DEPARTMENT OF INFORMATICS

LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Swarm Intelligence

Introduction and Replication of a Bird Flocking Behavior Simulation

Master Seminar Paper Modeling Dynamic and Adaptive Systems

Authors:	Valentin Scheibe
Supervisors:	Prof. Dr. Martin Wirsing, Prof. Dr. Wolfgang Hesse
Submission Date:	31.03.2023

Declaration of Independence

Name: Valentin Scheibe Matrikelnummer: 11599198

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt, alle Zitate als solche kenntlich gemacht sowie alle benutzten Quellen und Hilfsmittel angegeben habe.

München, den 27.03.2023 ....

Valentin Scheibe

Abstract

Swarm intelligence is an exciting area of research that is heavily inspired by the collective behavior found in nature, ranging from the schooling of fish to the flocking of birds.

In this paper I provide an introduction to swarm intelligence and I present an example of a bird flocking simulation. To explain the history and key characteristics of swarm intelligence, I introduce three different algorithms that use swarm intelligence mechanics and can solve a wide range of computational problems.

Bird flocking is a popular example of emergent intelligent behavior, so to better illustrate bird flocking, I replicated the recently published research of De Nicola et al., "Modelling Flocks of Birds from the Bottom Up." I was able to reproduce the findings of their work even though I was using a different method of evaluation and implementation while also adjusting parameters from their original work.

Schwarmintelligenz ist ein spannendes Forschungsgebiet, das stark vom kollektiven Verhalten in der Natur inspiriert ist, vom Fischschwarm bis zum Vogelschwarm.

In diesem Beitrag gebe ich eine Einführung in die Schwarmintelligenz und präsentiere ein Beispiel für eine Simulation eines Vogelschwarms. Um die Geschichte und die wichtigsten Merkmale der Schwarmintelligenz zu erläutern, stelle ich drei verschiedene Algorithmen vor, die die Mechanismen der Schwarmintelligenz nutzen und eine breite Palette von Rechenproblemen lösen können.

Vogelschwärme sind ein beliebtes Beispiel für intelligentes Verhalten. Um den Vogelschwarm besser zu veranschaulichen, habe ich die kürzlich veröffentlichte Forschungsarbeit von De Nicola et al. mit dem Titel "Modelling Flocks of Birds from the Bottom Up" nachgestellt. Ich war in der Lage, die Ergebnisse ihrer Arbeit zu reproduzieren, obwohl ich eine andere Auswertungs- und Implementierungsmethode verwendete und auch die Parameter ihrer ursprünglichen Arbeit anpasste.[1]

[1]Translated with Deepl [8]

Contents

1 Introduction

Swarming is a common behavior in nature and science that is widely studied, making it a highly interdisciplinary research field. Swarming is mostly described as a collective behavior from similar entities that stay together through decentralized actions [4]. It is found in nature from schools of fish, flocks of birds, and herds of animals to swarms of drones/robots in the technical world. In the earlier days of computer science, people like Craig Reynolds in 1987 started to realize the potential of natural swarming behavior for the field and started emulating it in artificial life programs [20].

In light of recent developments in artificial intelligence this is how swarm intelligence is defined in the Collins dictionary:
"1: the collective behaviour of a group of animals, esp social insects such as ants, bees, and termites, that are each following very basic rules"
"2: an artificial-intelligence approach to problem solving using algorithms based on the self-organized collective behaviour of social insects" [22]
Compared to ChatGPT's answer for a definition on swam intelligence:
"Swarm intelligence refers to the collective behavior of decentralized, self-organized systems in nature, such as flocks of birds, schools of fish, and colonies of ants. These systems exhibit emergent properties that arise from the interaction of individual agents with their environment and with each other. In summary, swarm intelligence is a powerful concept that can be used to develop intelligent and adaptive solutions to complex problems. By studying the collective behavior of natural systems and applying these principles to artificial systems, researchers are developing new and innovative approaches to solving a wide range of problems."[5]

In this work, I want to give a brief overview over swarm behavior and intelligence with regards to computer science. I do so by first highlighting three important swarming algorithms and their applications and then investigating recent research on the behavior of flocks of birds. For this, I tried reproducing the bottom-up implementation of bird flocking behavior and its results from authors De Nicola et al. [7] in 2022 by using the Python MESA framework.

This work has the following structure: In Section 2, I discuss swarm behavior and intelligence whilst highlighting three prominent examples. In Section 3, I present my method of replicating De Nicola et al. [7] work and the results obtained thereby. I then conclude this work in Section 4.

2 Swarm Behavior and Intelligence

Swarm behavior is the collective behavior of similar entities that aggregate in a group[4]. Swarm intelligence is a term coined by Beni & Wang in 1989 and describes the emerging intelligent behavior of a swarm [2]. Examples of such emergent behavior are ant colonies' food finding strategies or predator aversion and evasion from flocks of birds. The latter of which I will look at closer in Section 3. From here on out I will be using swarm intelligence as a collective term for both. In 1993 Millonas in his paper on models for application of artificial life described five principles of swarm intelligence [17]:

- Proximity Principle: The population of swarm should be able to compute space and time variables.
- Quality Principle: The population of the swarm should be able to respond to changes and qualities in the environment they are in.
- Diverse Response Principle: The population of the swarm should not adhere to activities along narrow response channels.
- Stability Principle: The population of the swarm should not always change its behavior when the environment is changing.
- Adaptability Principle: The population of the swarm must be able to change its behavior when its favorable for the swarm.

Implementations of swarm intelligence are often done via computer simulations using an agent based modelling approach. According to Korb and Mascaro the epistemology of a computer simulation is enough to satisfy the requirements of scientific experiment. In other words computer simulations can be used to investigate real world phenomenons like swarm intelligence [14]. A key aspect of swarm intelligence is that is lacks a central control structure. The emergence of intelligent behavior is a result of multiple simple agents that follow simple instructions [3].The most basic rules for such an agent were used by Reynolds in 1987 [20]:

- Move in the same direction as neighboring agents
- Move in closer to neighboring agents

- Avoid Collisions with neighboring agents

Bonabeau, Dorigo and Theraulaz [3] used those rules whilst looking at social insects to apply these concepts to complex systems. Their work is organized around multiple biological examples, "which [are] then used to develop an algorithm, a multi agent system, or a group of robots" [3]. There are two approaches to investigate swarm intelligence the mathematical and the evolutionary approach. Reynolds work follows the mathematical approach, using mathematical models to simulate swarm intelligence [20]. Others use an evolutionary approach to explain how swarm intelligence evolved [25][19][23][18].

According to Hamann, one reason why the natural behavior of insects is interesting to computer science and robotics is that a system that a large group of small systems forming one intelligent system is inherently robust against failure of a few of those small systems. He also postulates that the application of swarm intelligence can be scary or even detrimental to humans, especially in the context of automated weapons systems [11]. Swarm intelligence has a plethora of applications today. Some of those are used in solving optimization problems such as the travelling sales man problem or shortest path on a weighted graph problems.

2.1 Boids

In 1987 Craig Reynolds published his paper Flocks, Herd, and Schools: A Distributed Behavioral Model [20] where he presented his artificial life program called Boids. Boid stands for bird-oid object. He developed Boids to aid animating flocks of birds in simulated computer animation. Boids is one of the first examples of artificial swarm intelligence because it shows an emergent behavior of its agents. The bird-oid objects in the Boids program follow three simple rules [20]:

- Collision Avoidance: avoid collisions with nearby flockmates
- Velocity Matching: attempt to match velocity with nearby flockmates
- Flock Centering: attempt to stay close to nearby flockmates

Reynolds noted that some limitations of simulating real flocks of birds were the way he simulated the perception of the Boids. In real life birds rely on vision, hearing and many more factors that determine their actions but in the computer simulation the Boids had perfect knowledge of the world and positions of other Boids with in it. He mitigated that by among other things introducing localized flock centering [20]. Reynolds work was a necessary step forward in an under explored field at the time, which lead to improvements in several other fields. Such as Boids being used in animations of movies and video games [1].

2.2 Ant Colony Optimization

As the name already implies, ant colony optimization is an algorithm developed by Marco Dorigo in 1991 that solves optimization problems in a way that is inspired by the swarm intelligence of real life ant colonies [6]. It was first designed for shortest path on a weighted graph problems where much like real ants looking for the shortest route to food the ant agents would find the shortest path on a weighted graph.

In nature ants find the shortest path to their food source by first pouring out of their starting point randomly leaving pheromones on their trail. When an ant finds a food source it will remember that path by following its pheromone trail. Other ants follow paths with the highest pheromone concentration. Because the pheromones the ants leave have certain time, after which they decay, only the paths that lead to food are taken by the ants. This is most of the time also the shortest path since it takes the ant less time to travel the path and thereby renewing the pheromone trace earlier [9].

The algorithm works in a similar way. First the agents (ants) generate solutions on the weighted graph by considering the edges that exit from their current node and their corresponding pheromone levels. Second the solutions of all agents are compared according to their proficiency and the pheromone levels are updated accordingly [6].

The ant colony optimization algorithm comes in many different variations with its applications being even more plentiful. A prominent problem which the algorithm provides good solutions to is the travelling sales man problem. It also provides solutions to scheduling problems [10], vehicle routing problems [24], image processing problems [16] and many more.

2.3 Particle Swarm Optimization

James Kennedy and Russel C. Eberhart developed the basic Particle Swarm Optimization algorithm in 1995 [13], which Eberhart refined later on in 1998 together with Yuhui Shi [21].

The basic variant of the Particle Swarm Optimization algorithm as described in [13] works by improving solutions to a problem landscape in an iterative way. A population of solution candidates called particles are moved around in the search space with a mathematical formula over the particles' velocities and positions. The formula influences the position and velocity of a particle according to the locally best known position and the global best known position in the search space. The global best position is continuously updated by the particles as they discover new best positions.

This, in theory, should move the population of particles (called swarm) towards the globally best position in the search space. In practice they have shown to improve the correctness percentage of neural networks and finding the global optima in highly discontinuous data sets [13].

Particle Swarm Optimization is a very simple algorithm that can effectively optimize an extensive amount of different functions that don't need to be differentiable like classic optimization methods such as Gradient Descent. Kennedy first used Particle Swarm Optimization to successfully adapt and optimize neural network weights in an attempt to "simulate the ability of human societies to process knowledge"[12].

3 Replication of a Bird Flocking Behavior Simulation

To evaluate recent research in flocking behavior algorithms I reviewed De Nicola et al.[7] paper from 2022 and tried to replicate their results. For that I proposed the following research question: Can the results of De Nicola et al. be replicated by following the bottom-up description of their model in a different programming environment? Their results showed the evasive behavior of a flock of birds when facing a predator that moves on a fixed path. De Nicola et al. [7] used an incremental approach to model a flock of birds that is eventually able to evade an approaching predator. To achieve this incremental model they dissected the birds' behavior into five distinct steps:

1. Movement:
   A bird is enabled to move along vectors of maximum length 2 on a two dimensional grid.

2. Alignment:
   Birds are now capable of adapting their alignment vectors by changing the movement vector of a bird to the one of a randomly picked bird from the flock.

3. Cohesion: Birds in the flock can now gain cohesion by allowing a bird to observe the direction of another bird and allowing it to steer to an estimated future position of that bird.

4. Dispersion and Collision:
   Here they prevent collisions and dispersion of the flock by calculating if a bird is isolated and if a bird would move onto the same position as another bird. If a bird is isolated it cannot be chosen by other birds as a bird that is steered towards.

5. Fleeing The flock is now able to detect if a predator is too close and allow the birds to calculate a new movement vector that is the opposite direction of the predators movement vector. They call the new movement vector "a repulsive heading vector" [7]

3.1 System Architecture

To answer my research question I began by implementing the model descriptions of De Nicola et al. in the Python programming language framework MESA [15]. MESA is an agent-based modeling framework that provides a single grid and scheduling capabilities that progresses the simulation in a step-by-step manner. It places agents on the grid and selects agents in a random order on every step of the scheduler. When an agent is selected it activates the methods specified in its step() method. The MESA framework's single grid only allows a single agent on one cell of the grid thereby collision of birds is prevented by the program terminating. The MESA framework also allowed me to collect data on the birds and to visualize their behavior in a JavaScript animation. The Figure 3.1 shows a simplified class diagram of the implementation using the MESA framework. The FlockingModel handles all agents in the simulations and has a DataCollector. A separate config-file makes the simulation easily adjustable as all relevant parts implemented by me use parameters from it. All agents and the model implement the step() method, where the FlockingModel calls the step() methods of the agents. The MESA framework also provides a Visualization next to the Model and Agent classes.

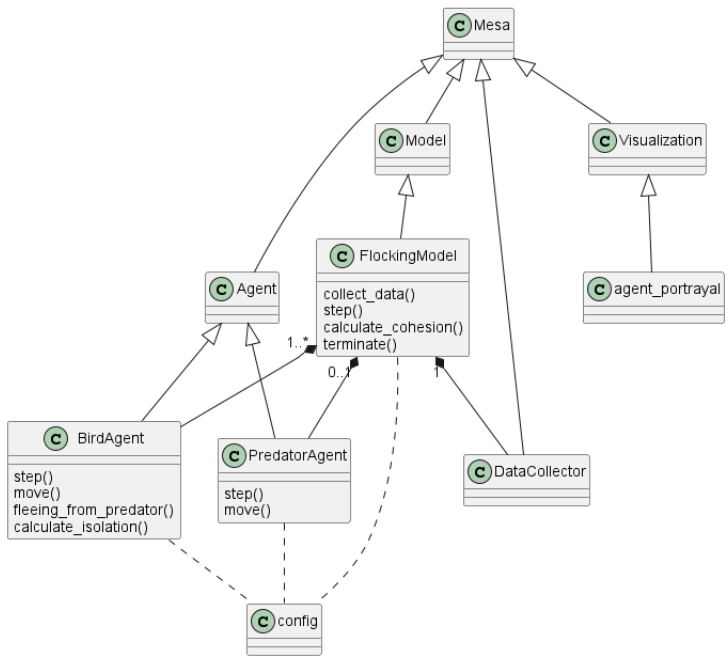

Figure 3.1: Class diagram of my simulation implementation

Listing 3.1 shows pseudo-code for how the algorithm of every single bird functions. In each step each bird first evaluates if it is isolated from the rest of the flock and updates its internal value accordingly. Then each bird calculates it's escape vector from the predator bird. Each bird also calculates its normal approach vector according to points 1-3 of the enumeration in chapter (3). After both movement vectors are calculated the escape vector gets checked against the safe distance parameter specified in the configuration 3.1. This is done by calculating the estimated future position of the predator and then calculating the Manhattan distance from the birds own position to this estimated future position. If this distance is smaller than the specified safe distance the bird updates its direction vector (dir_x, dir_y) to the escape vector that was calculated before. If this distance is larger than the specified safe distance then the bird updates its direction vector to the calculated approach vector. After the direction is updated each bird moves on the grid according to its newly calculated direction vector. The predator is very simply modeled. It has a larger movement vector and moves in a straight line through the flock of birds.

Listing 3.1: Python pseudo-code of bird behavior algorithm

```python
# Bird Behavior Model
class BirdAgent(Agent):
    dir_x
    dir_y
    isIsolated = False

    def step():
        calculate_isolation()
        calculate_escape_vector()
        calculate_approach_vector()
        check_safe_distance()
        update_direction()
        move()
```

3.2 Configuration

To appropriately model the system of De Nicola et al. I had to adjust a few different parameters which I highlighted in Table 3.1. I shrank down the size of the arena to allow for a better visual representation of the simulation using MESA's visualization module. The resulting arena was still large enough so that the birds never reached the bounds of the arena in my simulations. For a better visualization I also increased the number of birds in my simulation which did not change its results other than shrinking the cohesion value marginally. I decided for a different approach to evaluate my results than De Nicola et al. by using mean distance calculations instead of checking for cohesion property satisfaction after a fixed time period of 600 steps. I have left critical values like "Safe Distance" and "Isolation distance" the same and have taken them directly from De Nicola et al. [7].

Name	**Values by De Nicola et al.**	**Replication Values**
Maximum velocity of birds	2	2
Arena size	1024x1024	256x256
Safe distance from predator	32	32
Isolation distance	32	32
Amount of Birds	29	100
Cohesion property	40 after 600 steps	Mean

Table 3.1: Parameter configuration table highlighting changes compared to the model of De Nicola et al.

3.3 Data Collection Process

The MESA framework offers an inbuilt data collection module that is able to collect data in a highly customized fashion easily exported to statistics programs like R. In this case I collected the calculated mean distance between any two birds and exported them to a data-frame. The data collector stores those values at the end of each step.

3.4 Results of the Simulation

In the following I present the results of the simulation. I obtained my results by running the simulation 10 times with 10 different seeds. I then averaged the mean values of distance between any two birds for all my simulations. I did all statistics calculations with R version 4.2.2 and Rstudio version 2022.12.0 Build 353.

The Figure 3.2 shows a visualization of my simulation. In the Figure you can clearly see the sought after emergent behavior: the birds represented as black dots clearly evade the predator represented as a red dot. After the predator leaves the flock gets visually smaller again showing that the flock is no longer perturbed and returns to its usual flocking behavior. The numbers on this figure represent snapshots of the simulation in chronological sequence.

In Figure 3.3 you can see the actual data results from the simulations. My results are calculated differently from De Nicola et al. [7] as they check if the cohesion property is satisfied after 600 steps. The cohesion property is calculated for every bird by checking if the Manhattan distance between the bird and each other bird is smaller than a parameter k specified in their configuration. Their simulation satisfies the cohesion property if every bird satisfies it. For the scope of this work and a better visualization I have used a different approach to show the cohesion of the flock. Whilst De Nicola et al. only show a single point in time for each simulation, I have collected the mean distance between each bird each step. With that my cohesion property is visible throughout the whole simulation, clearly showing when the flock deals with the predator and when it returns to its normal cohesion. This can be seen in the steep rise in the graph in Figure 3.3, which show that the cohesion of the flock decreases as the predator approaches. It also shows that the flock's cohesion stabilizes after being disrupted. The flock of bird in my simulations shows similar behavior even in a adjusted environment following the behavior specifications of De Nicola et al. [7].

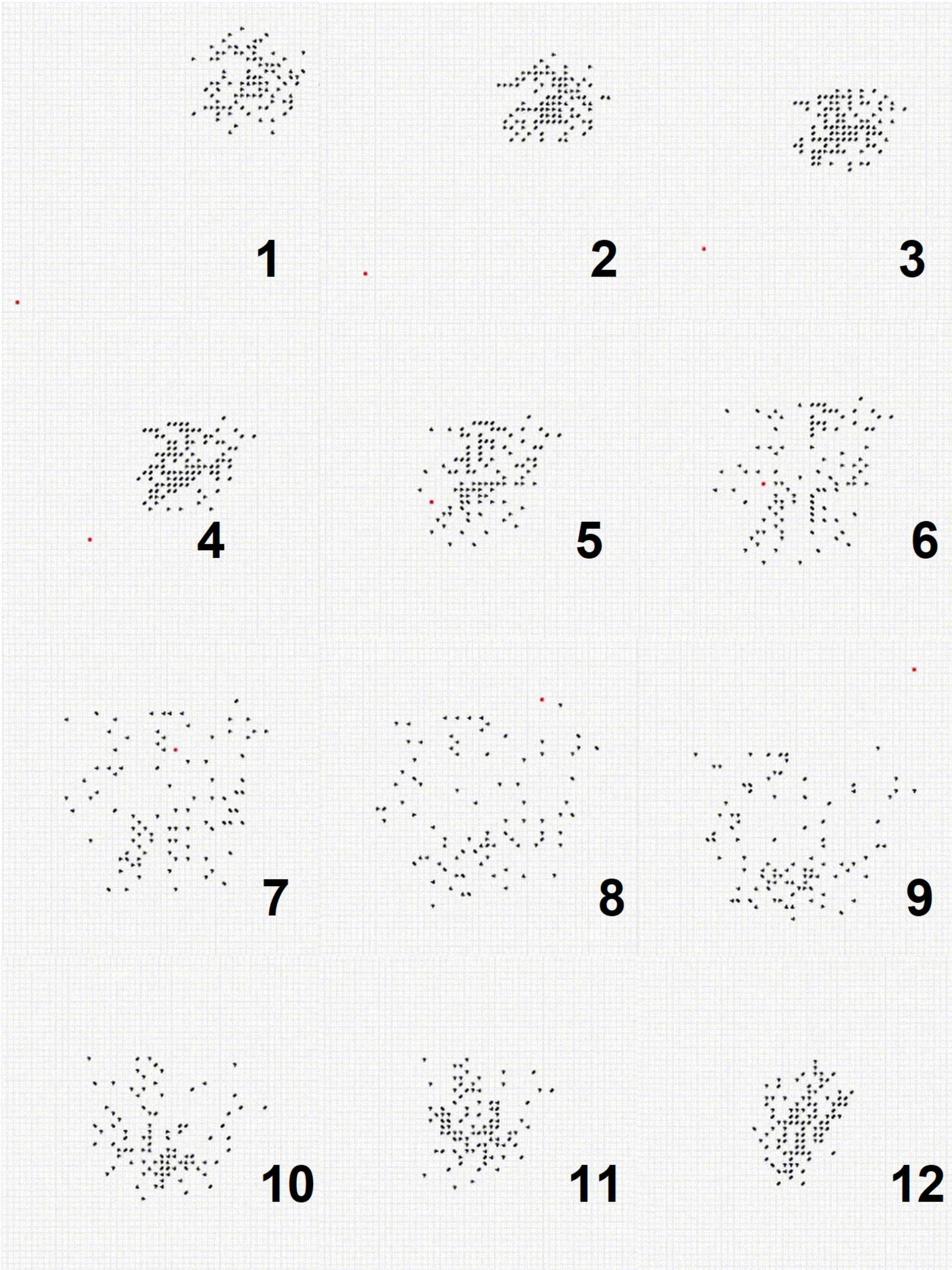

Figure 3.2: This figure shows a snapshot of a simulation's visual run, where black dots represent birds the red dot the predator. As the simulation proceeds the evasive maneuver of the birds can clearly be seen. The flock of birds also finds back together after the predator leaves. The numbers on this figure represent the time chronological sequence.

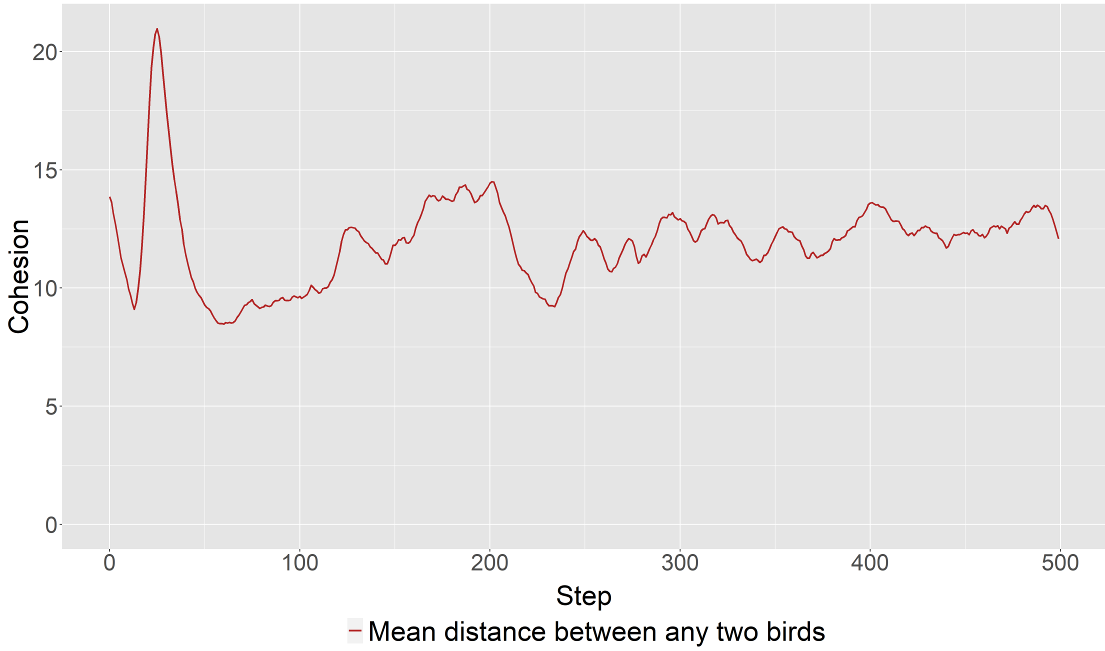

Figure 3.3: This figure shows the mean distance between any two birds plotted in red. The x-axis shows the steps of the simulation. The y-axis shows the Manhattan distance.

4 Conclusion

In this work, I highlighted what swarm intelligence is and how it is used in today's complex computational landscape. How they help with optimization problems and how inter-disciplinary the topic swarm intelligence actually is. I used three prominent examples that employ the concepts of swarm intelligence to showcase how simple behaviors can lead to the emergence of complex intelligent behavior.

To demonstrate that the basic concepts of swarm intelligence are simple and that recent research is replicable, I replicated the recent work of De Nicola et al. [7] and obtained conclusive results of intelligent swarm behavior. Their implementation and design descriptions were more than sufficient that it made their work easy to replicate. Them providing direct access to their code that was linked in their paper provided me with ample means to understand and replicate their work and results.

In that regard, many improvements can be made to my work in the future that would have exceeded the scope of this thesis. Some of those improvements include implementing smarter predators that use different prey-finding strategies and having the birds react to those strategies. Another improvement would be to compare larger batches of simulations and data whilst fine-tuning parameters to see their effects on the results. The latter of those improvements already lie withing the capabilities of the MESA framework.

All in all, swarm intelligence, in general will help solve many complex problems in the future, the more we continue to research it.

List of Figures

List of Tables

Bibliography

[1] I. L. Bajec and F. H. Heppner. "Organized flight in birds." In: *Animal Behaviour* 78.4 (Oct. 2009), pp. 777–789. DOI: 10.1016/j.anbehav.2009.07.007.

[2] G. Beni and J. Wang. "Swarm Intelligence in Cellular Robotic Systems." In: *NATO Advanced Workshop on Robots and Biological Systems* (June 1989).

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Oct. 1999. ISBN: 9780195131581. DOI: 10.1093/oso/9780195131581.001.0001.

[4] R. Bouffanais. *Design and control of Swarm Dynamics*. Springer, 2016.

[5] ChatGPT. *Swarm Intelligence*. OpenAI. 2021.

[6] A. Colorni, M. Dorigo, and V. Maniezzo. *Distributed Optimization by Ant Colonies*. 1991.

[7] R. De Nicola, L. Di Stefano, O. Inverso, and S. Valiani. "Modelling Flocks of Birds from the Bottom Up." In: *Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning: 11th International Symposium, ISoLA 2022, Rhodes, Greece, October 22–30, 2022, Proceedings, Part III*. Rhodes, Greece: Springer-Verlag, 2022, pp. 82–96. ISBN: 978-3-031-19758-1. DOI: 10.1007/978-3-031-19759-8_6.

[8] DeepL. *Translation of abstract from english to german*. https://www.deepl.com/translator. Translation generated using the DeepL translation service. 2023.

[9] J. L. Deneubourg, J. M. Pasteels, and J. .-. Verhaeghe. "Probabilistic behaviour in ants: A strategy of errors?" In: *Journal of Theoretical Biology* 105 (1983), pp. 259–271.

[10] L. M. Gambardella and M. Dorigo. "An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem." In: *INFORMS Journal on Computing* 12.3 (Aug. 2000), pp. 237–255. DOI: 10.1287/ijoc.12.3.237.12636.

[11] H. Hamann. *Schwarmintelligenz*. Springer Berlin Heidelberg, 2019. DOI: 10.1007/978-3-662-58961-8.

[12] J. Kennedy. "The particle swarm: social adaptation of knowledge." In: *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*. 1997, pp. 303–308. DOI: 10.1109/ICEC.1997.592326.

[13] J. Kennedy and R. Eberhart. "Particle swarm optimization." In: *Proceedings of ICNN'95 - International Conference on Neural Networks*. Vol. 4. 1995, 1942–1948 vol.4. DOI: 10.1109/ICNN.1995.488968.

[14] K. B. Korb and S. Mascaro. "The philosophy of computer simulation." In: *Logic, methodology and philosophy of science: proceedings of the thirteenth international congress*. College Publications. 2009, pp. 306–325.

[15] D. Masad and J. Kazil. "MESA: an agent-based modeling framework." In: *14th PYTHON in Science Conference*. Vol. 2015. Citeseer. 2015, pp. 53–60.

[16] S. Meshoul and M. C. Batouche. "Ant colony system with extremal dynamics for point matching and pose estimation." In: *Object recognition supported by user interaction for service robots* 3 (2002), 823–826 vol.3.

[17] M. M. Millonas. "Swarms, Phase Transitions, and Collective Intelligence." In: *arXiv: Adaptation and Self-Organizing Systems* (1993).

[18] R. S. Olson, P. B. Haley, F. C. Dyer, and C. Adami. "Exploring the evolution of a trade-off between vigilance and foraging in group-living organisms." In: *Royal Society Open Science* 2.9 (Sept. 2015), p. 150135. DOI: 10.1098/rsos.150135.

[19] R. S. Olson, A. Hintze, F. C. Dyer, D. B. Knoester, and C. Adami. "Predator confusion is sufficient to evolve swarming behaviour." In: *Journal of The Royal Society Interface* 10.85 (Aug. 2013), p. 20130305. DOI: 10.1098/rsif.2013.0305.

[20] C. W. Reynolds. "Flocks, Herds and Schools: A Distributed Behavioral Model." In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '87. New York, NY, USA: Association for Computing Machinery, 1987, pp. 25–34. ISBN: 0897912276. DOI: 10.1145/37401.37406.

[21] Y. Shi and R. Eberhart. "A modified particle swarm optimizer." In: *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*. 1998, pp. 69–73. DOI: 10.1109/ICEC.1998.699146.

[22] *Swarm intelligence definition and meaning: Collins english dictionary*.

[23] C. R. Tosh. "Which conditions promote negative density dependent selection on prey aggregations?" In: *Journal of Theoretical Biology* 281.1 (July 2011), pp. 24–30. DOI: 10.1016/j.jtbi.2011.04.014.

[24] P. Toth and D. Vigo. "Models, relaxations and exact approaches for the capacitated vehicle routing problem." In: *Discrete Applied Mathematics* 123.1-3 (Nov. 2002), pp. 487–512. DOI: 10.1016/s0166-218x(01)00351-1.

[25] W.-C. Yang and T. Schmickl. "Collective Motion as an Ultimate Effect in Crowded Selfish Herds." In: *Scientific Reports* 9.1 (Apr. 2019). DOI: 10.1038/s41598-019-43179-6.