Self-organized task allocation in a foraging swarm within a game simulation engine

Tomer Iwan
Supervisor: dr. ir. Eliseo Ferrante

A thesis presented for the degree of
Master of Science in Artificial Intelligence

Faculty of Science
Vrije Universiteit Amsterdam
The Netherlands
November 26, 2021

Abstract

In this work, we have built a simulation environment, focused on the survival of a swarm. The environment contains resources, which are used by agents to forage. We are interested in the effect of intra-population dynamics and external factors on the swarm. We propose a number of experiments in which the level of cooperativeness of the individuals are altered. Through this we aim to research whether being inactive and/or selfish could be beneficial for self-organized task allocation in a food foraging task.

We identify selfishness and laziness as two traits of uncooperative behaviours. In the former case, the individual keeps resources for itself, while in the latter it does not join in foraging. We perform experiments for each behaviour along with a combined approach. Furthermore we conduct experiments that look into intra-population dynamics.

The obtained results show that selfishness negatively affects the performance, while laziness has a positive effect. In a combined experiment, we found that mainly laziness affected the performance positively until a fully lazy population. Altering the sizes of the swarm and of the environment were found to have an effect as well. However, moving as a single unit, showed that also the selfishness did have an effect alongside laziness.

Overall, we can conclude to have obtained mixed results over the proposed experiments. We found discrepancies between the standalone experiments and those that combined two behaviours. Future work should concern itself with improving the simulation with the aim of obtaining more reliable results.

Keywords— Swarm Intelligence, Task Allocation, Foraging, Game AI

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

An emerging field, which explores the design of multi-agent systems, is swarm intelligence. It is heavily inspired by the observation of collective behaviour of social insects, such as ants and bees.

Ants, for instance, are shown to use pheromone trails to communicate indirectly with each other, in tasks such as foraging food. Bees, on the other hand, inform others in the swarm of new locations with an abundance of food, in a direct fashion through a specific dance. The collective behaviour is not limited to foraging alone, as social insects are also known to build nests in cooperation.

Accordingly, other animal societies also display a form of collective behaviours, as can be seen in flocks of birds and schools of fish.

Although the individual members of the swarm are simple beings, together they are able to carry out complex tasks. A key aspect of the collective behaviour is that it emerges from relatively simple actions and interactions between individuals.

As such, these decentralized collective behaviours have caught the eye of AI researchers. Therefore, the design of applications in SI adhere to similar constraints. As opposed to more traditional approaches, swarm intelligence does not make use of a sophisticated global controller to govern the behaviour of the system. Instead, just as in nature, individual controllers, carrying out unsophisticated behaviours are employed, in order to achieve the desired behaviour by means of cooperation.

This thesis is framed within the general context of Artificial Intelligence (AI), focusing on the development of swarm intelligence (SI) in multi-agent games. Early AI research concerned itself with the development of computational systems capable of displaying human-level intelligence. The aim was to apply this intelligence in tasks of problem solving and decision making. Such tasks were presented to the machines as a set of formal mathematical notations, which were able to be solved by means of symbol manipulation. Due to this highly formalized framing of the problems, early AI was able to succeed in a large number of tasks.

Games, in particular board games, have long been considered as the perfect research environment for AI methods. This is partly due to them being a complex human activity. However, the main reason is that they offer environments which are formal and highly constrained, suitable for decision making tasks.

Game AI (GAI) is a sub-field of AI, which came to be quickly after AI itself. It is a broad field, which is concerned with the challenge of developing AI for playing games at a high level, but also ventures into high level applications, such as the automated generation of game aspects.

With the simultaneous rapid development of AI and the introduction of video games, the area of research has seen an overwhelming number of advancements in the last couple of years. An early breakthrough occurred in the 1950s, when Shannon [1] developed a program for playing chess. Other multi-player board games have been researched, for example, Othello [2], Hex [3], Shogi [4], Go [5], Backgammon [6], Scrabble [7] and Checkers [8]. Perhaps the most well-known feat in this field, was the victory of Deep Blue over world chess champion Garry Kasparov in 1997. Another major achievement occurred in 2016 when DeepMind's AlphaGo defeated Lee Sedol in the game Go.

Numerous AI methods have been employed to tackle challenges in games, which include search algorithms, reinforcement- and deep learning, among others. Many of these applications, however, are concerned with the performance of a singular agent.

1.1 Motivation

This thesis is motivated by the following quote by John McCarthy.

> *"The computer game Lemmings can serve as a new Drosophila for AI research."*

The quote by one of the founders of the AI field, suggested that as the game of Chess was largely solved, researchers should look into the game of *Lemmings*.

This game in which a population of agents needs to find its way through a number of environments, could prove interesting in the sub-field of swarm intelligence as well. The agents need to cooperate in order to achieve a common goal, which would be near impossible to do on their own.

The puzzle game has one caveat in its suitability for swarm intelligence research. Namely, that the agents are not equal in their capabilities. If it were the case that each Lemming could carry out all tasks, then without a shred of doubt the game would be a perfect testbed for research in swarm intelligence.

Despite this limitation, the game still is considered as a source of inspiration for this thesis. However, the main motivation is biological systems. In particular the dynamics within natural swarms and the way they interact are of interest. As with *Lemmings*, the members of natural swarms have a common goal, which often only can be achieved by cooperation. The impressive feat of these social animals is that they appear to demonstrate collective behaviour despite the lack of a leader. Indeed, the behaviours of interest emerge in a decentralized fashion.

In this thesis we shall present research on a game environment, which is inspired by foraging in nature. We will look into the dynamics within a swarm of agents with equal capabilities. Such intra-swarm dynamics include the willingness to cooperate, which we will manipulate by acting lazy or selfish. The former case can be compared to brood in a nest. When animals are too young to care for themselves, they cannot go out and forage food. Older members of the population need to take it upon themselves to bring the young ones food to grow strong.

In the selfish case, we simply consider the possibility that some members simply care more about their well-being, than that of the collective. These agents do care for themselves and look for resources in the environment, but are not willing to share them. We hope to find whether these dynamics will lead to changes in the performance of the population in the simulation runs.

1.2 Research goals

This thesis will address the following research goal.

- We aim to research whether being inactive and/or selfish could be beneficial for self-organized task allocation in a food foraging task.

The main goal of this thesis is to develop a simulation environment which could serve as a testbed for swarm intelligence research. The current available engines rely either on high cpu usage or are a flat 2D representation. To our knowledge the implementation presented in this thesis would be the first self-contained environment built in the Pygame python module, making use of a faux 3D projection.

In this simulation, the agents of a swarm need to work together in order to achieve the common goal of survival. This collective behaviour should emerge from simple behaviours by the individuals. An important note to make is that a decentralized

approach is emphasized. We do not want to have a leader which steers the population, as this would go against the principles of swarm intelligence.

In line with researching the swarm behaviour we would like to show how intra-swarm dynamics could influence the performance of the swarm in this collective goal. These dynamics follow from hierarchical ranks in the swarms, such as the presence of a brood. When young, members cannot contribute to the collective and place the additional burden of needing to be cared for.

Another way in which the dynamics could influence the swarm, is greediness. An agent could prefer its own preservation over that of its peers. Therefore, such an agent will not concern itself with sharing resources with the others. However, it will try to profit from efforts of the swarm, whenever possible.

In addition to the main objectives, we would like to motivate the implementation of swarm intelligence in video games. Especially, in real time strategy games (RTS), in which the player controls a large number of agents, we believe that a decentralized approach could be applied as such games are challenging in terms of reaction time. The objective is paired with a hypothetical approach in the simulation, in which some logic is thrown out the window. In this proposed approach traditional foraging and local sharing is replaced by a global sharing mechanism. Each collected resource can, through a centralized storage, reach all the members of the swarm.

1.3 Thesis structure

The thesis is organised as follows. In chapter 2 background information is presented.

First, we define the basic concepts of swarm intelligence. These include basic behaviours such as **foraging**, **flocking**, and **aggregation**. This is followed by a description of self-organized behaviour, in particular task allocation. The driving force behind actions in a swarm, Finite State Machines (FSM) is explained next.

We follow this with a historic overview on Game AI, highlighting methods and major achievements.

Next, in chapter 3 the methods are presented. First, the game objectives are described. Afterwards, the methods to build the game are described, which include the environment, camera and physics. Further elements of the game are described in detail such as agents, the swarm and resources.

The overall approach of the experiments are presented in chapter 4. From chapter 5.1 up to chapter 5.4, the various experiments are delved into.

First, the selfishness experiment (chapter 5.1) is conducted to observe the importance of sharing. Next, in chapter 5.2, we allow a specified number of agents to act lazily. Such agents do not look for food themselves, but rather stay back in the hope of receiving some food from their peers. Following these two experiments, a combined approach is proposed. Chapter 5.3 aims to research if an optimal ratio exists for a swarm to act lazily and selfishly. In the final experiment (chapter 5.4) the entire swarm moves as a single unit.

An overall discussion is presented in chapter 6. We will go over the obtained results from the experimental chapters, coupling them to each other and aim to explain how they came to be. Furthermore, we will look into the methodology and experimental setups, highlighting what worked and what did not. Next, we will look into the limitations of the proposed methods and will offer suggestions for future work.

Finally, the thesis is concluded in chapter 7 in which we give a brief summary of the thesis and present the main take-aways.

2 Related work

This chapter starts with the introduction of the swarm intelligence field. We show how the field is influenced by nature, highlighting phenomena in the wild. Next, we present a formal definition of the swarm to be considered intelligent. Furthermore, we present major achievements in this field. This is followed by a description of Finite State Machines (FSM), as our implementation will rely heavily on its implementation.

Next, we introduce the field of Game AI (GAI). We start with a brief overview of the achievements in this sub-field of AI. Lastly, we present related work, which touch both on swarm intelligence and Game AI.

2.1 Swarm Intelligence

Swarm intelligence is heavily inspired by social animals. The basis of the field is on emergent behaviour that such animal societies display. From observations by biologists, it became evident that interactions between members of the swarm were integral to their ability to carry out complex tasks. Such tasks were considered to be too difficult to be performed by the individual. However, as the individuals performed simple tasks, the collective was able to achieve common goals. The phenomenon, in which individual labor leads to complex dynamics, is called emergence. Emergence is at the base of swarm intelligence. We will present below a number of examples in which different social organisms display feats of collective behaviour through emergence.

Fish are well-known to swim in schools. This self-organized behaviour also has the advantage of protection from predators, as there exists a sense of safety in numbers. Hypothesized has been that the large number of fish in the school, lead to a visual sensory overload for the predator [9].

A collective behaviour known as flocking [10] is well known in birds. When birds are in flight, they often appear to be in a "V" formation. This group behaviour emerges from simple individual behaviours.

Regarding collective behaviour, social insects in particular, have caught the eyes of biologists. Observed can be that these species can coordinate their actions, in order to accomplish tasks the individual could not.

Perhaps the single most influential social animal, in the field of swarm intelligence, is the ant. Ants show collective behaviour when foraging. When moving between their nest and resources, they search for the shortest path. By means of chemical pheromone trails, the ants are able to communicate, informing each other of the ideal path.

The theory is underlined with the following example. Assume two paths that lead the swarm from their home to a resource and back. Those ants that took the shortest path will return sooner to the base. During this back and forth, the individual releases a pheromone trail. This attracts others in the swarm, causing the path to accumulate a high concentration of pheromones [11].

See Figure 2.1 for a visual representation of the path finding phenomenon.

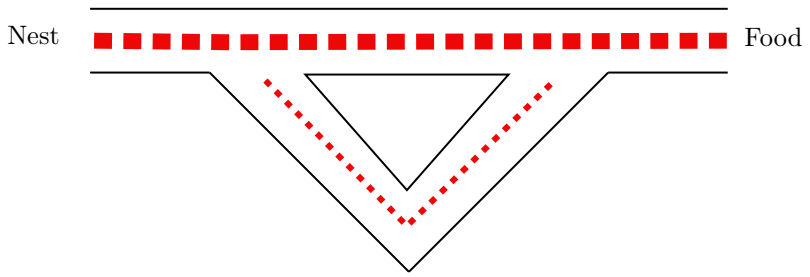

Figure 2.1: Visual representation of path finding in ant colonies. Starting from the nest there are two paths identified towards food. The red dots represent the concentration of the pheromone trails. The shortest path in this case being the straight path therefore has the highest concentration. (Adapted from Goss et al., 1989.)

This behaviour has been researched extensively, and has led to one of the first algorithms inspired by swarm intelligence. Ant colony optimization (ACO), is used in applications of path finding in graphs. Over the years, many variations of this algorithm have been proposed.

Furthermore, ants are known to collectively carry large prey. While a single ant would not be able to support the weight of a much larger animal, the collective moves in a formation in which the burden is divided among the members [12].

Termites have been found to build domes by following two simple rules. They need to move in the direction with the strongest pheromone concentration. Once there, they deposit the dirt at the point where the smell is strongest. This can be considered as being a self-organized system. There is no central control. All members contribute to a common goal. The collective behaviour emerges from simple tasks [13].

Honeybees employ a dance to communicate the location of food sources. It is believed they have the ability to distinguish between different dances. The specific dance patterns help communicate to other bees the direction and quantity of resources found. Wenner and Wells [14] suggested that in addition to the two pieces of information, the dances also communicate the distance. Their range has been found to be as large as ten kilometers in distance. Honeycomb construction can be attributed to coordinated behaviour of the swarm.

2.1.1 Foraging

The act of foraging is wide-spread in nature. Logically, as all animals require nutrition to survive. As mentioned earlier, ants make use of pheromone trails when searching for food.

Foraging in swarm intelligence is not limited to food only. It can be considered as a broad behaviour, which can be boiled down to a seek and retrieval phase.

Therefore, we could also consider the behaviour of rescue swarms to be a part of foraging. Such swarms seek the injured, and bring them back in order to receive treatment.

2.1.2 Flocking

When a group of birds, called a flock, moves in the well-known V-shape, we speak of the collective behaviour called Flocking. Flocking emerges from simple rules performed by the individual despite the absence of a leader. The paper by Reynolds, titled *boids*, modelled the flocking of birds [15]. He made use of the following three rules:

- The agent should avoid crowding its neighbours
- The agent should steer towards the average heading of its neighbours
- The agent should steer towards the average position of its neighbours

The first rule, better known as **separation**, is implemented in order to achieve short range repulsion. It is a behaviour best observed in a large school of fish. When the numbers become very large, there exists little space between the individuals. The school becomes dense with fish. Yet, despite this crowded situation, the members manage to avoid collisions among themselves.

Next, the **alignment** behaviour ensures that the members of the swarm move into a single direction.

Lastly, **cohesion** ensures that agents can join the swarm from long range. The V shape observed in flocks of birds is an excellent example of the latter two behaviours.

Through alignment, all members of the flock move into the same direction resulting in the shape. By means of cohesion, more birds join and the V formation becomes more pronounced, as they join the "ends".

It is important to note that these three rules are entirely local. They refer only to what an individual can observe and carry out.

2.1.3 Aggregation

In aggregation, all group members move into a specific region of the environment. It is a widespread collective behaviour in nature, as it can be observed in various insect species, birds, fish and even bacteria [16]. This simple, but very useful behaviour, enables members of the swarm to get close enough in order to interact.

2.1.4 Task allocation

Above we have introduced certain behaviours that can be found in social societies. However, what we have presented are single behaviours. In task allocation, the swarm displays a form of labor division. It can be observed in social insect colonies, such as ants [17, Chapter 3] and bees [18]. An example of task allocation is when part of the swarm performs foraging, while the others stay behind to look after the young. The division of labor depends often on factors such as the morphology or age of the individual. Of course, chance could also play a role.

However, in swarm intelligence applications, this mechanism of preset tasks is too rigid. Rather, the agents in an artificial swarm should dynamically distribute themselves over a range of tasks. Each member here should be equally capable to carry out any task. The goal is, of course, comparable to natural swarms. Through task allocation the aim is to maximize the performance of the swarm.

We can distinguish two methods of task allocation. One being offline, and the other online. In the former approach, the tasks are distributed before starting the procedure. Accordingly, the online method performs task allocation during the run. While the offline approach is simpler, it cannot adapt to changes in the environment. Therefore,

if the procedure involves a dynamic environment, the online approach should be preferred. This method comes at the cost of devising an efficient way for switching tasks. We should avoid switching often, as this will harm the efficiency of the swarm.

Consider a large swarm which limits itself to carrying out two tasks only. A number of members will do task A, while the other will perform task B. Then we can view the swarm as a cluster of agents, which are able to break off into smaller groups. Together these two groups form one large swarm, but at the moment of performing their respective tasks, they interact within their specialized groups. A visual representation is provided in Figure 2.2, along with a mathematical representation in Equation (1).

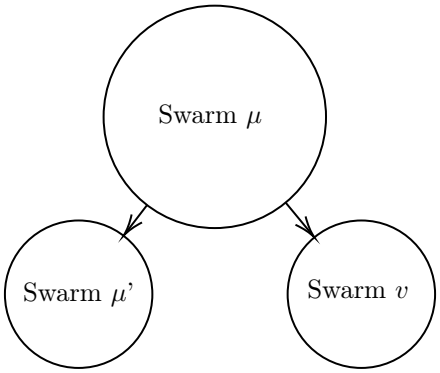

Figure 2.2: Swarm splitting in clusters

$$Swarm^{\mu}(new) \cup Swarm^{v} = Swarm^{\mu} \tag{1}$$

2.2 Game AI

Game AI (GAI) is a sub-field of artificial intelligence, in which algorithms are applied in game settings. The algorithms were not solely developed for these purposes, as the majority of them made their way to other fields, such as computer graphics, search problems and general robotics.

2.2.1 Achievements of AI in games

Game AI has a rich history, tracing back to the inception of the AI field itself. Therefore, it can be considered an integral part of the field. In the early days, research was focused on classic board games, such as backgammon, checkers and chess. The reason being that these games, while being composed of simple rules, presented a high level of complexity. As such they have challenged the best human minds since their introduction.

See Table 2.1 for a recap of the timeline of game AI.

Table 2.1 Timeline of game AI

1956	The founding event of AI at The Dartmouth Summer Research Project on Artificial Intelligence
1992	**Backgammon**: TD-Gammon was developed, relying on ANN
1994	**Checkers**: Chinook defeats Marion Tinsley
1996	**Chess**: Garry Kasparov defeats IBM's Deep Blue (4 - 2)
1997	**Chess**: IBM's Deep Blue defeats Garry Kasparov (3.5 - 2.5)
2014	Google Deepmind uses raw pixel input to play Atari games
2016	**Go**: AlphaGo defeats Lee Sedol (4 - 1)
2017	**Go**: AlphaGo defeats Ke Jie (3 - 0)

In 1992, the backgammon software capable of playing at professional level, named TD-Gammon, was developed by Gerald Tesauro. Playing against itself millions of times, the underlying artificial neural network was trained in order to achieve that level.

Two years later, Chinook, the checkers software, managed to defeat the world checkers champion Marion Tinsley. Chinook's achievement was based purely on domain knowledge by the creators, as the tree search used went through the pre-programmed actions.

Chess, in particular, long held the honour of being considered the "drosophila of AI", as it was considered to be a perfect testing ground for novel AI methods.

IBM developed the first software capable of displaying human-level chess playing, named *Deep Blue*. Deep Blue relied on a Minimax algorithm running on a supercomputer. Furthermore, the software was given chess specific modifications. IBM first attempted to defeat the chess grandmaster Garry Kasparov in 1996. During this match, the world champion managed to defeat Deep Blue, winning four out of six rounds. The rematch in the following year, saw IBM's Deep Blue emerge victorious, by winning twice, losing once and drawing three times in six rounds. This historic event, is by many considered the day machines surpassed human intelligence.

Nowadays, the computing power of supercomputers is no longer necessary to play better than human players. The software can be downloaded and run on the vast majority of devices, such as laptops and smartphones.

With the game of chess being considered solved, the game of *Go* was proposed to become the new benchmark for game-playing AI, due to its much larger search space

compared to chess. It was also considered to be the most challenging board game.

Two decades after the Deep Blue - Kasparov game, Google Deepmind's AlphaGo managed to defeat Lee Sedol in the game of Go. The game, taking place in 2016, was won with a convincing score of 4 to 1. The very next year, AlphaGo outclassed Ke Jie, then ranked number one in the world, by winning all three games. The reason for its success was the novel method of deep reinforcement learning. In contrast to Deep Blue, Alpha Go was run on a single computer.

Now that the most complex board game was solved, there was no merit to create and solve a new board game. A more complex game would not be interesting for human players, and any achievement by a computer on such games would hold little significance. Luckily for the field of game AI, there is more to games than the classical turn-based board games.

With the booming gaming industry bringing out large numbers of video games, there are endless new possibilities to develop AI to be able to play them.

Video games, as opposed to board games, require a different kind of intelligence. Instead of turn based decision making, the software needs to be able to decide in real time. As such they pose an interesting testing field for AI. John McCarthy, one of the founding fathers of AI, [19] even singled out the *Lemmings* game as an important game in the field of Game AI, as early as 1998. A couple of years later, Kendall et al. [20] used the game to demonstrate its relevance to Game AI.

In 2014, Google Deepmind developed a number of algorithms, which were able to use raw pixel input to play Atari games [21]. These algorithms took inspiration from the achievement of TD-Gammon. Deepmind however took reinforcement learning one step further, as their approach was based around deep reinforcement learning.

Having introduced the concepts of swarm intelligence and Game AI we aim to present work related to a combination of these techniques. Our focus will be mainly on the implementation of task allocation.

A widely used method to combine collective behaviour and finite state machines, is the response threshold function. The transition probability, p, is calculated for each state as presented in Equation (2). It takes into account the variables s and θ, which are the stimulus and stimulus threshold, respectively. Furthemore, the sensitivity parameter is depicted by the β. As can be observed from the equation, the function is non-linear. The transition probability is low when the stimulus is much smaller than its threshold. Alternatively, p is very high when the stimulus is considerably larger.

The method was pioneered by Granovetter in 1978 [22]. For years it had been applied to problems concerning the labour division of social insect colonies [23].

$$p = \frac{s^\beta}{\theta^\beta + s^\beta} \tag{2}$$

Krieger and Billeter [24], applied a task allocation mechanism based on individual activation-thresholds for a food foraging task. The threshold was linked to the perceived energy level of the nest. When this level dropped below the threshold, certain agents were to collect food items. Demonstrated was that the division of labour occured in a decentralized manner. Basing their method around the perceived foraging behaviour of ants, the authors observed the agents working efficiently towards their goal. The energy levels achieved by the swarm was higher when the group cooperated, then when individuals foraged alone. However, they did observe a diminishing return, as perceived benefit decreased in large populations. The authors discussed that interference during foraging was the cause.

Yang et al. demonstrated that a simple threshold model could be effective in task allocation [25]. The authors set out to tackle a foraging problem. The paper presents a model in which the agents decide to seek food, based on thresholds concerning the home base. These thresholds are the food density and the food consumption rate. Their experiment starts with all members of the swarm at the home base, which already has some food at the beginning of the simulation. Based on a preset food consumption rate, the food density at the base drops. Throughout the simulation the food density in the simulation environment is kept constant. Randomly new instances of food are spawned in the foraging area. The authors found that the proposed method allowed the population to carry out task allocation in an efficient manner. By increasing the threshold values, they also observed that the performance improved, due to lower interference between the agents.

Tavares et al. [26] modelled the game *StarCraft* as a task allocation problem. The game is part of the genre of Real Time Strategy (RTS) games. These games are known to be played at a fast pace, and as such are quite challenging. The player is tasked with leading a population consisting of three races to victory. To ensure victory all buildings of the opponent need to be destroyed. As mentioned the player is given three races. These have different characteristics, such as capabilities and resources needed. The authors applied an out-of-the box algorithm, swarm-gap, in order to demonstrate task allocation. The main driving force behind the performance in this paper, is the genetic algorithm responsible for configuring the swarm-gap parameters.

3 Methodology

In this section we will go through the elements of the simulation environment. The engine was created entirely using python with the PyGame module. Assets, were obtained from an external website. [1]

3.1 The game

The objective of the game is to survive in the wilderness. A time limit is given, that in-game coincides with one day. In order to survive, agents need to collect resources which spawn randomly in the environment. These resources are depicted as mushrooms.

A world clock is displayed, in order to know how far into the simulation we are. Furthermore, in the top left corner of the screen, information about the simulation is displayed. These include the current swarm size, amount of resources in the environment and the amount in transit.

3.2 The environment

The environment consists of a grid, in which cuboids are drawn in the isometric projection, in order to achieve a faux 3D environment, or more appropriately, 2.5D environment. These individual cuboids are referred to as the grass tiles. Additionally, the boundaries of the environment were assigned a collision variable. This variable was used to detect obstacle collisions between sprites and the environment and to keep said sprites within the boundaries of the environment. Lastly, a base location, depicted by a large red flag, is spawned together with the agents.

3.3 Camera

A scrolling camera was implemented to accommodate larger maps. Based on the position of the mouse on the screen, the camera can scroll in any of the four directions. The offset created by scrolling was added dynamically to the sprites in the environment, in order to correct their position.

3.4 Agent

An agent is depicted by a sprite. Each agent has its own health level, which gets depleted during the simulation. The mushrooms can replenish the health of the agent. Hovering with the mouse on a specific agent, will display its health bar.

Furthermore, the agents are assigned a predefined maximum and minimum speed. These will be depicted as v_max and v_min from now on.

Agents also have a mass, which is taken into account when steering.

Weights are utilized for steering behaviours. These include the wandering, alignment, cohesion and separation behaviours.

Furthermore, the agents need to take into account the wandering radius and angle.

Each agent in the environment is controlled by a Finite State Machine. The behaviours are triggered in these FSM by thresholds and boolean expressions. The FSM of the agents is depicted below in Figure 3.1.

Regarding the movement of the agents, two driving forces can be identified. These are **action selection** and **steering**. The former determines the immediate goal to be

[1] www.kenney.nl

achieved by the agent, as can be seen in the FSM. Ultimately, to achieve the goal set by action selection, steering is needed. The mechanism is responsible for calculating the desired trajectories required to carry out the selected action. For steering to be applied, a steering force is necessary. This force describes the direction and velocity of the desired displacement.

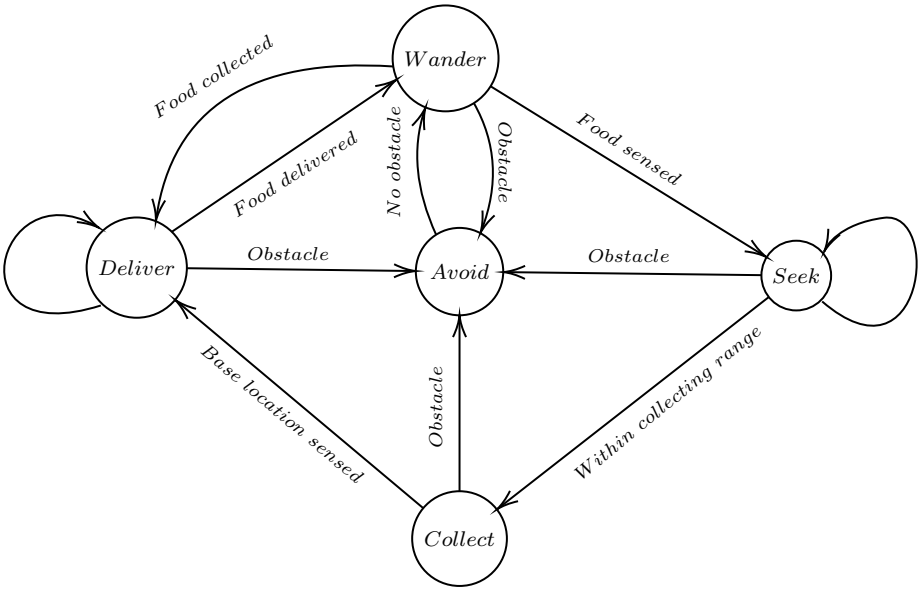

Figure 3.1: The finite state machine of the individual agents.

When an agent acts selfishly, the FSM changes as depicted in Figure 3.2. The main difference is that instead of moving into a delivery state, the agent consumes the resource for itself.

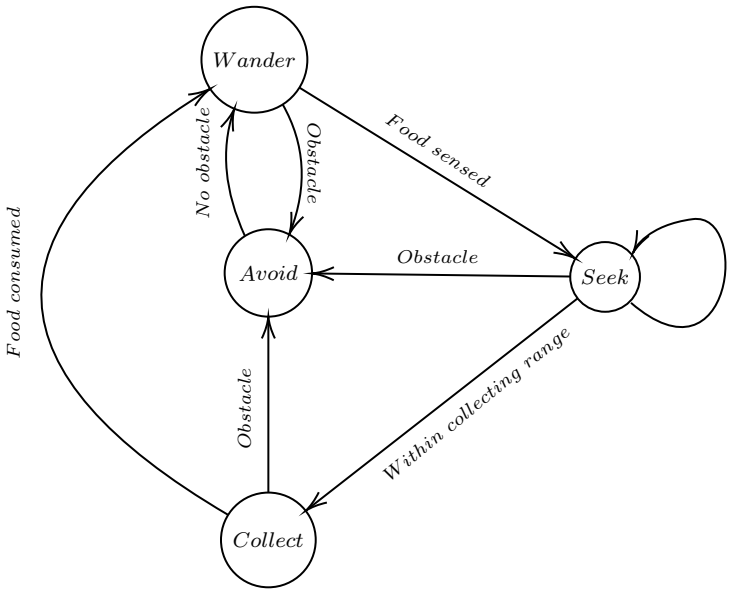

Figure 3.2: The finite state machine of the individual agents in the selfishness experiment.

3.5 Behaviours

Each agent has the ability to carry out a number of behaviours. The priority for carrying out the behaviours depend on the environment. When the agent fails to detect a resource, it will wander in the environment.

If the agent reaches the boundaries of the map, the aim is to avoid step out of bounds. Failing to do so, it takes damage until it steers away from the edge of the map

When food is available, the agent starts foraging, see Algorithm 3.1. An agent walks towards the food item, and collects it. The agent that collected the mushroom, switches from foraging to delivering. With a delivering task, the agent is entrusted to bring the mushroom back to the base. When enough food is available at the base, the agents can return and stay at the base to replenish their health. In all of the scenarios, the movements of the agents are affected by forces derived from their neighbors. These forces include alignment forces, cohesion forces and separation forces.

The behaviours are driven by vector manipulations. A standard rotation matrix, Equation (3), is used to rotate the matrix by a random θ.

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3}$$

The vectors are normalized (Equation (4)) to become $\hat{\mathbf{u}}$, which in the equations will be used as the function *normalize*.

$$\hat{\mathbf{u}} = \frac{\mathbf{u}}{|\mathbf{u}|} \tag{4}$$

Furthermore the euclidean norm, Equation (5) is used as the *norm* function

$$\|\mathbf{x}\|_2 := \sqrt{x_1^2 + \cdots + x_n^2} \tag{5}$$

Seek

The **seek** steering behaviour returns a force that directs an agent toward a target position. The behaviour is set in motion when an agent senses a mushroom in the environment. Implemented is that a velocity is calculated, which takes into account the current position of the agent and that of the target. After normalizing the resulting vector, the current velocity of the agent is subtracted. This leaves us with the desired velocity in order to move an agent towards a target.

Algorithm 3.1 Seek steering behavior

```
1: desired_velocity = target_pos - agent_pos
2: desired_velocity = normalize(desired_velocity)
3: desired_velocity = desired_velocity - agent_velocity
4: return desired_velocity
```

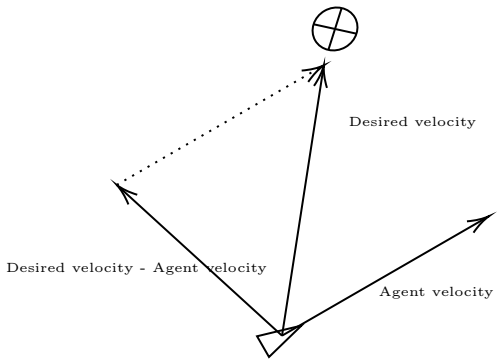

Figure 3.3: Vectors of the velocities

Deliver

The **deliver** steering behaviour is triggered when an agent collects a mushroom. A force is returned that steers the agent towards the home base. It is therefore, in regards to vector calculus, similar to the seek behaviour. The only difference is whether the agent is in the state of delivery, as displayed in the FSM. While the agent has not reached the base, and delivered the collected food, it makes its way towards the base.

Algorithm 3.2 Food delivering steering behavior

```
1: if Food collected then
2:     desired_velocity = agent_pos - base_pos
3:     desired_velocity = normalize(desired_velocity)
4:     desired_velocity = desired_velocity - agent_velocity
5:     return desired_velocity
6: end if
```

Obstacle avoidance

Obstacle avoidance aims to steer agents away from obstacles that are in their path. The identification of two tiles are necessary for this behaviour; namely the tile on which the obstacle is placed and the tile on which the agent currently is located. If the trajectory of the agent is towards the obstacle tile, and the distance is less than one tile; then the obstacle avoidance behaviour is triggered.

In our implementation, the tiles are assigned a boolean value whether an obstacle is present or not. This is used in the decision making mechanism of the individual agents.

Algorithm 3.3 Tile based obstacle avoidance steering behavior

```
1: if agent_pos on obstacle_tile then
2:     agent_velocity = rotate(normalize(agent_velocity))*norm(agent_velocity)
   ▷ See Equation (5).
3:     agent_pos = agent_pos + agent_velocity
4: end if
```

Wander

The **wander** steering behaviour returns a force that directs an agent towards a random location. The behaviour is influenced by the following behaviours and variables.

Algorithm 3.4 Wandering steering behavior

```
1: normalized_velocity = normalize(velocity)
2: circle_center = normalized_velocity * wander_dist
3: displacement = R(θ)·(normalized_velocity * wander_radius)        ▷ See
   Equation (3).
4: wander_force = circle_center + displacement
5: return wander_force
```

Next, we move into the group behaviours. These three behaviours below together lead to the phenomenon of flocking. To determine the steering force, the agent needs to consider all other agents within a predefined radius. Those other agents are considered to be the direct neighbours of that agent. All other agents, those outside the *neighborhood radius*, are not taken into consideration.

All three behaviours make use of the following method (Algorithm 3.5) to find neighbours.

Algorithm 3.5 FindNeighbours algorithm

```
1: neighbours = []
2: for agent in agentList do
3:     for self, agent in agentList do
4:         if self == agent then
5:             continue
6:         else
7:             neighbours.append(agent)
8:         end if
9:     end for
10: end for
11: return neighbours
```

Alignment

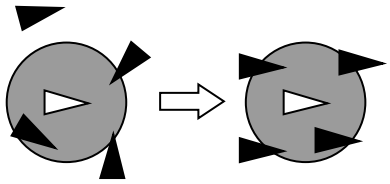

Figure 3.4: Alignment (Adapted from Reynolds (1987))

In **alignment**, the velocity of an agent is manipulated to match that of its neighbours.

Algorithm 3.6 Alignment collective steering behavior

```
1: FindNeighbours(agent_pos)
2: if LENGTH Neighbours > 0 then
3:     average_heading = average_heading + neighbours_heading
4:     vector_to_agent = normalize(vector_to_agent)
5:     steering_force = steering_force + vector_to_agent / LENGTH vec-
   tor_to_agent
6: end if
7: return average_heading
```

Cohesion

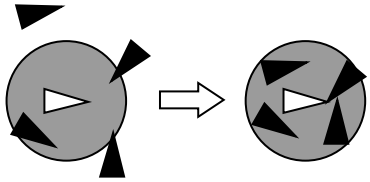

Figure 3.5: Cohesion (Adapted from Reynolds (1987))

The **cohesion** collective behaviour steers each agent to the geometric center of all nearby agents.

Algorithm 3.7 Cohesion collective steering behavior

```
1: FindNeighbours(agent_pos)
2: if LENGTH Neighbours > 0 then
3:     center_of_mass = center_of_mass + neighbours_pos
4:     steering_force = Seek(center_of_mass)
5: end if
6: return steering_force
```

Separation

Separation attempts to steer agents away from others, in order to prevent them crowding each other. The algorithm iterates over all the perceived neighbours of the agent. The vector to each neighbour is then normalized and divided by the distance to the neighbour. This vector is then added to the steering force.

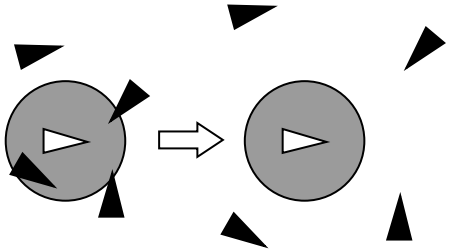

Figure 3.6: Separation (Adapted from Reynolds (1987))

Algorithm 3.8 Separation collective steering behavior

```
1: FindNeighbours(agent_pos)
2: if LENGTH Neighbours > 0 then
3:     vector_to_agent = agent_pos - neighbours_pos
4:     vector_to_agent = normalize(vector_to_agent)
5:     steering_force = steering_force + vector_to_agent / LENGTH vector_to_agent
6: end if
7: return steering_force
```

3.6 Swarm

The agents were grouped into a sprite group, called the swarm. Based on the behaviours introduced above, the agents decide whether to move as a single unit or break off and wander the environment on their own. When an agent in the swarm reaches a health level of zero, it dies and is removed from the swarm.

3.7 Food

The available food in the wilderness are mushrooms. These are spawned randomly in the environment. Consuming these mushrooms replenishes the health of the agents.

The only constraints for spawning are that food cannot spawn on the tile of the base or on the tiles on the edge of the environment.

4 Experimental setups

In this section, the experimental setups are introduced. These setups have been designed to observe how different settings can affect the behaviour of the agents, and in turn how it may affect the survival of the swarm.

Each experimental chapter will begin with a short introduction, explaining the experiment and providing the rationale behind it. Next, we delve into the methodology. After which we present the results and a statistical test. Finally, the chapter ends with a discussion.

As shown in chapter 3, each agent can be expected to be involved in the act of foraging. Logically, it would be possible that agents could also not be involved. In the context of the simulation this non-involvement can be achieved in two ways. The most obvious way is for the agent to not go out and look for food at all. Another way would be to come along, collect food and not share it with the rest. These ideas form the basis for all the experiments.

The first experiment will look into the effects of selfishness. Two setups are proposed. The first setup will look into the case in which the entire swarm acts either cooperative or selfish. The other setup will allow a part of the population to be selfish.

Next, we go into the laziness experiment. This experiment can be compared to the second setup of the previous experiment. Here again we will make use of a degree. The only difference being that instead of selfishness, laziness is researched. Laziness in this case being that an agent will not actively look for resources in the environment.

The third experiment combines the two degrees of experiments one and two. Both the degree of laziness and the degree of selfishness will be looked into. This experiment will have four different setups, which will alter the population and map sizes. Two setups will make use of a default map size of 20x20 tiles, while the other two will use a larger map (30x30 tiles). Each of these pairs of setups will have once a default population size of 20, and once a larger number of agents, namely 50.

The next experiment requires active agents to move as a single unit. Here we again look into the degree of laziness.

The final experiment is based on the swarm splitting phenomena. Agents are only to engage in foraging if they are part of a group consisting of a minimum number of agents.

The metrics presented for each experiment will be the survival rate and the fitness. The former represents the number of agents that are left over at the end of the simualtion run. The fitness is calculated based on three pieces of information. These are the survival rate, initial swarm size and the average health of the suriving agents. Combining these three metrics we calculate the fitness as follows.

$$fitness = \frac{survival\ rate\ *\ average\ health}{initial\ swarm\ size}$$

4.1 Assumptions

The following assumptions have been applied:

- The agent can detect the presence of a mushroom within a predefined radius.
- The agent can detect the presence of a neighbour within a predefined radius.
- The agent has a global knowledge of the base location.
- Without consuming a (part of a) mushroom, an agent will not survive the simulation.

- Agents that step outside of bounds are killed off.
- Mushrooms cannot spawn at the base.
- Mushrooms cannot spawn on the edges of the environment.
- Mushrooms spawn randomly.

Due to the stochastic nature in which the resources are spawned in the environment, multiple runs on the same problem are necessary to get a good estimation of performance. We have chosen to run each experiment twenty times for each of the settings.

4.2 Statistical tests

In order to acquire evidence to either accept or reject the hypotheses in the experimental chapters, we propose the following statistical tests. The significance level of test, α is set to 0.05 for all the statistical tests conducted. In experiments in which two strategies are compared we will employ Student's t-test or Welch's t-test based on whether the groups are normally distributed. When more than two are compared we move onto either One-Way Anova with post-hoc Tukey test or Kruskal-Wallis test with post-hoc Dunn test.

Finally, in our post-hoc tests, we choose the conservative method of p-adjustments with the Bonferroni correction [27, 28]. The method simply divides the p-value by the number of tests that are compared.

5 Experiments

5.1 Experiment Selfishness

In this chapter we present the first experiment. We will look into the effect of having selfish individuals in the swarm will have in the performance. Through this we hope to answer the following research question.

- *RQ1. What is the effect of having selfish agents on the performance in the food foraging task?*

5.1.1 Motivation

In populations it is not out of the question that individuals may choose to not contribute to the collective. There may be factors involved such as the preference of ensuring self-preservation. An agent in the swarm could be on the brink of death. In that case, it would probably prefer to take all actions possible to ensure its own survival.

Intra-population dynamics could hypothetically also play a part. For this example, consider a herd of animals, in which only a small number of females are present. The need to be the alpha male in the herd is then a legitimate reason to abstain from sharing resources with its competitors.

Finally, we could identify gluttony as a reason to not contribute to the swarm. An agent could well enjoy consuming large amounts of food and can simply be described as being greedy.

This experiment concerns itself with selfish behaviour. When collecting a food item, agents should have the choice to either share or to keep the food for themselves. The aim of the experiment is to research whether a strategy based on sharing is preferred over being selfish within a swarm.

By omitting the delivery phase, i.e. bringing back the food to the base, the agent prefers an immediate reward. We must remember that the non-selfish implementation requires agents to be at the base to receive part of the resource. Also, all agents are out in the simulation looking for food. This makes it less likely that a large number of agents will be at the base at the moment of delivery. Therefore, there exists a chance that the selfish strategy may be able to match or even outperform the sharing strategy.

On the other hand, having an agent consume the resource immediately, allows that same agent to immediately continue foraging. In the cooperative case, the agent is busy with performing delivery and as such cannot continue foraging until the food has been brought back to base. This implication could lead to the situation that an agent, in the selfish strategy, snatches consecutive food from its peers. Understandably, this occurrence will most likely lead to a lower chance of survival for the other agents.

This experiment takes as inspiration the notion of intraspecific competition. While the more well-known form of competition is between two different species, intraspecific competition delves into competing individuals from the same species.

The phenomenon involves a shared resource, which the individuals deplete for their own benefit. However, this selfish behaviour often is detrimental to all individuals.

Wise and Wagner conducted a study on the competition between juvenile wolf spiders [29]. They observed that the increased density of the population led to a reduction of the available resources in the environment. In turn this lowered the growth of the individual spiders. It must be made clear that there was no direct

competition for the food. Spiders did not attack one another, but rather through indirect competition, the individuals experienced reduced fitness.

The selfishness experiment therefore will be used to demonstrate this type of competition, as individuals will prefer to collect resources for themselves.

5.1.2 Methodology

Furthermore, to ensure that the swarm can fully explore the map, chosen is to allow each agent to break away and wander the environment on its own.

This was accomplished by setting a high level of separation.

A second experiment concerning selfish behaviour is considered as well. This experiment takes the premise of the previous one, but introduces a degree of selfishness. This degree determines the percentage of agents which will act selfishly. The degree will range from 0 to 1 with increments of 0.1.

For example, at a degree of 0.1 and a swarm size of 20 agents, two agents will act selfishly. Therefore we are left with 18 agents who will share their food by means of the FSM, in Figure 3.1.

Those two agents, in this example case, will wander in the environment in search of resources. However, instead of going into delivering, these agents will consume the mushroom immediately and continue their search.

5.1.3 Results

The results of this experiment are divided in two sub-experiments. First, we will discuss the results of populations that are either entirely selfish or cooperative. Next, we will show the results when a degree of selfishness is implemented.

Experiment A: Selfishness

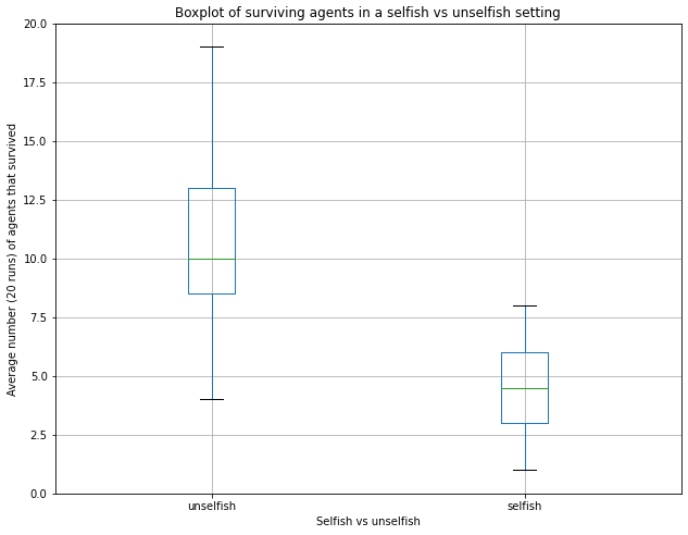

Figure 5.1: The average number of surviving agents for the two strategies

In Figure 5.1 the number of surviving agents are shown for the selfish versus unselfish strategy. A very clear result is obtained, in which the unselfish strategies far outperforms the selfish strategy. With a mean survival rate of 10.60 the unselfish group outperformed the selfish population which achieved a mean of 4.45 only. The best performance by the cooperative group was just shy of perfection, with 19 out of 20 agents having survived. Meanwhile the best performance of the other group was when merely eight agents were left at the end of the simulation. Additionally, the worst performances of the population were 20% and 5%, for the unselfish and selfish strategies respectively.

However, the results obtained vary a lot in the unselfish group, which can directly be attributed to the limited availability of food. We can observe this by means of the standard deviation which is approximately twice as much than that of the selfish population The simulation also requires that sharing of the food only occurs if agents are close to the base. In the case that every agent is actively searching for food to share, there will not always be a moment that there are many agents back at the base, as the mechanism is activated when a food item is collected.

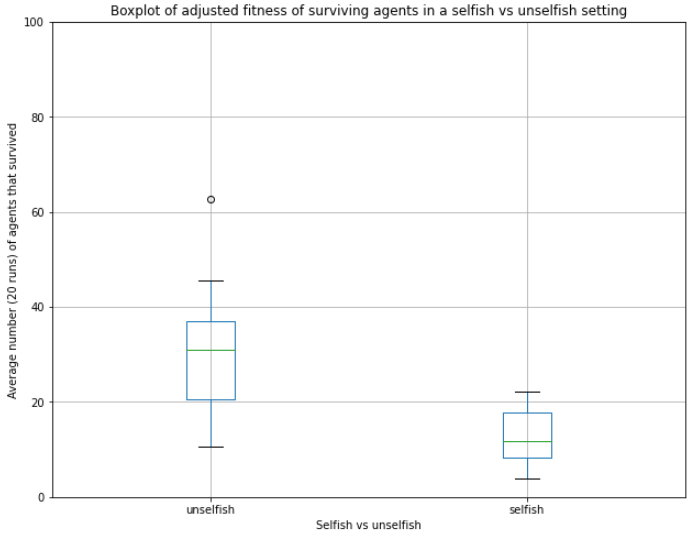

Figure 5.2: The average fitness of the swarm for the two strategies

The fitness, in Figure 5.2 shows that the strategy in which no agent is selfish, leads to high fitness levels. We can observe that the unselfish population achieve a mean fitness of around 30 over the simulation runs. Meanwhile, the selfish swarm managed to achieved an average fitness of just over 12 in this experiment. Another observations is the standard deviation. In the unselfish group this deviation reaches a value of 12.40, while the selfish group has a value of 5.57 for the standard deviation. Furthermore the lowest fitness achieved are 10.70 and 3.83 for respectively the unselfish and selfish strategies, while the values 62.64 and 22.07 represent their maximum achieved fitness.

Experiment B: Degree of selfishness

Moving on from the previous experiment where either the agents are cooperative or selfish, we take a look at the degree of selfishness. First, in Figure 5.3, we can observe the boxplots for the degrees of selfishness.

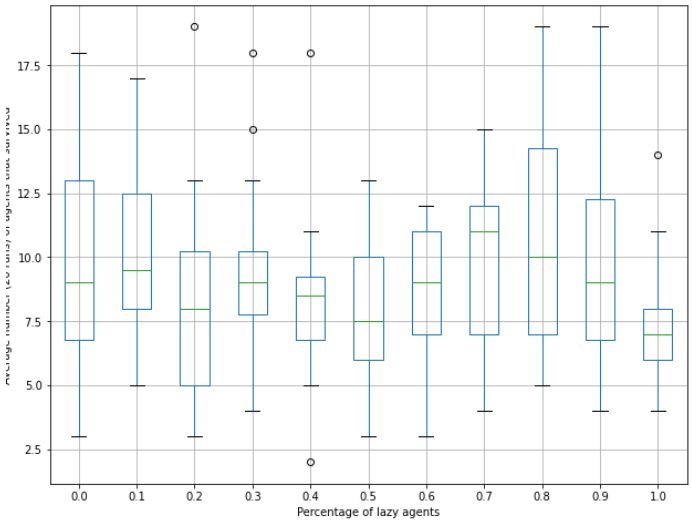

Figure 5.3: The average number of surviving agents for the degrees of selfishness

We can observe that the lowest survival was achieved in the group of fourty percent selfish agents, with 2 surviving swarm members. In the ten and eighty percent selfish population the highest minimal survival was achieved. These groups ended with five, which means that 25% survived in the worst simulation run.

The lowest mean survival was found in the fully selfish group, with a mean of around 7.25 agents, while the highest was achieved by the group in which 80% were selfish.

With 12 surviving agents, the sixty percent population achieved the worst maximum survival rate with a total of 12 agents. The best performance (95%) were achieved by the thirty, eighty and ninety percent selfish swarms.

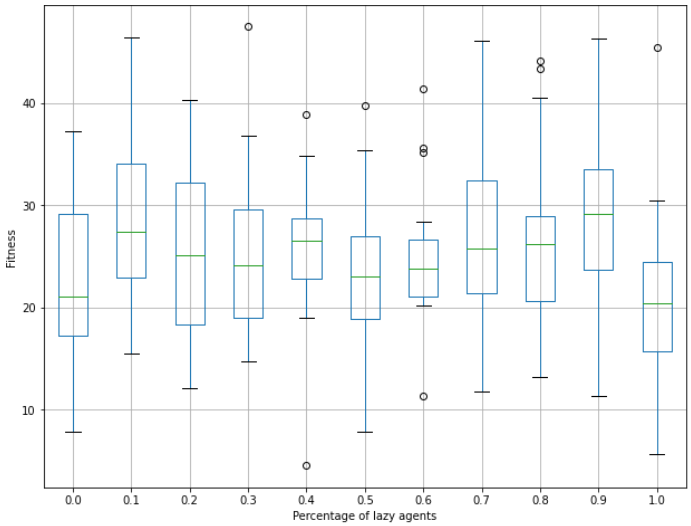

Figure 5.4: The fitness for the degrees of selfishness

Finally, the fitness achieved by each of the selfish strategies is plotted in Figure 5.4. The lowest mean fitness was achieved by the fully selfish population, with value of around 20. On the other hand, in the swarm with 10% selfishness, the highest average fitness (29) was achieved. Between twenty and sixty percent selfishness, the mean fitness remained similar with each group achieving an average fitness of around 25. Higher average fitness was achieved in the more selfish populations up until the already mentioned fully selfish population.

Out of all the groups, four managed to achieve fitnesses below 10. These are the extreme cases, either fully cooperative or fully selfish, and the populations in which fourty and fifty percent of the individuals were selfish. With a value of 47, the highest fitness was achieved in the thirty percent selfish population. Alternatively, the fully cooperative group achieved the lowest maximum fitness with 37.

The standard deviations are largely comparable between the groups. As the lowest found value was 6.49, while the highest was 9.11 we found merely a 40% maximum difference between groups.

We will first discuss the statistical analysis of the experiment concerning two groups. Next, we delve into the experiment in which a degree of selfishness was implemented.

Experiment A: Selfishness

Both populations seem to follow a normal distribution. However we find that the selfish group has a variance of 3.0475, while the cooperative group has a variance of 13.84.

Therefore, we cannot continue with the student's t-test, and need to use the Welch's t-test. We find a p-value of 5.41e-7 which is smaller than our $\alpha = 0.05$. Therefore we reject the null hypothesis.

Experiment B: Degree of selfishness

Barring a few extreme values, all strategies seem to follow a normal distribution.

Moving on to the ANOVA test we find the following. According to the p-value approach, we fail to reject the null hypothesis as the found p-value is larger than our α.

Using the critical value approach we end up failing to reject the null hypothesis. This follows from observing that the F-score is lower than the critical F between the groups.

Therefore we can say that our results are not significant. We do not have sufficient evidence to conclude that the degree of selfishness has an effect on the performance.

5.1.4 Discussion

The results obtained clearly show that sharing should be preferred over being selfish in order for the swarm to survive. However, the obtained boxplots do indicate variance in the results, which rings especially true in the unselfish strategy.

To understand this high variance we need to backtrack to the behaviours involved in the unselfish strategy. As all agents are out to look for food, the expectation is that most of the map is being explored. Therefore, each individual agent is exploring on its own, often far away from the base.

When a resource is collected, and the agent intends to share it with others at the base, the chance that his peers are near the base is slim.

In the first experiment we reject the null hypothesis, therefore we can state that the results of this experiment are statistically significant. The mean of the two strategies are not equal. Therefore in this experiment we have as answer for RQ1 that selfishness has a negative effect on the performance.

In the second experiment we fail to reject the null hypothesis. This indicates that our results are not statistically significant. We cannot claim that the change in strategies differ enough from one another. Due to this we cannot claim that degrees of selfishness have an effect on the performance.

We believe that these two findings can be directly attributed to the following two possible flaws. First, it is highly likely that our sample size was too small to detect the effect of selfishness in the two experiments. Furthermore, we may have encountered noise in our data. We must keep in mind that the stochastic nature of resource spawning leads to high variability. By chance some samples could have been at a disadvantage from the get go.

We recommend the following improvements in the experimental setups for future work. First we should increase the sample size, in the hope that the findings will be statistically significant in that case. A possibility would be to observe 50 samples per setting.

While we do stand behind the mechanism of random spawning of food, it would probably be advised to add a minimal amount. This preset amount needs to be present during the simulation run, to allow a fair comparison between the different experimental runs. Another idea is to spawn a number of resources at the beginning of the simulation run.

5.2 Experiment Lazy

In this chapter we present the second experiment. We will look into the effect of having lazy individuals in the swarm will have in the performance.

Therefore our research question is as follows in this experiment.

- *RQ1. What is the effect of having lazy agents on the performance in the food foraging task?*

5.2.1 Motivation

This experiment delves into the effects of having lazy agents within a swarm. A lazy agent is described as an agent that prefers to stay behind, while others look for food.

The default behaviour of agents in the simulation is to perform exploration with the aim to find resources. However, when being lazy, an agent will not perform this exploration, preferring to exploit the homebase location.

Two motivations are given for this experimental setup which are:

- this setup resembles the dynamics between animals and their young;
- this setup ensures the presence of agents at the base, at high degrees of laziness.

The first motivation is adapted from observing animals in nature. Until a certain point in life, many species rely heavily on the care of their parents. For example in bird species, the young stay behind at the nest, while the parents look for food. Once food has been found it is brought back to the nest to feed the offspring.

The second motivation is based on the previous experiment. In that experiment, it often occured that when a cooperative agent brought back food to the base, no other agents were present to receive a part of the resource. This ultimately rendered the act of sharing useless for the collective swarm.

As mentioned in Chapter 2, interference was identified in earlier work to have a negative effect [24, 25]. This may have occurred due to a large population size or due to external influences. We can identify for the latter either an inappropriate environment size or the resources being scarce.

Balch [30] studied whether moving from homogeneous to heterogeneous tasks would reduce this observed obstruction between individuals. In the homogeneous case, all agents were tasked with foraging all available resources. In the latter case, a distinction is made between red and blue attractors. The author expected that the latter case would outperform the former, as the two different resources would prevent the agents from obstructing each other. The author's expectation was met, as the heterogeneous case led to less interference, however, found was that agents performing the homogeneous task performed better.

As we have a homogeneous task in our experiments we should investigate the effect of interference among the individuals in carrying out this task.

5.2.2 Results

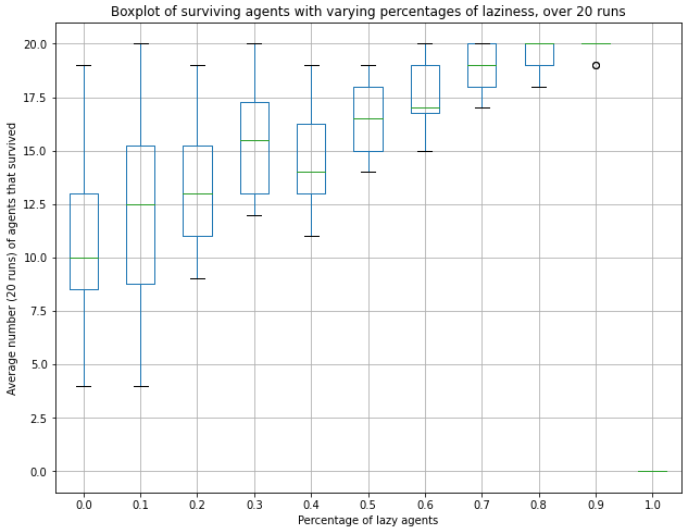

Figure 5.5: Boxplot of the survival versus the degree of laziness

Figure 5.5 shows the average number of surviving agents for each strategy over the twenty runs. Observed can be that a population composed of solely lazy agents, those that do not forage at all, lead to the worst survival rate. In each of the twenty runs, no survivors were left. This is logical, as the only way to survive is by foraging food.

However, by this logic, one could assume that a population, consisting of foragers only, would fare the best. This, as can be observed, has proven to be an incorrect assumption.

Exclusing this group, the worst survival rates were achieved when all agents were active and when only ten percent were lazy. These two strategies achieved a survival rate of only four.

The very best strategy, according to this metric, would be 0.9, in which 18 members stay behind, while two look for food. In all but one simulation, did every member survive.

Furtermore, the majority of groups were able to achieve a perfect survival rate in at least one simulation run. The only groups not able to do so were twenty, fourty, fifty percent lazy and the fully active population.

The resulting boxplot shows that the spread is large. This can be addressed to the stochastic spawning of the mushrooms. Some runs may have seen enough mushrooms to allow many agents to survive, while other runs did not.

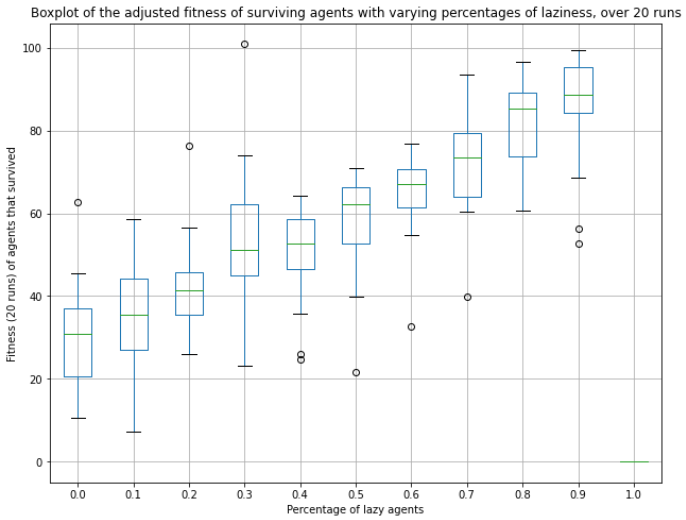

Figure 5.6: Boxplot of the fitness versus the degree of laziness

Regarding the fitness (Figure 5.6), we can observe the following when excluding the fully lazy population. The worst mean fitness was found when all agents were active, with an average fitness of 30.9 only. Meanwhile, the lowest maximum fitness (58.5) was found in the population with ten percent of its members being lazy.

However, when looking into the median results, we find the following results. See Table 5.1 for the median survival. In bold the best strategy is highlighted. We observe that two strategies, namely 80% and 90% laziness, achieved a perfect survival. However, when taking into account the fitness, we can observe that the strategy in which more agents were lazy performed slightly better. Therefore, a population with 90 percent of its population acting lazy outperformed one with 80 percent laziness.

Table 5.1: The results of the median survival for each of the laziness percentages

Laziness	**Survived**	**Fitness**
0.0	10.0	30.914097
0.1	12.5	35.615764
0.2	13.0	41.306875
0.3	15.5	51.178472
0.4	14.0	52.636780
0.5	16.5	62.212452
0.6	17.0	67.072725
0.7	19.0	73.615654
0.8	**20.0**	85.223746
0.9	**20.0**	**88.765490**
1.0	0.0	0.000000

The resulting normality plots do not give us confidence that all our data indeed follows a normal distribution. Instead of ANOVA, we will therefore conduct a Kruskal-Wallis test, as this test does not require normality of the data. Additionally, we opt to bin our groups based on ranges.

- Low: 0.0 - 0.4
- Medium: 0.5
- High: 0.6 - 0.9

We find a p-value of 5.32e-24, therefore we can reject H_0.

As we have rejected the null hypothesis, we move on to the post-hoc test.

From the Dunn table we find that the p-values indicate that the three groups differ significantly from each other.

5.2.3 Discussion

As observed in the plots, having a population which is entirely lazy, has proven disastrous. In hindsight, this result was to be expected. As no agent leaves the radius of the homebase, and in addition does not go into the foraging behaviour, no resources can be collected.

Due to this non-performance, we have decided to not compare the two extreme cases separately, such as in the previous experiment. Ultimately, we can argue that the proposed experimental setup is flawed. As we had set lazy agents, as those that will not seek any resources in the environment, regardless of the proximity, we basically created a situation in which such agents were completely helpless on their own. A better approach would have been to limit the movement of the lazy agents to a small radius of the base. As food items may spawn on tiles near the base tiles, it would be only fair that the lazy agents should be able to collect them. In that case, we need to adjust our definition of laziness. Laziness should have been the unwillingness to exert too much energy in the pursuit of resources. With this new definition, the agents would be encouraged to seek out food, at very little effort.

Moving on tho the degree of laziness setup, we found the following. Excluding the case in which all agents are lazy, we did observe a positive correlation for laziness. As the swarm became more lazy, the achieved fitness became better.

To understand why this has occurred we need to remember how the act of sharing is conducted. Essentially, sharing is only effective if there are recipients ready at the base. Therefore, having more agents stay back, ensures that any resource collected will be shared with a large part of the swarm.

Finally, the results of the median fitness singled out one strategy as being the best. The population in which 90% percent of the agents were lazy achieved the highest fitness. In our setup, this indicates that of the 20 agents, only two were tasked with exploring the map in search for resources. Having only these two agents out, the 18 others were able to get a part of any resource that the two had found. The worst case scenario, in which the two would have failed to have found any food, did not happen in any of the twenty runs.

We can conclude that the results obtained indicate that the different strategies do differ significantly from one another. Due to the low p-values obtained we have received strong evidence to reject the claim that the strategies achieve the same performances.

5.3 Experiment laziness vs selfishness

In this chapter we present the third experiment. We will look into the interplay of selfish and lazy agents in a swarm and their effect on the performance. Furthermore, we will investigate whether changing the population density will have a significant effect as well. We will alter this density in two ways, namely by changing the number of agents in a swarm, or by changing the size of the world map.

Therefore we identify the following research questions.

- *RQ1. What is the effect of having both selfish and lazy agents in the foraging task?*
- *RQ2. What is the effect of changing the population size?*
- *RQ3. What is the effect of changing the map size?*

5.3.1 Motivation

Taking into account the previous two experiments, we are interested to know how the two behaviours together may impact survival.

The experimental setup considers four cases, which are:

- Experiment A: Regular map of 20x20 tiles and 20 agents
- Experiment B: Regular map of 20x20 tiles and 50 agents
- Experiment C: Large map of 30x30 tiles and 20 agents
- Experiment D: Large map of 30x30 tiles and 50 agents

Based on these four cases, we have the following addtional hypotheses:

- A larger map will lead to lower survival
- A larger swarm will lead to higher survival

The first hypothesis, we believe that having a larger map will make foraging less fruitful than in a smaller map. Therefore, the swarm will perhaps collect less food, which in turn negatively affects survival.

A larger swarm size is expected to achieve higher fitness. The reasoning behind this is that a larger number of agents will be out looking for food. Having more numbers exploring the environment, should lead to more collected resources.

Given the results in the previous experiments, the expectation is that a population in which 90% of the agents are lazy would fare best. Additionally, the lower the selfishness, the higher the achievement is expected to be.

This experiment aims to research the interplay of selfishness and laziness. It is directly influenced by the previous two experiments. As with the laziness experiment we may deal with a nest of breed. Alternatively, the population is encouraged to prevent interference while foraging.

We see the carrying capacity as direct interpretation for this experiment. It is the theory which concerns itself with the maximum population size of a species that is sustainable in a specific environment. This is based on resources such as water, food and habitat [31].

Given this definition, an increase in population size, with limited resources, reduces the available resources for each individual.

$$\frac{dN}{dt} = rN(1 - \frac{N}{K})$$

where N is the population size, r is the intrinsic growth rate, K is the carrying capacity of the environment. We hope to find by means of the proposed variations in the experiments how the carrying capacity of the environment changes, while the population size changes and the environment size is enlarged.

5.3.2 Methodology

The methodology used in this experiment is a combination of the previous two experiments. To mimic laziness, the agent remains near the base. Accordingly, selfishness is performed by consuming the collected resource directly for itself.

However, as the selfishness experiment used a boolean expression to determine whether a swarm is selfish or not, this experiment uses a degree of selfishness.

The degree forms a ratio with the degree of sharing, meaning that the two degrees need to total to 1.

Consider the following cases:

- Case A': degree of selfishness 0; degree of sharing 1;
- Case B': degree of selfishness 0.5; degree of sharing 0.5;
- Case C': degree of selfishness 0.2; degree of sharing 0.8;
- Case D': degree of selfishness 1; degree of sharing 0;

Simply put, the cases A' and D', will mimic the selfishness experiment with case A' being the unselfish case and case D' the selfish case. Case B' introduces a swarm in which half of the population will be selfish, while the other half will cooperate. Finally, case C' shows a swarm with a minority of selfish agents.

Now laziness also becomes a factor. Considering the four cases mentioned above, we need to consider the laziness of the population. The laziness ranges from 0 to 1, meaning that either no one is lazy or everyone is lazy, respectively.

We then encounter the following possibilities for the cases above:

- Case A": degree of selfishness 0; degree of sharing 1; degree of laziness 1;
- Case B": degree of selfishness 0.5; degree of sharing 0.5; degree of laziness 0.5;
- Case C": degree of selfishness 0.2; degree of sharing 0.8; degree of laziness 0.8;
- Case D": degree of selfishness 1; degree of sharing 0; degree of laziness 0;

Case A" now shows a population willing to share, but too lazy to search for food. On the other hand, case B" depicts a selfish population in which all agents actively look for resources. Cases B" and C" have swarms with populations consisting of mixed degrees of laziness. While some agents are active, others are lazy.

Given the current experimental setup, in which two parameters are used to retrieve three variables, the boxplot no longer is considered an appropriate way to visualize the results. Chosen is to use a ternary plot, where these variables must sum to the constant value of 1. On the bottom edge we will plot laziness, while the sides of the triangle will take in the degrees of selfishness. The left edge will depict the degree of sharing in ascending order, while the right edge will depict the degree of selfishness in descending order.

For the statistical tests in this chapter we opt to create sub-categories for the parameter settings. In the previous two experiments, we had 11 different groups to compare the degrees. However, when combining the two degrees, we end up with a

large number of groups. Chosen is to combine them in the following groups, see Table 5.2.

Table 5.2: The ranges for which the two variables are binned.

Laziness range	Selfishness range	Group
0.1 - 0.3	0.1 - 0.3	Low-Low
0.1 - 0.3	0.4 - 0.6	Low-Medium
0.1 - 0.3	0.7 - 0.9	Low-High
0.4 - 0.6	0.1 - 0.3	Medium-Low
0.4 - 0.6	0.4 - 0.6	Medium-Medium
0.4 - 0.6	0.7 - 0.9	Medium-High
0.7 - 0.9	0.1 - 0.3	High-Low
0.7 - 0.9	0.4 - 0.6	High-Medium
0.7 - 0.9	0.7 - 0.9	High-Low

5.3.3 Results

In this section we will present the obtained results. Each variation will be treated in its own separate section.

Experiment A: regular map and 20 agents

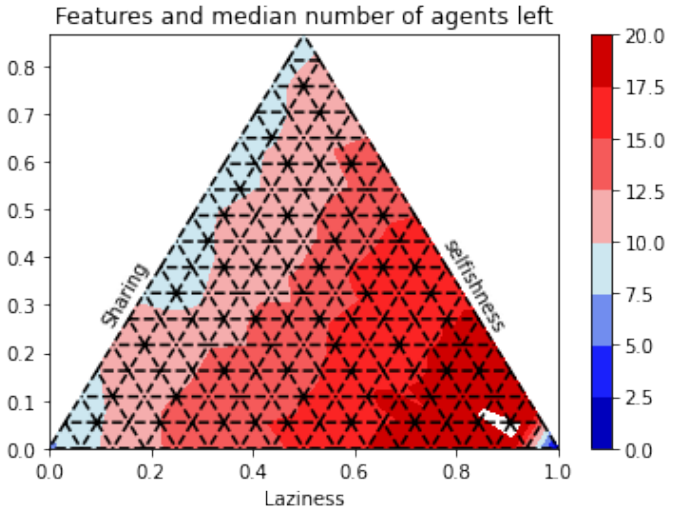

Figure 5.7: The survival metric of a swarm of size 20 in a regular environment.

In Figure 5.7, we can observe that a low number of agents have survived in the case that all agents were lazy, see the bottom right corner. In the bottom left corner, we observe the poor performance when none of the agents are lazy and are all selfish.

The top corner, the case in which all agents are active and cooperative also performed poorly.

Diving deeper into our obtained results we find that the greatest median survival was achieved when 90% of the swarm was lazy. Regardless of the selfishness all cases achieved perfect survival. This settings is closely followed by the eighty percent lazy population. While perfect survival was achieved, it came short in a number of selfishness settings. The extreme selfishness cases both achieved perfect survival, as did the 50 and 60 percent selfish swarms. The other cases achieved 90 and 95% median survival rates.

At low levels of laziness, we observed poor survival rates. Below 20% laziness, regardless of the selfishness, did a swarm achieve a median fitness greater than 60%.

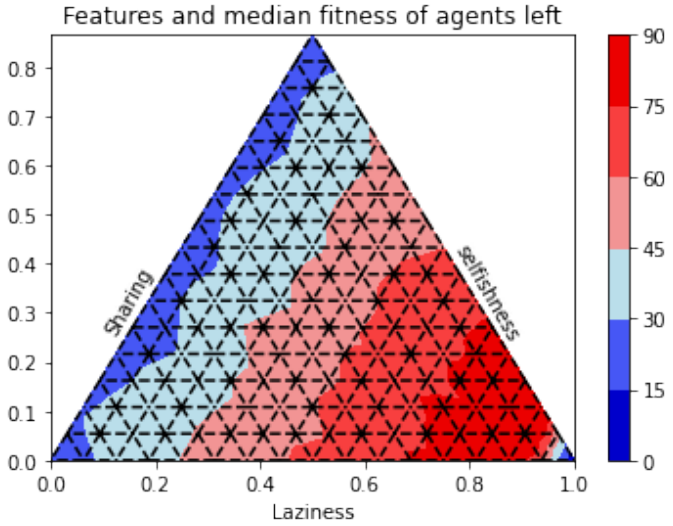

Figure 5.8: The fitness metric of a swarm of size 20 in a regular environment.

The plot of the fitness in Figure 5.8 shows a similar trend as observed in the survival plot. The greatest fitness (89%) was achieved when 90% of the swarm was lazy and 70% allowed themselves to be selfish. Below 70% laziness, regardless of selfishness, no population was able to achieve a median fitness greater than 75%. Furthermore, a trend can be observed in which an increase in the laziness, led to a jumps of 10 for the median fitness. This trend continued to show up until the aforementioned seventy percent, after which the jumps became noticeably smaller.

Based on the observations in Table 5.3 we can observe that in six cases the selfishness influences the survival. In the other cases, the exact same was achieved. Both strategies outperform the other in half of these occurrences. The largest discrepancy is found in the case where no member of the swarm acts selfishly. Here the median results show that two more agents survived when agents acted selfishly, as opposed to sharing. Due to the large number of groups we opt to show the results of the combined extreme approaches. Either being all or none selfish and/or lazy. The results are shown in Table 5.3. Highlighted is the approach which yielded the best performance.

Table 5.3: The median results of the combinations of the two vectors for the extreme values.

Laziness	Selfishness	Survival	Fitness
0.0	1.0	7.0	20.425972
0.0	0.0	9.0	21.018125
0.1	1.0	10.0	31.379514
0.1	0.0	11.0	31.058125
0.2	1.0	12.0	40.240972
0.2	0.0	12.0	44.273819
0.3	1.0	14.0	50.395347
0.3	0.0	13.0	48.653264
0.4	1.0	14.0	55.074975
0.4	0.0	14.0	54.434653
0.5	1.0	15.5	64.196896
0.5	0.0	16.5	59.636688
0.6	1.0	17.0	70.757440
0.6	0.0	16.5	67.442382
0.7	1.0	19.0	76.353889
0.7	0.0	18.0	78.378991
0.8	1.0	20.0	80.940572
0.8	**0.0**	**20.0**	**84.640278**
0.9	1.0	20.0	76.888264
0.9	0.0	20.0	78.919792
1.0	0.0	0.0	0.000000
1.0	1.0	0.0	0.000000

Other than the best performance we can observe the following. The selfishness does not influence the result in a consistent manner. We can identify both cases that being selfish led to higher and lower survival as opposed to being cooperative. Furthermore, we can also observe an increased survival when the population became increasingly lazy.

Experiment B: regular map and 50 agents

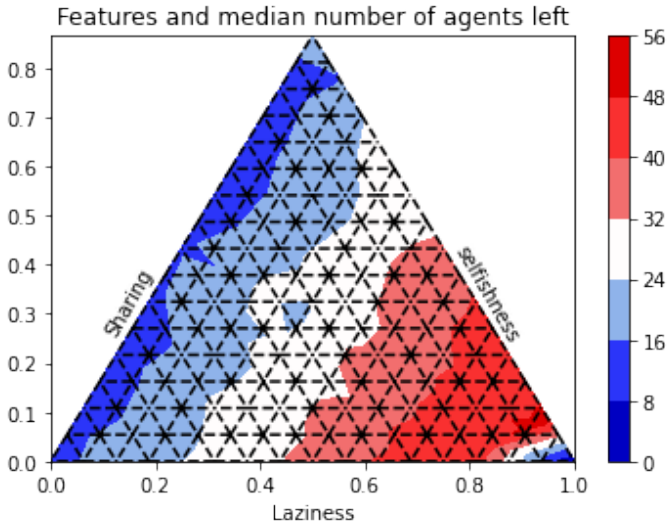

Figure 5.9: The survival metric of a swarm of size 50 in a regular environment.

We can observe in Figure 5.9 that the three extreme cases led to the following results. When all agents act lazy, the survival is zero in each and any case, regardless of the selfishness. In the case that all agents are active, and are all selfish, we observe a survival of around fourty percent. The rest of the selfish strategies in this case ended up lower than 30%, with a fully selfish population achieving the worst with a survival rate of merely 18 percent.

Furthermore, we can identify a positive trend for laziness, excluding the extreme case of being fully lazy. The higher this degree, the higher the survival becomes. While the jumps in the previous experiment were roughly of size ten, here we observe more conservative jumps of five. Starting from 70% laziness, regardless of the selfishness, the populations achieve survival rates of 80% and greater. It must be noted that no strategy obtained perfect survival, as opposed to the experiment with a regular sized swarm. The highest survival rate (95%) was achieved in the 90% lazy population, regardless of the selfishness.

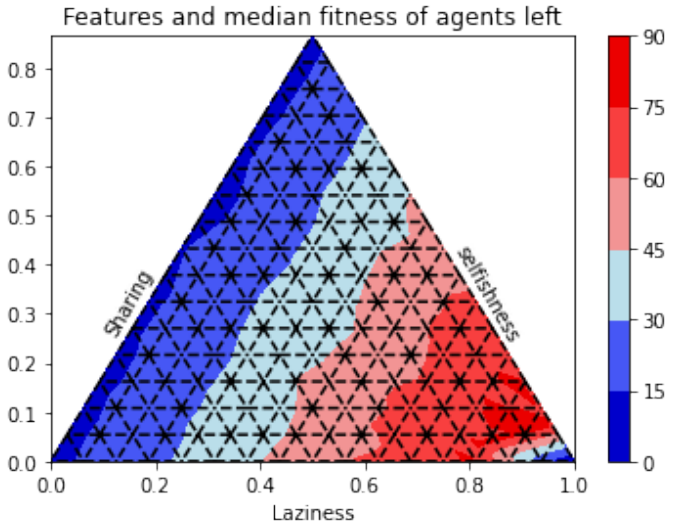

Figure 5.10: The fitness metric of a swarm of size 50 in a regular environment.

Next, in Figure 5.10 the fitness is plotted. The plots reinforces what we have observed in the survival plot, namely that the extreme cases performed poorly. Furthermore, here also we detect a positive trend for the laziness variable. The greatest median fitness (83%) occured in the 90% laziness swarm, when only ten percent allowed themselves to act selfishly.

The results are shown in Table 5.4. Highlighted is the approach which yielded the best performance.

Table 5.4: The median results of the combinations of the two vectors for the extreme values for a large swarm size in a regular sized environment.

Laziness	**Selfishness**	**Surival**	**Fitness**
0.0	1.0	9.5	9.191889
0.0	0.0	19.5	11.690139
0.1	1.0	21.0	21.485389
0.1	0.0	15.5	20.051278
0.2	1.0	21.0	28.001907
0.2	0.0	24.0	29.536545
0.3	1.0	26.5	34.216183
0.3	0.0	27.5	36.829520
0.4	1.0	29.5	44.301317
0.4	0.0	28.0	47.948275
0.5	1.0	35.0	54.759247
0.5	0.0	33.5	52.055443
0.6	1.0	39.5	61.531384
0.6	0.0	39.5	59.891482
0.7	1.0	43.0	70.789129
0.7	0.0	43.5	69.345438
0.8	1.0	46.0	77.731177
0.8	0.0	46.0	71.743555
0.9	**0.0**	**49.0**	**72.093290**
1.0	0.0	0.0	0.000000
1.0	1.0	0.0	0.000000

Here again we observe that the laziness has an influence on the performance, while selfishness does not affect the performance consistently.

Experiment C: large map and 20 agents

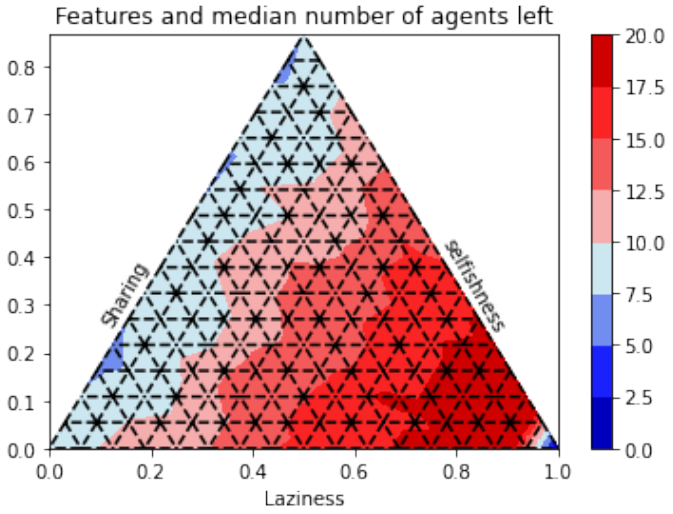

Figure 5.11: The survival metric of a swarm of size 20 in a large environment.

The survival in the third case is plotted in Figure 5.11. We can observe, in the bottom right corner, that when all agents are lazy, they perform poorly. On the other hand, regardless of being selfish, the active swarm achieves below-average survival.

As with the previous experiments, did the population in which ninety percent of agents were lazy outperform the rest. All but the case when 60% of the agents allowed themselves to act selfishly, the swarm achieved perfect survival.

Below 20 percent laziness, only three times did a population achieve a median survival greater or equal to 50%. Only when above 70% laziness, were populations able to achieve 75% survival.

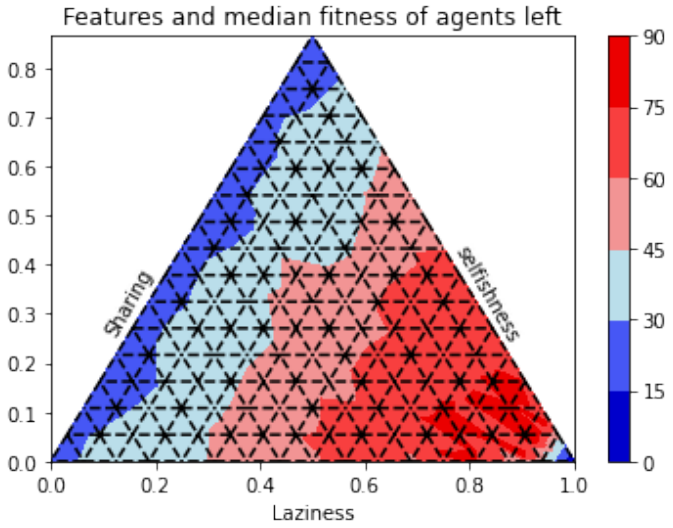

Figure 5.12: The fitness metric of a swarm of size 20 in a large environment.

The fitness in Figure 5.12 shows that all three extreme cases lead to poor survival. Also selfishness seems not to affect performance, whereas the laziness does.

While the greatest fitness (84%) occured at 80% laziness and 20% selfishness, it was coupled with 95% survival. Taking both metrics into account, we find that the greatest performance was achieved by the 90% laziness and 50% selfishness. Here all agents survived with a median fitness of 80%.

The results are shown in Table 5.5. Highlighted is the approach which yielded the best performance.

Table 5.5: The median results of the combinations of the two vectors for the extreme values for a regular swarm size in a large environment.

Laziness	**Selfishness**	**Survival**	**Fitness**
0.0	1.0	7.5	24.875764
0.0	0.0	7.5	25.088542
0.1	1.0	10.0	34.510000
0.1	0.0	9.0	29.443056
0.2	1.0	11.5	40.206042
0.2	0.0	11.0	37.364653
0.3	1.0	12.0	44.855903
0.3	0.0	12.0	48.877639
0.4	1.0	14.5	57.649350
0.4	0.0	14.0	54.340669
0.5	1.0	15.5	57.935391
0.5	0.0	14.0	59.665322
0.6	1.0	16.0	67.896532
0.6	0.0	16.0	67.514823
0.7	1.0	18.0	56.425694
0.7	0.0	18.0	74.015069
0.8	1.0	19.0	77.676160
0.8	0.0	19.0	72.680359
0.9	1.0	20.0	49.426988
0.9	**0.0**	**20.0**	**65.447643**
1.0	0.0	0.0	0.000000
1.0	1.0	0.0	0.000000

The findings of the extreme values in this experiment further show that the laziness drives performance up, and that selfishness fails to have a clear influence.

Experiment D: large map and 50 agents

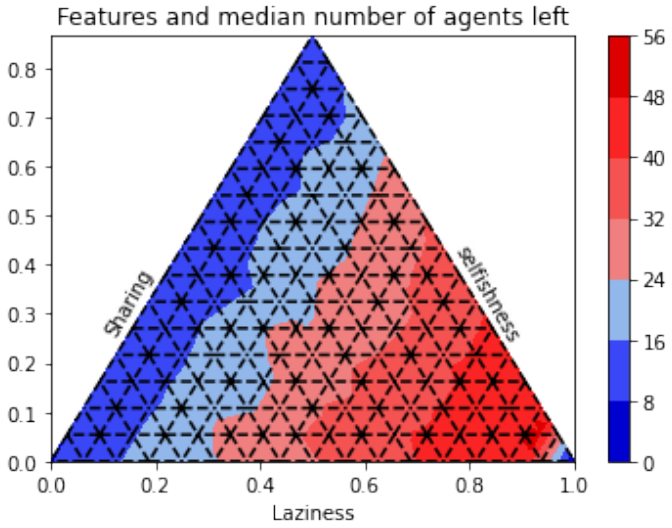

Figure 5.13: The survival metric of a swarm of size 50 in a large environment.

In the final experiment, we can observe that the extreme cases achieve poor peformances, see Figure 5.13. Selfishness did again not affect the survival, whereas the laziness did. A similar trend can be observed as with the previous experimental settings. The increased laziness led to increased survival rates.

Perfect survival was not achieved by any of the groups. In this experiment the highest survival rate was 95%.

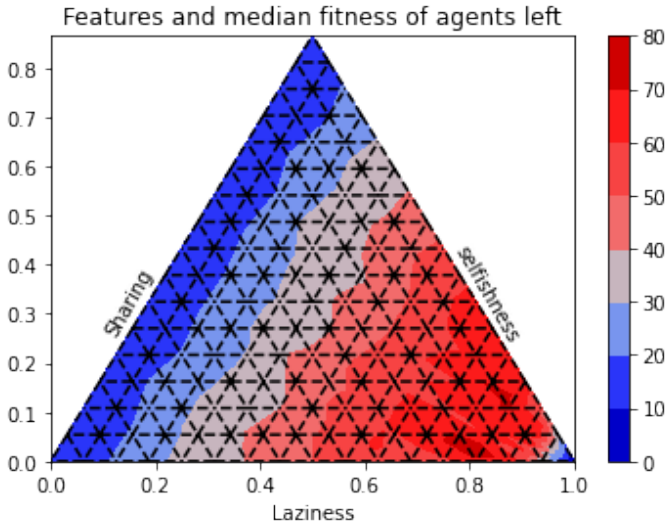

Figure 5.14: The fitness metric of a swarm of size 50 in the large environment.

Lastly, the fitness reinforces the observation from the survival plot. In Figure 5.14 we again observe that laziness is important, extreme cases perform poorly, and selfishness does not seem to affect fitness.

As opposed to Experiment B, the median fitnesses are lower. The highest value found was 77, when the population was 80% lazy and 90% selfish. Barring one other observations did no other population achieve a median fitness greater than 70%.

The results are shown in Table 5.6. Highlighted is the approach which yielded the best performance.

Table 5.6: The median results of the combinations of the two vectors for the extreme values for a large swarm size in a large sized environment.

Laziness	Selfishness	Survival	Fitness
0.0	1.0	7.5	9.544583
0.0	0.0	10.0	10.824528
0.1	1.0	14.5	18.954417
0.1	0.0	15.0	18.796127
0.2	1.0	18.5	28.546507
0.2	0.0	19.0	27.112490
0.3	1.0	23.0	34.472199
0.3	0.0	25.0	33.621368
0.4	1.0	27.0	42.196536
0.4	0.0	31.0	43.349815
0.5	1.0	31.0	49.820326
0.5	0.0	35.0	51.695465
0.6	1.0	37.5	57.982287
0.6	0.0	37.0	60.661591
0.7	1.0	40.0	54.583846
0.7	0.0	42.0	69.252961
0.8	1.0	46.0	61.436917
0.8	0.0	45.0	66.961935
0.9	**1.0**	**48.0**	**57.530028**
0.9	0.0	48.0	45.001278
1.0	0.0	0.0	0.000000
1.0	1.0	0.0	0.000000

In this last experiment, we observe the same as in the previous experiments.

In this section we will present the statistical analysis. Each variation will be treated in its own separate section. At the end we will perform three additional tests, in order to answer our hypothesis regarding the effect of changing either the swarm or environment sizes. These two tests will be preceded by a examination of the four experiments, in order to ensure that they differ significantly from each other.

Experiment A: regular map and 20 agents

All but the groups high-low, high-medium and high-high show a normal distribution. We prefer to err on the side of caution and move on to a non-parametric test.

We perform the Kruskal-Wallis test and find a p-value of 8.21e-215, which is much smaller than our α. Therefore, we can reject the null hypotheses that stated that all groups achieved similar results.

Next, we move on to the Dunn test for the post-hoc statistical analysis.

The Dunn test p-values indicate that each group is similar within its laziness range. For example all the groups with a log degree of laziness, regardless of their degree of selfishness received a p-value of 1. Between each laziness group, we do see a significant difference.

From the test values we cannot conclude that each and every group differs significantly from each other. However, we can clearly observe that when laziness levels are similar, the groups are considered similar regardless of the selfishness. Therefore all groups in which laziness was low, were found not to differ from each other. The same was found for medium and high levels of laziness. We do see a clear difference between the laziness levels, as the three sub-groups are found not to be similar at all.

Experiment B: regular map and 50 agents

From the normality test we conclude that our data follow a normal distribution. Nevertheless, we will try to deploy a ANOVA test on the gathered data.

In both methods described, the null hypothesis is rejected. Also, using the critical value approach we end up failing to reject the null hypothesis. This follows from observing that the F-score is lower than the critical F between the groups.

Therefore, we move on to the Tukey test.

Observed was that only those pairwise comparisons in which the laziness levels were similar failed to get rejected. Those pairs that did reject the null hypothesis did have different levels of laziness. Therefore, we end up at a similar conclusions as in the previous experiment, namely that the selfishness does not have an effect. On the other hand, the levels of laziness do show significant differences between each other.

Experiment C: large map and 20 agents

We found that not all our samples are normally distributed. Therefore, we move on to the Kruskal-Wallis test.

We find a p-value of 1.59e-168, which is much smaller than our α. Therefore, we can reject the null hypotheses that stated that all groups achieved similar results.

We move on to the Dunn-test for post-hoc statistical analysis. The resulting Dunn test shows here too that the laziness groups are not affected by selfishness.

Based on this test, we can conclude that the effect is exactly the same as found in Experiment A. Namely, the selfishness did not lead to differences when the laziness was the same. However the degrees of laziness groups did differ significantly from each other.

Experiment D: large map and 50 agents

We found that not all our samples are normally distributed. Therefore, we move on to the Kruskal-Wallis test.

We find a p-value of 5.48e-205, which is much smaller than our α. Therefore, we can reject the null hypotheses that stated that all groups achieved similar results. The Dunn test in this experiment also shows that selfishness did not affect the performance.

Again we end up at the same conclusion as in Experiments A and C. The degrees of laziness differ significantly from each other, while the selfishness seems not to have much of an effect.

Comparing the four groups

In this subsection, our aim is to find answers for RQ2 and RQ3 through the use of statistical analysis. We employ Kruskal-Wallis on the four different groups. Found is a p-value of 2.43e-95, which is much smaller than our alpha. Therefore, we can reject the

null hypothesis which stated that the means of the groups were equal. The Dunn post hoc test is carried out to find out which groups differ significantly from each other.

Found was that the other than experiments C and D, all the groups differ significantly from each other. This would indicate that the change in swarm size did not lead toa significant difference in a large sized map. We will conduct two statistical tests next, to have a complete overview regarding the effect of changing the sizes of either the swarm or the map.

Map size test

The following statistical test will try to answer the first hypothesis we stated with regards to the four experimental designs. We will test the following:

- H_0 : The map size does not has an effect on the fitness
- H_A : There is a effect

We will compare the following couples in a Welch's t-test.

- Experiment A (regular swarm size, regular map) and Experiment C (regular swarm size, large map)
- Experiment B (large swarm size, regular map) and Experiment D (large swarm size, large map)

From the first test we receive a p-value of 1.0539e-05. This value is lower than our alpha, therefore we reject the null hypothesis. The means of the two groups are not equal, and as such the larger map did have an effect in a regular sized swarm.

Next the experiments concerning the large swarm size led to a p-value of 0.029. Again, we are to reject the null hypothesis.

Taking both comparison into consideration we can conclude with strong evidence that changing the size of the map does affect the performance of the swarm.

Swarm size test

In this test we compare equal map sizes for a possible effect of the swarm size. Therefore we will compare the following couples in a Welch's t-test.

- Experiment A (regular swarm size, regular map) and Experiment B (large swarm size, regular map)
- Experiment C (regular swarm size, large map) and Experiment D (large swarm size, large map)

We will test the following hypothesis:

- H_0 : The swarm size does not has an effect on the fitness
- H_A : There is a effect

The first comparison led us to the following p-value of 2.71e-52, which is much smaller than our alpha. The second pair also achieves a very small p-value, namely 1.81e-42. Overall, we can reject the null hypothesis with strong evidence. As such we conclude that the swarm size has an effect, as the means are not equal between the groups.

5.3.4 Discussion

In these experiments, we looked into the interplay between laziness and selfishness. The results seem to indicate that a lazier, but not fully lazy, population is preferred over a more active population. These findings coincide with the conclusions drawn in the laziness experiment in chapter 5.2.

Regarding selfishness, it seems that the influence of this parameter does not matter much. There were few instances in which selfishness did influence the results. However, these discrepancies were few and far between.

These observations are somewhat given the conclusions drawn in the selfishness experiment in chapter 5.1. Given these findings, we argue that RQ1 can be answered as follows. The effect found when both lazy and selfish agents are present in a swarm on the foraging task performance is that the laziness has a greater influence than the selfishness.

From visual inspection it was difficult to determine the influence of the sizes of the map and the swarm. However, after conducting statistical analysis we did find that they indeed had an effect. Therefore, we conclude for RQ2 and RQ3 that a diminishing effect on the performance was found when making either the swarm or the map larger.

We found that both experiments with a large swarm size, failed to reach perfect survival, regardless of the map size in the extreme cases of selfishness. This is in contrast to the two experiments conducted with a regular sized swarm. In those experiments combined, six cases were found in which perfect survival was achieved.

Overall, we can conclude the experiments as follows. In a hybrid setting, the laziness had an influence on the performance in the foraging task, while the selfishness did not show to have a significant effect. Furthermore found was that changing the swarm densities through the size of the swarm or the map had a significant diminishing effect on the overall swarm health.

5.4 Experiment Group

In this chapter we present the final experiment. As with the previous experiment, will look into the effect of having selfish individuals in the swarm will have in the performance. However, the main difference in this experiment, is that all agents are expected to move together as a single entity. Through this we hope to answer the following research question.

- *RQ1. What is the effect of having both selfish and lazy agents in the foraging task, while moving as a single unit?*

5.4.1 Motivation

In this chapter we will look into the experimental setup in which the swarm moves as one unit. The emphasis is put on the word moves, as this experiment also takes into account the laziness. Those agents that are lazy will not join the group in searching for food, and will stay behind at the base.

The wandering swarm explores the environment, with the hope of finding resources.

The sharing is done with the active part of the population only.

The expectation is that the exploration of the active swarm will lead to low numbers of food collected. In previous experiments, the swarm was broken up from the beginning. Each agent moved away in different directions, which enabled the swarm as

a whole to explore the entire map. Now however, the swarm wanders together leading to a low level of exploration.

Given the limited sensing range of the agents, it is well possible that during a simulation run, the swarm misses out on the majority of resources that have spawned.

We can identify two units, one that goes out to forage and the other left behind. We can presume that those left behind were already candidates likely to perish. Such individuals could have either negative traits or be injured. Therefore, the foraging group forms its own swarm, in which it collectively forages food. Here we can encounter natural selection with regards to selfishness.

5.4.2 Methodology

In order for the swarm to forage as a single entity, the separation force was taken out of the equation. By means of alignment and cohesion alone, the agents remain close together.

The resulting flock size depends on the proportion of lazy agents (ϵ) in the initial swarm (N). Only those agents that do cooperate, do participate in the foraging flock. As such we can determine the flock size ($\hat{N}$) as follows.

$$\hat{N} = N - \epsilon * N$$

Given the FSM in Figure 3.1, the delivery behaviour is omitted. After collecting a mushroom the agent senses its direct neighbours, and shares among those agents.

5.4.3 Results

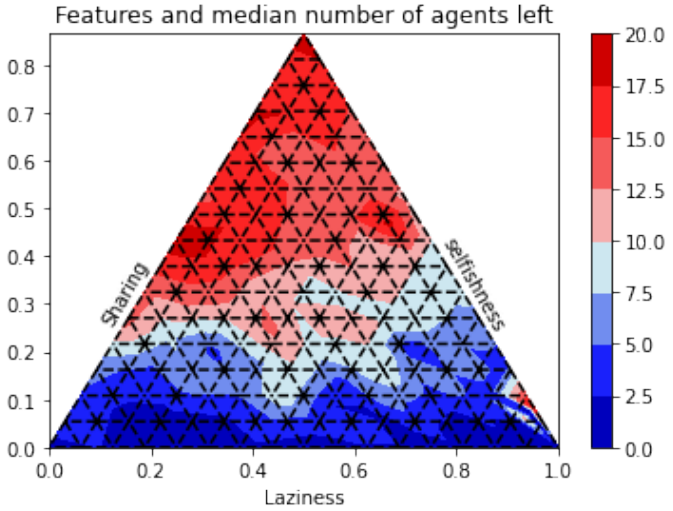

Figure 5.15: The survival metric against the three variables.

The observations mentioned above are clearly seen in the median plot of Figure 5.15. We clearly observe the red region in the upper part of the triangle. This indicates that a strategy that emphasizes on sharing outperforms that of being selfish.

A trend can be observed that when the laziness is low, and less than half of the population allows itself to act selfishnessly, then the majority of agents are able to survive.

However when the more than half of the population is lazy, only two cases were found in which a survival rate greater than fifty percent was achieved. These cases, both found in ninety percent laziness, were with a selfishness of 30% and in a fully cooperative population. These two strategies achieved survival rates of 52.5% and 90%, respectively. Furthermore, in only 10 cases out of 44 total, was a median survival achieved over 37.5%. Leaving around 77% percent of cases to have a survival rate which was lower than that.

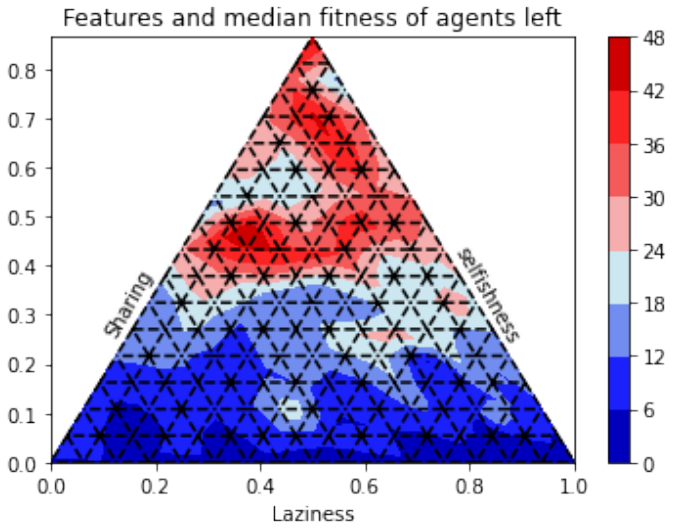

Figure 5.16: The fitness metric against the three variables.

Figure 5.16 shows that, as opposed to survival, the fitness does have variance even in a highly cooperative population.

In the population with ten percent lazy agents and fourty percent selfishness, the highest median fitness (47) was achieved. Overall, in only five out of the 220 total setups, did the median surpass a fitness of 40.

When the laziness surpassed 60 percent the highest median fitness achieved was 26, with the majority of values diving below 10. The very worst achievement was found in a 90% lazy population in which 60% of agents allowed themselves to act selfishnessly. In this scenario, a median fitness of only 1.52 was reached.

Both metrics show that when all agents are lazy, not one agent is able to survive. As such the fitness is zero in each and any of these cases.

Testing normality, we found that not all our samples are normally distributed. Therefore, we move on to the Kruskal-Wallis test.

A p-value of 5.604e-24 was found, which is much smaller than our α. Therefore, we can reject the null hypotheses that stated that all groups achieved similar results.

The Dunn post-hoc test shows that there are numerous groups that differ significantly.

The low laziness group saw significant difference between selfish groups. When the population had high levels of selfishness, it differed enough from those with low to medium levels.

Furthermore, low and medium laziness groups were similar when their levels of selfishness were comparable. This can be seen between the low-low and medium-low groups and low-medium and medium-medium pairwise comparison.

Somewhat surprising we found that the low-high group did not differ significantly from either high-low or high-med groups. These groups neither have similar levels of laziness nor levels of selfishness. The same can be said about the pairs medium-medium versus high-low and high-low and medium-medium.

5.4.4 Discussion

In this experiment we can conlude that sharing is preferred over being selfish. This finding coincides with the conclusions drawn in the selfishness experiment (chapter 5.1). Furthermore, found was that laziness also had an effect on the performance.

As with the previous experiments, the fully lazy populations did not achieve any results. Here also we should have changed the implementation in order to allow these population a chance to survive.

The conducted statistical tests allowed us to reject the null hypothesis that all groups would be similar. Upon further investigation, the post hoc test showed that the majority of group comparison also led us to reject the null hypothesis. However, we did observe that some groups did not differ significantly from one another.

For instance the low laziness groups with low and medium selfishness were deemed to not differ significantly from each other. However both these groups did differ enough from the swarm which was eqaully lazy, but had high levels of selfishness.

Additionally, these two groups also did not differ significantly, from their counterparts in the medium laziness groups. The low selfishness was similar enough to the population in average laziness, while the medium selfishness was similar to both low and medium selfishness groups in the medium laziness group. Furthermore, both low and medium selfishness in a highly lazy swarm did not differ enough from this group.

We see a similar trend between the groups in the high laziness class. Populations that were either composed of low and medium selfish agents did not differ significantly from each other. However from the highly selfish sub-group they did both differ enough.

Overall, given these results we can conclude that low to medium laziness and selfishness are quite similar. Only when we delve into major cases of either behaviour do we find a significant difference.

6 Discussion

In this work, we have presented a simulation engine for the development of swarm intelligence. Self-organizing behaviours in the context of task allocation has been demonstrated.

We have achieved all of our goals in this thesis. Therefore, we can identify the following contributions to research

1. In chapter 2 we motivated work on swarm intelligence in video games by highlighting relevant work in the field of Game AI. In particular we looked into real time strategy games.
2. In chapter 3 we documented the methods to develop a multi-agent survival game engine, in which swarm intelligence can be implemented.
3. The experiments conducted show that task allocation has been used for self-organized behaviour, without the need for a centralized decision mechanism.
4. The selfishness experiment demonstrated that a negative effect was found in extreme cases of selfishness.
5. The laziness experiment showed that being inactive could lead to better performances in the foraging task.

Motivating work on swarm intelligence

By means of the literature review provided in chapter 2, we demonstrated how swarm intelligence could be used in video-game environments. We have highlighted work done on the foraging tasks, both in the scope of video-games and outside of it. While the works discussed concerned themselves with a narrow range of tasks, it is certainly possible to expand upon them. The main requirement is that a number of simple agents need to collaborate in order to achieve a goal.

One could imagine the use of swarm intelligence in sports video-games, such as football or basketball. With a decentralized decision making approach the agents would be able to dynamically allocate tasks depending on the situation they are in. This can be achieved simply by applying the methodology as presented in the work by Reynolds [15]. The group steering behaviours of alignment, cohesion and separation would be sufficient to create a challenging bot. For instance, when one player makes a forward run, the others move in such a way that losing possession will not lead to a goal scoring chance for the opposition. Alternatively, when out of possession, the agents form a pattern in which they are able to pressure the opponent into making a mistake.

The game engine

The choice of building the environment around the world tiles, has led to complications. These include the difficulties in precisely placing elements, and collecting resources. When running a simulation, there exists a certain displacement of mushrooms. Therefore adjustments had to be made to allow agents to collect mushrooms by being within close vicinity, as opposed to having the rectangles fully overlap.

The implemented obstacle avoidance failed a number of times, which led to agents stepping out of bounds, and therefore to be killed off. Had we implemented a deceleration phase when sensing the edge, we could have prevented this undesirable outcome.

Experiments

We can conclude that, overall, the obtained results were mixed. While the experiments with single vectors did result in clear observations, the combined experiments seemed to lessen the impacts. The selfishness experiment (chapter 5.1), for instance, showed a clear decline in performance when the swarm was acting selfishly However, the first ternary experiment in chapter 5.3, failed to show a clear effect of selfishness. The results in that experiment coincided with the laziness experiment (chapter 5.2) only. Observed could be a clear positive trend with increasing laziness, excluding fully lazy, which was exactly what was concluded in chapter 5.2.

However, when agents moved as a single unit (chapter 5.4), we observed the exact opposite. Now the selfishness was the deciding factor, while the laziness was barely affecting the performance.

6.1 Limitations

At this point in development, the food items are spawned by chance. This has led to some simulation runs to spawn insufficient mushrooms to have a chance in achieving high levels of survival of the swarm. In order to alleviate this problem, a predefined range should have been implemented. This range would, on average, need to provide enough mushrooms in the environment over the simulation runs for each parameter setting.

Motivating work on swarm intelligence

While we have motivated how and why swarm intelligence could be implemented in video-games, the industry seems less likely to cooperate in our endeavours. While Deepmind and Blizzard have graceously open-sourced *StarCraft II* for AI research purposes, not many other games are made available.

This unfortunately means that those who are interested in applying the techniques on certain games will most likely need to building a clone from scratch. While this is a time intensive process in and of itself, enthusiasts might be pleased with the currently available game engines, which are free and open-source, that are regularly updated. Such engines, or game development environments, include Godot, Cocos2d-x, CryEngine, Openage, and Armory among others.

The game engine

The main limitation presented by the game engine is the difficulty in scaling. Pygame is not intended to be used to simulate a large number of agents, and as such it showed, as with increasing numbers, the game became much slower and less reactive. Ultimately, this is acceptable, as the engine developed is intended for academic purposes solely, however for more elaborate research, either hardware with stronger capabilities is needed, or the game needs to be ported to engines more capable of handling multi-agent environments.

In terms of implementation the behaviours show a flaw. The two conflicting behaviours, seek and obstacle avoidance, lead to odd actions. A certain shakiness occurs initially, after which the agents seem to abandon obstacle avoidance altogether in order to reach the resource.

The simulation

The current implementation of the simulation is not complex. The game has only one real aim, which is to survive by collecting resources. One could argue that a game devoid of competition or other hazards, is not that challenging. We have aimed to introduce such rivalry, indirectly, by implementing selfish strategies. However, this only led to competition within the swarm, which is far from logical from a video game perspective.

Experiments

This part should emphasize first the errors that were made in the experimental setups. First, as mentioned before, the mechanism responsible for the spawning of the food items should have relied less on chance. In the current implementation, the function implemented looks iteratively for empty tiles in the environment. Whenever such a tile has been found, the food is spawned in 10 percent of the cases.

The rationale was that this low percentage would prevent the simualtion to spawn too many mushrooms, which in a number of test runs led to a noticeable slow down of the simulation. However, we overlooked the fact that we did not store the location of the tile before continuing. Due to this, the state search space grew needlessly larger. If we had stored the location of the tile, we could have prevented the few runs in which too little resources were available to the agents.

Secondly, in the laziness experiment we could have added a way in which the inactive agents could have ensured their survival. As could be seen in the case that the entire population was lazy, not one agent survived. To recap, the reason that none survived was that no-one left the base to collect a food item. However, upon running a few test runs afterwards, we noticed that some food items spawned at the tile next to the base.

In hindsight, to have a useful dataset for this strategy we could have changed the rule as follows. Instead of not going into foraging at all, we could have set the wandering radius of the agent to the base only. This changes the experiment in such a way that the lazy agent will not exert itself when looking for food. It simply acknowledges that nearby a food item is present, and without travelling too far it is able to collect it.

6.2 Future work

Future work concerning the proposed framework, should focus on adding more complexity to the simulation. This could be achieved by adding more challenging elements to the environment, such as bodies of water, bridges and fire.

Furthermore, the introduction of competition for survival in the environment could prove to be interesting. This could be achieved with the introduction of another species, which competes with the swarm for the food items.

The food class could be expanded upon. One can imagine food items that may increase the health more than the current mushrooms, or poisonous mushrooms which will have a detrimental effect on the agents. Having such a distinction, could also lead to a more complex reasoning within the agents. For instance, in the case that a food item is deemed more healing, the agent could prefer to be selfish based on that assumption alone.

The implementation in this thesis is in a way inspired by the game of *Lemmings*. We believe the original game would lend itself brilliantly for research in task allocation. In the game, a swarm is tasked with moving from an entrance to an exit location. The path to the exit is dangerous due to environmental threats. Such threats include bodies of fire and lava. Other than threats, hurdles are present that make the journey difficult. These include walls, large drops and bodies of water. In order to make the journey, the agents need to work together. Each agent has its own tasks, which are set at the beginning of the level. For instance, in order to go through walls, a basher agent is required to clear the path.

In a number of our experiments we ended up failing to reject the null hypothesis. We can attribute this mainly to the small sample sizes and the mechanism responsible for food spawning.

7 Conclusion

In this work, we have presented a simulation engine for the development of swarm intelligence. Self-organizing behaviours in the context of task allocation has been demonstrated. Through simple tasks, the swarm was able to achieve the common goal of survival.

By means of the experiments performed in this work, we have shown that the degree of laziness has an effect on the survival of the swarm. A fully lazy population performed terribly, which is logical. In nature, a nest of young animals without the help of older members of the population would also struggle to survive.

There has to be a balance between being lazy and actively foraging for food within swarms. Selfishness could work well enough for a primarily active population, however a lazier population should prefer to share.

Furthermore, through statistical analysis, we have found that changing either size of the swarm or that of the map did affect the performance. This, however, was not clearly observed in the obtained results. We may attribute this to the relatively low number of simulation runs. We do believe that running the experiments with a greater number of simulation runs would also lead to a clear visual effect in the plots.

Moving as a single unit, decreased the exploration space of the swarm, which led to lower population survival.

While we did find that some of our results were not statistically significant, we do feel that our work could be used in future endeavours. As mentioned in the discussion, repeating the experiments for more runs may prove useful to achieve statistical significance.

Overall, we found the following. When looking into selfishness solely we can see a negative effect when selfish. Using degrees of selfishness however, the effect was not significant. The laziness degree was found to positively effect the swarm health up until a fully lazy population. In a hybrid setting in which both strategies are present, the laziness solely affects performance. The selfishness in this case has little to no effect. However, when the hybrid setting is combined with an approach of moving as a single unit, the selfishness affects performance more than the laziness.

References

[1] Claude E Shannon. Xxii. programming a computer for playing chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 41(314):256–275, 1950.

[2] Michael Buro. Improving heuristic mini-max search by supervised learning. *Artificial Intelligence*, 134(1-2):85–99, 2002.

[3] Vadim V Anshelevich. A hierarchical approach to computer hex. *Artificial Intelligence*, 134(1-2):101–120, 2002.

[4] Hiroyuki Iida, Makoto Sakuta, and Jeff Rollason. Computer shogi. *Artificial Intelligence*, 134(1-2):121–144, 2002.

[5] Martin Müller. Computer go. *Artificial Intelligence*, 134(1-2):145–179, 2002.

[6] Gerald Tesauro. Programming backgammon using self-teaching neural nets. *Artificial Intelligence*, 134(1-2):181–199, 2002.

[7] Brian Sheppard. World-championship-caliber scrabble. *Artificial Intelligence*, 134(1-2):241–275, 2002.

[8] Kumar Chellapilla and David B Fogel. Evolving neural networks to play checkers without relying on expert knowledge. *IEEE transactions on neural networks*, 10(6):1382–1391, 1999.

[9] Manfred Milinski and Rolf Heller. Influence of a predator on the optimal foraging behaviour of sticklebacks (gasterosteus aculeatus l.). *Nature*, 275(5681):642–644, 1978.

[10] John T Emlen. Flocking behavior in birds. *The Auk*, 69(2):160–170, 1952.

[11] Simon Goss, Serge Aron, Jean-Louis Deneubourg, and Jacques Marie Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76(12):579–581, 1989.

[12] John H Sudd. The foraging method of pharaoh's ant, monomorium pharaonis (l.). *Animal Behaviour*, 8(1-2):67–75, 1960.

[13] Oebele Herman Bruinsma. *An analysis of building behaviour of the termite Macrotermes subhyalinus (Rambur)*. PhD thesis, Bruinsma, 1979.

[14] Adrian M Wenner, Patrick H Wells, and F James Rohlf. An analysis of the waggle dance and recruitment in honey bees. *Physiological Zoology*, 40(4):317–344, 1967.

[15] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[16] Scott Camazine, Jean-Louis Deneubourg, Nigel R Franks, James Sneyd, Guy Theraula, and Eric Bonabeau. *Self-organization in biological systems*. Princeton university press, 2020.

[17] Eric Bonabeau, Directeur de Recherches Du Fnrs Marco, Marco Dorigo, Guy Théraulaz, Guy Theraulaz, et al. *Swarm intelligence: from natural to artificial systems*. Oxford university press, 1999.

[18] Guy Theraulaz, Eric Bonabeau, and JN Denuebourg. Response threshold reinforcements and division of labour in insect societies. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1393):327–332, 1998.

[19] John McCarthy. Partial formalizations and the lemmings game. Technical report, Citeseer, 1998.

[20] Graham Kendall and Kristian Spoerer. Scripting the game of lemmings with a genetic algorithm. In *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, volume 1, pages 117–124. IEEE, 2004.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[22] Mark Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.

[23] Eric Bonabeau, Guy Theraulaz, and Jean-Louis Deneubourg. Fixed response thresholds and the regulation of division of labor in insect societies. *Bulletin of Mathematical Biology*, 60(4):753–807, 1998.

[24] Michael JB Krieger, Jean-Bernard Billeter, and Laurent Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995, 2000.

[25] Yongming Yang, Changjiu Zhou, and Yantao Tian. Swarm robots task allocation based on response threshold model. In *2009 4th International Conference on Autonomous Robots and Agents*, pages 171–176. IEEE, 2009.

[26] Anderson R Tavares, Hector Azpúrua, and Luiz Chaimowicz. Evolving swarm intelligence for task allocation in a real time strategy game. In *2014 Brazilian Symposium on Computer Games and Digital Entertainment*, pages 99–108. IEEE, 2014.

[27] Carlo Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, 8:3–62, 1936.

[28] Olive Jean Dunn. Multiple comparisons among means. *Journal of the American statistical association*, 56(293):52–64, 1961.

[29] David H Wise and James D Wagner. Evidence of exploitative competition among young stages of the wolf spider schizocosa ocreata. *Oecologia*, 91(1):7–13, 1992.

[30] Tucker Balch. The impact of diversity on performance in multi-robot foraging. In *Proceedings of the third annual conference on Autonomous Agents*, pages 92–99, 1999.

[31] Nathan F Sayre. The genesis, history, and limits of carrying capacity. *Annals of the Association of American Geographers*, 98(1):120–134, 2008.