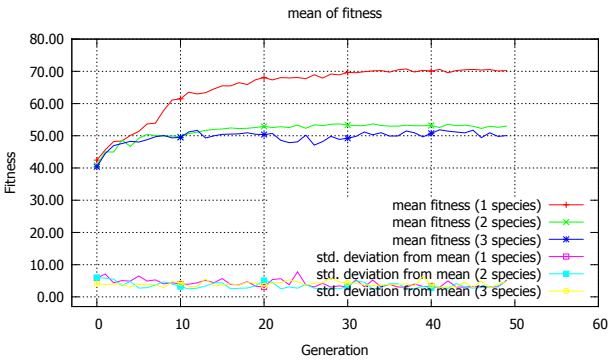


Figure 3: Collective flocking of two species. Yellow and blue species are flocking together while the green species is on its own.



(a) Results of survival experiment with varying number of species, first run.



(b) Results of survival experiment with varying number of species, second run.

5.2 Sensitivity Analysis

Evolving parameter values of crowd simulations is only the first part of the problem because once one or more individuals are found, it has to be made sure that this solution is really robust enough to repeat its performance under changing environments. The kind of sensitivity analysis we conduct here is also inspired by Stonedahl et al. (Stonedahl and Wilensky 2010) who experimented with evolving contrasting fitness functions.

This analysis is executed by limiting the parameter ranges to an interval containing all parameter values of the individuals of the last generation. These are supposed to be the fittest ones produced by the EA. Furthermore the fitness function, survival, is now inverted to evolve the lowest survival rate possible within those parameter ranges. Simply put, we intend to find out how stable the optimized solutions of the EA are.

As shown in the sensitivity analysis for the one species model in figure 5 we can see that our optimized solutions with an average fitness of 70 can lead to about 20 casualties in the worst case. This means that our solutions are guaranteeing at least 50 survivors, while having an average of 70 survivors. The same applies for the two species model.

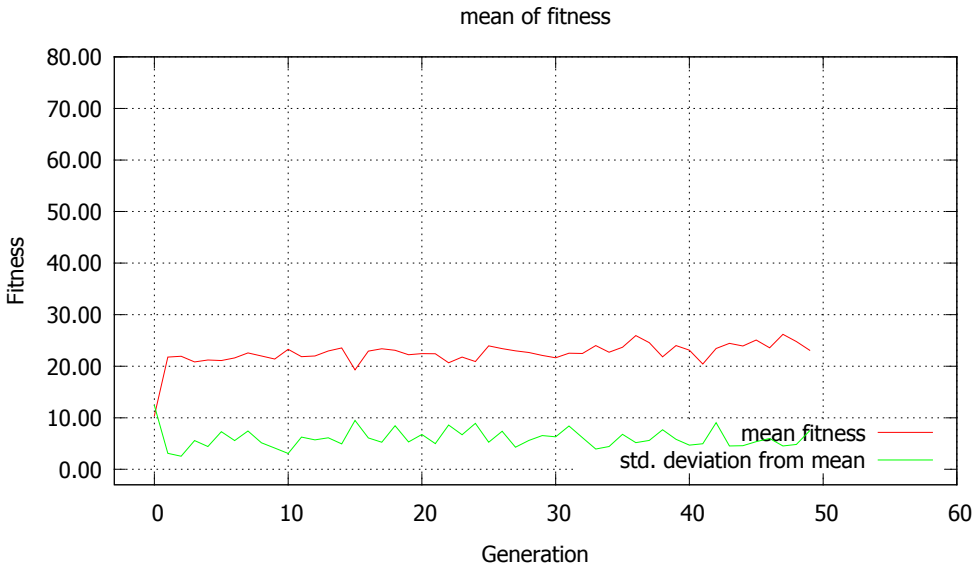


Figure 5: Sensitivity Analysis: maximizing the number of boids getting eaten, while using the best parameters for the survival fitness.

Therefore we did not only determine the robustness of the final generation but also managed to filter out the best and most stable individuals among the potentially best ones. Our sensitivity analysis furthermore indicates that consistency and momentum are less influential to the best solutions than neighborhood and consistency since their parameter values are distributed over a larger interval.

5.3 New Emergent Behavior

Consistent as well is the emergent behavior observed in the optimized solutions. They show a new kind of emergent behavior throughout all EA runs. One of the species gathers in very dense flocks while one or two allied species group more sparsely, leading to ring-shaped groups around the dense ones, as shown in figure 3. If one of the two remaining species is not allied with the dense flocking boids, they form their own sparse and loosely connected flocks. This behavior leads to an unevenly distributed number of casualties. While the dense flocking boids get eaten only on very rare occasions due to their enclosure,

the surrounding allies account for far more losses. This is caused by their exposed positioning around the dense flocks which makes them virtually act as guards, thus suffering frequent predator attacks.

The boid species also follow the rules of Schelling's Segregation (Schelling 1969). Allied species always gather in groups, if their parameters allow them to flock. Additionally it appears that alliances are transitive. If species 1 and 2 as well as 2 and 3 are allied, but not 1 and 3 then still all of them are going to flock together using boids of species 2. Furthermore the experiments indicate that a relatively small number of hungry boids is sufficient to steer the whole flock towards a food source. In our simulations we noticed that an amount of hungry boids about 1/4 to 1/3 of the flock size suffices to enforce the flock to move towards food.

Lastly, during the experiments we noticed that none of the Multi-Species Experiments was able to produce parameter vectors that reach the fitness values of the Single-Species Experiments.

6 CONCLUSIONS AND FUTURE WORK

In this paper we demonstrated a method for evolving agent-based model parameters towards an objective, indirectly leading to the creation of emergent behavior. Additionally we increased the model complexity and examined the consequent effects on the evolutionary process. It is to be noted that noise plays an important role in the workings of the EA. Too noisy fitness landscapes can severely affect the convergence characteristics and cause the EA to get stuck in local optima. Furthermore we discovered that increasing the model complexity has negative effects on the outcome of parameter evolution. It is not absolutely clear what causes the multi-species experiments to show declining results in fitness. However, the most likely rationale is that boids of a single species are naturally better at surviving in the scenario and the increased complexity is too much for the EA to handle with the current configuration.

There are still open questions to address in future work. The problem of evolving agent behavior in models of varying complexity applies especially to crowd simulation. The concept of our work can easily be transferred to the social force model and be tested for more complex objectives. It will be of interest to see whether the anomaly of decreasing fitness with increasing complexity persists. Furthermore it is to be considered to extend this work towards rule-based crowd simulation. Objectives that are to be achieved in crowd simulation often involve emergent behavior and the proposed method can be the way to examine models in regard of such properties.

REFERENCES

Beyer, H. 2001. *The Theory of Evolution Strategies*. Natural Computing Series. Springer.

Calvez, B., and G. Hutzler. 2005. "Automatic Tuning Of Agent-based Models Using Genetic Algorithms". In *In MABS*, 41–57.

Das, T., G. Venayagamoorthy, and U. Aliyu. 2008, sept.-oct.. "Bio-Inspired Algorithms for the Design of Multiple Optimal Power System Stabilizers: SPPSO and BFA". *Industry Applications, IEEE Transactions on* 44 (5): 1445 –1457.

Ho, S.-Y., H.-S. Lin, W.-H. Liauh, and S.-J. Ho. 2008. "OPSO: Orthogonal Particle Swarm Optimization and Its Application to Task Assignment Problems". *Trans. Sys. Man Cyber. Part A* 38 (2): 288–298.

Hunt, G., N. Harrisom, W. Hamner, and B. Obst. 1988. *Observations Of A Mixed-Species Flock of Birds Foraging On Euphausiids Near St. Matthew Island, Bering Sea*.

Ikegami, T., and S. I. Nishimura. 1997. "Emergence of Collective Strategies in a Prey-Predator Game Model". *Artificial Life* 3 (4): 243–260.

Jin, Y., and J. Branke. 2005. "Evolutionary Optimization in Uncertain Environments - A Survey". *IEEE Trans. Evolutionary Computation* 9 (3): 303–317.

Johansson, A., D. Helbing, and P. Shukla. 2007. "Specification of the Social Force Pedestrian Model by Evolutionary Adjustment to Video Tracking Data". *Advances in Complex Systems* 10 (supp02): 271–288.

Luke, S., G. C. Balan, and L. Panait. 2003. "MASON: A Java Multi-agent Simulation Library". In *Proceedings of the Second International Workshop on the Mathematics and Algorithms of Social Insects*. Atlanta, Georgia.

Luke, Sean and Panait, Liviu and Balan, Gabriel and Et 2007. "ECJ 16: A Java-based Evolutionary Computation Research System".

Oboshi, T., S. Kato, A. Mutoh, and H. Itoh. 2003. "Collective or Scattering: Evolving Schooling Behaviors to Escape from Predator". In *Proceedings of the eighth international conference on Artificial life*, ICAL 2003, 386–389. Cambridge, MA, USA: MIT Press.

Reynolds, C. W. 1987. "Flocks, Herds and Schools: A Distributed Behavioral Model". In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, 25–34. New York, NY, USA: ACM.

Schelling, T. 1969. *Models of segregation*. Memorandum (Rand Corporation). Rand Corp.

Spector, L., J. Klein, C. Perry, and M. Feinstein. 2003. "Emergence of Collective Behavior in Evolving Populations of Flying Agents". In *GECCO*, 61–73.

Stonedahl, F., and U. Wilensky. 2010. "Finding Forms of Flocking: Evolutionary Search in ABM Parameter-Spaces". In *MABS*, 61–75.

Young, G. F., L. Scardovi, A. Cavagna, I. Giardina, and N. E. Leonard. 2013, February. "Starling Flock Networks Manage Uncertainty in Consensus at Low Cost". *PLoS Computational Biology* 9 (1): e1002894+.

AUTHOR BIOGRAPHIES

MICHAEL WAGNER is a Project Officer in the Parallel and Distributed Computing Centre at Nanyang Technological University, Singapore. His research interests include Evolutionary Algorithms with a focus on Simulation Optimization. He holds a Bachelor of Science Degree from Rostock University. His email address is mwagner@ntu.edu.sg.

WENTONG CAI is a Professor in the School of Computer Engineering at Nanyang Technological University, Singapore. He is also the Director of the Parallel and Distributed Computing Centre. He received his Ph.D. in Computer Science from University of Exeter (UK) in 1991. His expertise is mainly in the areas of Modeling and Simulation (particularly, modeling and simulation of large-scale complex systems, and system support for distributed simulation and virtual environments) and Parallel and Distributed Computing (particularly, Cloud, Grid and Cluster computing). He is an associate editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS) and an editor of the Future Generation Computer Systems (FGCS). His email address is aswwtcai@ntu.edu.sg and his web page is http://www.ntu.edu.sg/home/aswtcai/.

MICHAEL LEES is an Assistant Professor in the School of Computer Engineering, Nanyang Technological University (NTU). His research interests are primarily in modelling and simulation of large scale complex systems, he is particularly interested in understanding the effect that human behavior has on such systems and the important role that individual behavioral interactions have on system level dynamics. His email address is mhlees@ntu.edu.sg and his web page is http://www.mhlees.com.