

Simulating Ant Colonies To Investigate How Minor Worker vs Major Worker Ant Distributions Affect Colony Energy

Harry Howarth, Frazer Bennett-Wilford and Blayze Millward

May 14, 2018



Contents

1	Introduction and Background	1
2	Methodology	3
2.1	System Description	3
2.2	Assumptions	3
2.3	Experimental Setup	4
3	Results	5
3.1	Control Results	5
3.2	Main Results	5
3.3	Computational Efficiency	6
3.4	Repeatability	7
4	Discussion	8
4.1	Future Improvements	8
5	Conclusions	9
A	Appendix State Machines	11
B	Appendix Computational Efficiency Profiling	14

List of Figures

1	Line graph depicting colony energy against iterations (time) for various percentages of Major Worker Ant.	5
2	Bar Chart depicting colony energy against iterations (time) for various percentages of Major Worker Ant.	6
3	Profiling for the child functions of Environment - Environment.step	7
4	Profiling for the child functions of TerrainTile - TerrainTile.step	7
5	High Level State Diagram for the Ant	11
6	State Diagram for the Ant Exploring	12
7	State Diagram for the Ant interacting with the nest	13
8	State Diagram for the Ant returning to the nest	13
9	Profile summary of our Matlab application	14
10	Child functions of the ant-simulation, which contains the majority of the computational cost	14

List of Tables

1	This table details the parameters used for each ant in our experimentation .	4
---	--	---

Abstract

Ant colonies have the unique ability to dynamically search the terrain surrounding their environments, detect and retrieve sources of food. This is an emergent behaviour that has been well explored by a variety of past studies [1, 2, 3, 4, 5, 6] often with a focus on examining the way that paths to and from food sources are created and organised [6, 7, 5]. In reality, many species of ants have a number of different castes of ants within their ranks [8]. This study examines what affect different distributions of two castes of ants in particular – the *minor*- and *major*- – worker ants within a colony have on that colony’s overall health when faced with differing conditions.

An ant colony is simulated in Matlab on terrain that has been descritised into chunks, and food is spawned randomoly throughout the terrain. Ant colonies with differing ratios of simulated minor and major worker ants are created, and their success – as measured by the total energy collected from spawned food sources – is plotted against these ratios in order to attempt to identify the most sucessful distribution of minor and major worker ants.

We found that our model seemed to emulate the behaviour of ant colonies in the real world, with the model indicating that a ratio of around 35% minor to 65% major worker ants is the most successful, reflecting the average distribution of minor to major worker ants seen in real-world mature nests as mesured in a previously published work[9].

1 Introduction and Background

Ants are *eusocial* creatures, meaning that they possess the highest level of organisation that any social species of animal can possess[10, 11]. Behaviour allows them to perform incredible feats as a whole, even when individually each ant has very little actuation over its environment. One such example of this is an ant colony’s ability to forage for food from its surrounding area, while avoiding obstacles and successfully navigating changes in its environment. This ability in itself has been examined a significant amount in past research[5, 6], and has even lead to algorithmic methods inspired by the behaviour being developed to solve problems faced in areas of computer science, such as those similar to the *travelling salesmant problem*[12, 13].

Another characteristic of eusocial creatures – and the focus of our own investigation – is the separation of groups within the colony into differing *castes* that exhibit specialized behaviour, making them more fit for certain types of jobs over others of their colony[11]. Ants in particular are often divided into up to four differing casts: Queens, Minor Workers (Workers), Major Workers (Soldiers) and Drones[14, 15]. According to work performed by W. R. Tschinkel in his journal article on the distribution of worker ant populations within colonies, *Colony growth and the ontogeny of worker polymorphism in the fire ant, Solenopsis invicta** [9], minor workers typically make up 65% of the worker population within a mature colony with major workers making up just 35%. Tschinkel measured these stable distributions only in more mature colonies (colonies over six years of age), in particular, in *fire ant* colonies.

Our experiment is interested in the area of finding the ratio of minor to major worker ants that has the most positive impact on an ant colony’s ability to thrive within a changing environment. It will revolve around an agent based simulation of ant foraging behaviours around a nest in an environment where food is randomly spawned to be consumed by the colony, and the colony can produce two differing types of ants representing the major and minor worker ants.

Agent based models are models, typically of natural systems, that simulate the system they are modelling through the use of individual *agents*. As opposed to mathematical models that aim to simulate a system by generalizing the behaviour of potentially many components into few mathematically described expressions (such as population over time, or growth rate with food consumed), agent based models simulate each component individually and the interactions between them[?], allowing them to simulate potentially unexpected emergent behaviour that is hard to describe using mathematical approaches – this makes them particularly useful when evaluating complex systems that exhibit strong emergent behaviour such as the flocking of birds or, as in our case, the retrieval of food exhibited by ant colonies.

A number of past studies have examined the way that ant colonies search their surrounding areas to locate and retrieve food [5, 6]. Often the these studies focus on simulating ant colonies to apply the emergent behaviour that they exhibit to solve abstracted technical problems [12, 13], with seemingly a smaller number of past works focusing on simulating ant colony behaviour as it is seen in nature [5]

The 2004 work by Vittori et al.[5] examined the way that ants navigate their environment, using trails of deposited pheromones which attract other ants from the colony toward food or back to the colony. This work in particular was of interest to us, as it evaluated the results of the model by comparing them to real ants in a number of experiments. Vittori et al implement a very reduced model for their simulation that allowed each ant to only allowed to move through a relatively small graph of various positions, with each node having a reference to a real-world counterpart and ant behaviour when deciding which path to take not only being based on pheromone depositions but also partially on the angle at which the ant would have to move through to get to the next node. The model also introduced restrictions to certain actions that the ants could take, such as the a maximum number of U-turns an ant could perform on a long branch, allowing the ants a way to backtrack in a similar way to observed behaviour in real ants but not allowing them to get caught in a repeated loop of U-turns.

Panait et al. introduces the concept of a dual-pheromone system in “Ant Foraging Revisited”[6] demonstrating the use of a two-pheromone system allowing for learning of paths to both food sources and the colony’s nest. This is an alternative to many previous systems that used ingrained knowledge within each ant or the environment itself to direct returning ants toward the nest. It appears to have the benefit of better mirroring real-world ants, but does increase total the complexity of the system. Comparing the dual-pheromone method explored by Panait et al.[6] with the more direct methods of behaviour coding described by Vittori et al.[5], it can be seen that there benefits to both systems. Vittori et al. often choose to simplify ant behaviour, resulting in faster models that still appear to reflect behaviour exhibited in the real world, whereas Panait et al. opt for more realistic real-world behaviours based on the assumption that the ants have no intrinsic knowledge of their environment.

However, there are examples of ant species such as desert ants that can find their way back to their nest seemingly using ‘natural pedometers’[?], lending support to the idea of modelling a system in which the agents know intrinsically information about themselves and the position of their nest.

2 Methodology

2.1 System Description

At the highest level our ABM consists of one agent, Ants, and an environment that contains their colony, food and then the pheromones that the ants leave behind. This high level behaviour is described by the state diagram Figure 5 in Appendix A. We have modelled two types of ants, major workers and minor workers. These are called workers, for major workers, and scouts, minor workers, in this project’s code and some of the diagrams.

The system aims to simulate colonies of ants as they forage for food in a 2-D environment. Each colony of ants exists as a single spawn point out of which ants leave to look for food. The state diagram in Figure 6 found in Appendix A details this behaviour. Each tile contains a value for the strength of food pheromone for each colony. Ants can move from tile to tile to follow pheromone trails towards food. If an ant cannot see a food pheromone trail then they will explore away from the nest for new food sources or food pheromone trails to follow. As can be seen in Appendix A Figure 8 Ants will reinforce a pheromone trail when they return to the nest with food. Over time these pheromone trails will decay. This is described by Robinson’s journal article [3] which details the decay rates of varying types of pheromones in ants’ foraging networks. Ants will only reinforce those food pheromone trails that they find food at the end of.

2.2 Assumptions

An assumption made was to not model factors that affect ant speed. One example is ant encounters, this being where two ants move over each other and cause a slowdown. The reasoning for this was that as according to one study ‘direct or interaction effects, has a much smaller effect on walking velocity than does body size’ [2], therefore the speed of an ant is determined entirely by their body size, which is given as the most important factor. Another example is ant size, we are assuming that ants of all sizes move at the same rate.

Additionally, although ants were modelled to have continuous locations, the environment is grid based and so pheromone trails were located by their tile position. The purpose of this model is that it finds statistical patterns based on the proportion of scout/worker ant populations in each colony, therefore, it is not important that the entire system is modelled in a way that is physically accurate on a ground level, rather that the simulation provides results which are statistically representative of ant colony behaviour.

The properties of food were assumed. The two main assumptions that were made here were the generation rate of the food and the decay rate of the food. The generation of the food was done by starting with n tiles of food and then generating n tiles of food in random places in the environment every n iterations. The decay rate of the food was not implemented so food was assumed to exist from creation until it was used up by the Ants.

Another, food based, assumption that was made was that food remained in the same place as it was generated.

2.3 Experimental Setup

This sub-section outlines the various parameters used in our experimentation. It starts by detailing the values used for this models agents, two types of Ant. Where values have been taken from literature, they are clearly cited. Any values that have been assumed have been described in the sub-section above.

The two types of ant are Major workers and Minor Workers. We are modelling them using four parameters: speed, strength max energy level and energy use per iteration. The values for each of these are in table 1. The Speed value comes from the journal article [2] and the fact that they are the same has already been explained. The values for the minor worker’s strength, maximum energy level and energy use per iteration were arbitrarily chosen. However, we used literature to scale the difference between the minor worker and major worker. Tschinkel’s journal article [9] states that the average major worker’s size is 4 times that of a minor worker. We then took this and applied it to the characteristics as seen in table 1.

Table 1: This table details the parameters used for each ant in our experimentation

Ant Type	Speed (mm s^{-1})	Max Energy Level	Energy Use Iteration $^{-1}$	Strength
Major Worker	51.9	600	4	120
Minor Worker	51.9	150	1	30

As detailed in the introduction with reference to W. R. Tschinkel’s journal [9], minor workers typically make up 65% of the worker population and major workers make up just 35%. Our project is investigating how minor worker vs major worker distributions affect colony energy. Hence our experimentation is going to be focusing on changing the amount of major workers in the colony to find the optimum for our colony. The values, percentage of major worker ant, used in experimentation were 35% and 0% to 100% with a step size of 10.

The length of each simulation is 1200 iterations. In real world terms an iteration is 1 minute, this means our simulation is running for 20hours. We allow the ants to move 1 step each iteration at a pace of 51.9mm s^{-1} . Hence the size of a tile is 3.1m by 3.1m. The environment is set to a fixed 50x50 tiles therefore the area that our experimentation is covering is 24025m^2 (155m by 155m).

To produce our results 5 runs, with 5 separate seeds (1:5), for each percentage of major worker ant were done. Each run takes c.60s to complete, so doing this many runs that this level of iterations is not computationally prohibitive.

3 Results

3.1 Control Results

3.2 Main Results

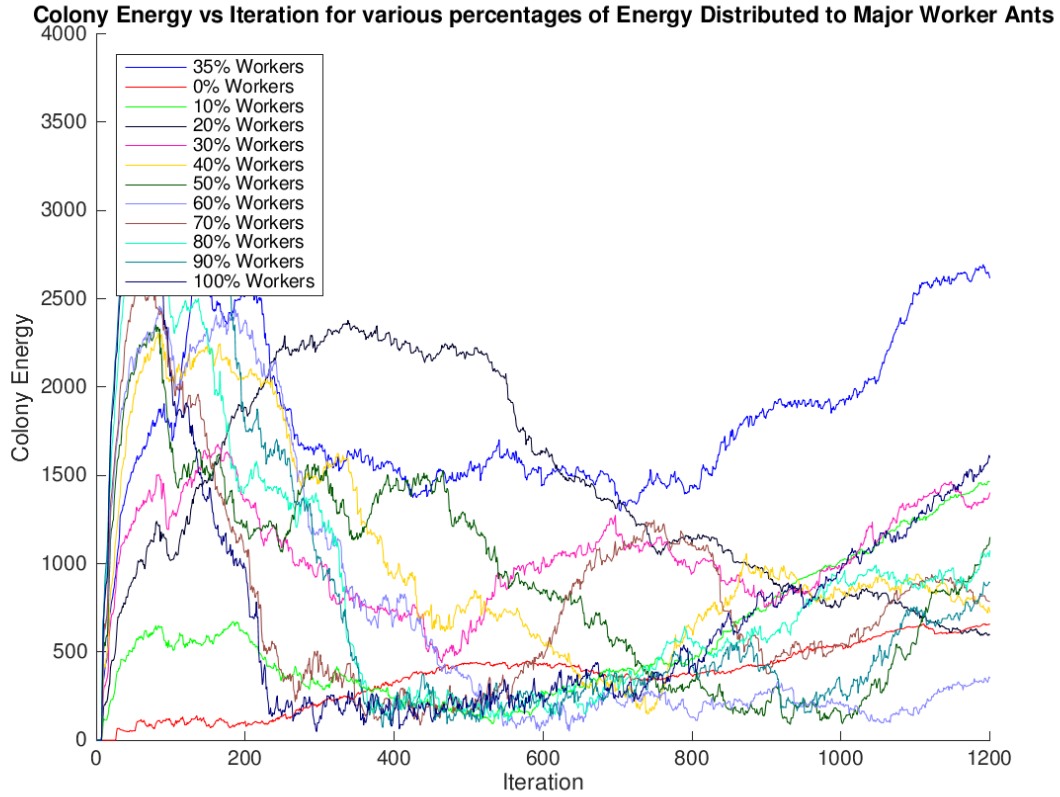


Figure 1: Line graph depicting colony energy against iterations (time) for various percentages of Major Worker Ant.

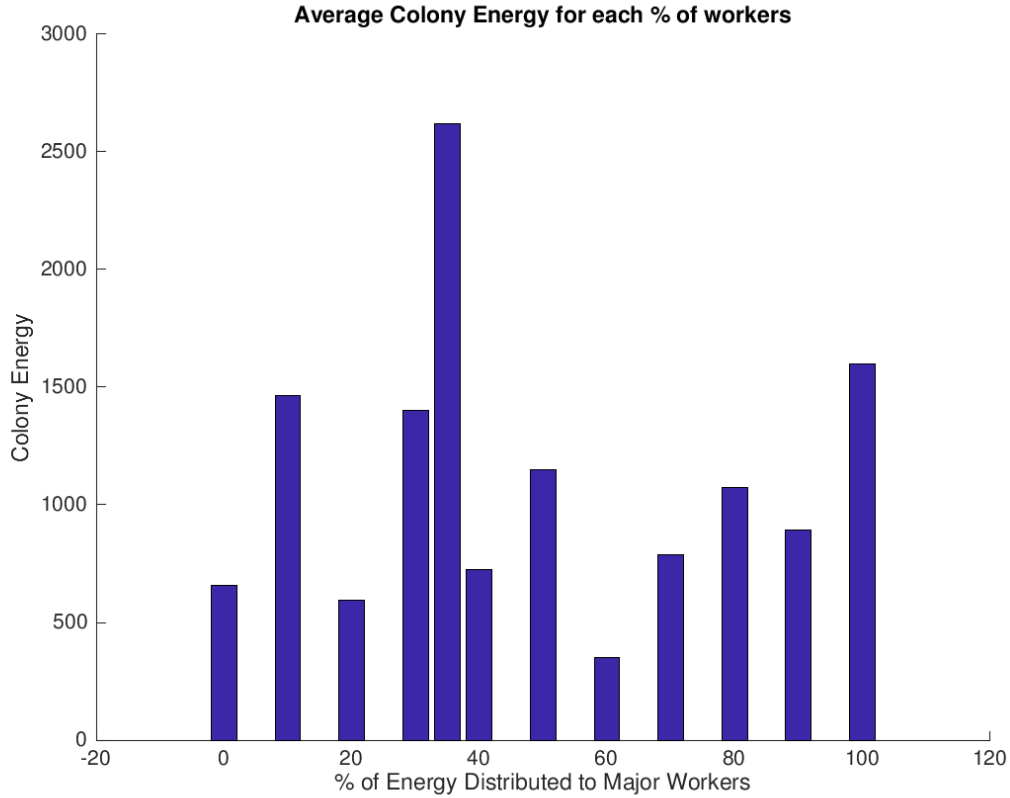


Figure 2: Bar Chart depicting colony energy against iterations (time) for various percentages of Major Worker Ant.

3.3 Computational Efficiency

When running the simulation with graphics enabled the largest portion of computation is spent on drawing the simulations results. In order to find bottlenecks in the simulation itself the profiler was used with the graphics disabled, to give a time plot table of what functions are the most expensive during the running of an entire simulation. Appendix B, figure 9 shows the summary of the system and figure 10 of the same appendix shows that the environment is the place where most computation happens.

When overviewing the computational costs of the environment step, it's highlighted in figure 3 that the terrain tile is where the large majority of time is spent. As shown in figure 4, a large amount of the computation that is done during the simulation of the terrain tile is a result of the pheromone step function. As a result, it can be seen that the pheromone step is taking a large amount of computation to run, the reason for this being so expensive is that each tile of the map contains a pheromone which is then updated each timestep of the simulation. Running it this way means that tiles without active pheromones will still be receiving updates each timestep. However, a way of improving this to greatly reduce

Children (called functions)




Function Name	Function Type	Calls	Total Time	% Time	Time Plot
TerrainTile>TerrainTile.step	class method	2500000	61.226 s	71.0%	
Colony>Colony.step	class method	1000	11.306 s	13.1%	
Environment>Environment.generateFood	class method	20	0.004 s	0.0%	
Self time (built-ins, overhead, etc.)			13.649 s	15.8%	
Totals			86.185 s	100%	

Figure 3: Profiling for the child functions of Environment - Environment.step

Children (called functions)



Function Name	Function Type	Calls	Total Time	% Time	Time Plot
Pheromone>Pheromone.step	class method	5000000	30.581 s	49.9%	
Self time (built-ins, overhead, etc.)			30.644 s	50.1%	
Totals			61.226 s	100%	

Figure 4: Profiling for the child functions of TerrainTile - TerrainTile.step

computation would be to instead keep a list of active pheromones and then only update those inside that list. When a pheromone decays to zero strength, the pheromone can be removed from the list and won't be unnecessarily updated each timestep.

3.4 Repeatability

4 Discussion

4.1 Future Improvements

By improving the computational efficiency of the simulation, tests can be run for longer amounts of time to provide longer term trends and repeated to give averaged results. Implementing the improvement suggested in the computation efficiency section of this report would greatly improve the run-time of the simulation and such improvements to computational efficiency will allow for larger amounts of experimentation.

In order to further improve the model itself, there are several properties of real ant colonies which could be modelled to improve biological accuracy. One example of this is reproduction, by giving ant colonies the ability to reproduce would allow the simulation to run over longer periods of time and provide long term trends of ant behaviour. The current system does not include reproduction and so is only suitable for shorter simulations, where reproduction has very little impact. If a colony were to have a large amount of ants die, then as the environment becomes dense with food, the population will slowly increase again until an equilibrium point is reached. Modelling such behaviour would provide results which are more closely aligned to nature, where colonies do have the ability to produce new ants.

Another area of improvement could be to simulate colony interactions. In the current system several colonies can be simulated simultaneously, however ants from two colonies do not interact. Giving ants the ability to fight each other would allow for colonies to compete for food sources. If the simulation was extended to include ant interactions and reproduction, the simulation would likely depict a more natural representation of ant colony behaviour.

4.2 Modelling Ant Colonies with an Agent Based Model

In order to simulate the behaviour of an ant colony, each individual ant was modelled as an agent and functioned following a list of behaviours. This agent based approach allowed for emergent behaviour to result from a collection of relatively simple rules. This being the main advantage of an agent based model, behaviour becomes easy to modify, understand and the simulation of ants can be extended to include new behaviours with ease. However, the primary disadvantage of using an agent based model in this instance was that by modelling each individual ant, the computational costs become very large. When compared with a mathematical model, when considering the type of system developed, it can be considered appropriate that an agent based model was used because the mathematical equation of such a complex system could not easily be understood, tested and modified.

5 Conclusions

References

- [1] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, and G. Theraulaz, *Self-Organization in Biological Systems*. Princeton studies in complexity, Princeton University Press, 2003.
- [2] M. Burd and N. Aranwela, “Head-on encounter rates and walking speed of foragers in leaf-cutting ant traffic,” *Insectes Sociaux*, vol. 50, pp. 3–8, Feb 2003.
- [3] E. J. H. Robinson, K. E. Green, E. A. Jenner, M. Holcombe, and F. L. W. Ratnieks, “Decay rates of attractive and repellent pheromones in an ant foraging trail network,” *Insectes Sociaux*, vol. 55, pp. 246–251, Sep 2008.
- [4] J. M. HERBERS and E. CHOINIERE, “Foraging behaviour and colony structure in ants,” *Animal Behaviour*, vol. 51, no. 1, pp. 141 – 153, 1996.
- [5] K. Vittori, J. Gautrais, A. F. R. Arajo, V. Fourcassi, and G. Theraulaz, “Modeling Ant Behavior Under a Variable Environment,” in *Ant Colony Optimization and Swarm Intelligence*, Lecture Notes in Computer Science, pp. 190–201, Springer, Berlin, Heidelberg, Sept. 2004.
- [6] L. A Panait and S. Luke, “Ant foraging revisited,” Jan. 2004.
- [7] L. A Panait and S. Luke, “Learning ant foraging behaviors,” May 2018.
- [8] “Caste Terminology - AntWiki.” http://www.antwiki.org/wiki/Caste_Terminology. [Online; accessed 13-May-2018].
- [9] W. R. Tschinkel, “Colony growth and the ontogeny of worker polymorphism in the fire ant, *solenopsis invicta*,” *Behavioral Ecology and Sociobiology*, vol. 22, pp. 103–115, Feb 1988.
- [10] E. Wilson, *The Insect Societies*. Jan. 1971.
- [11] D. Hadley, “What Are Social Insects?.” <https://www.thoughtco.com/what-are-social-insects-1968157>. [Online; accessed 13-May-2018].
- [12] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” *IEEE Computational Intelligence Magazine*, vol. 1, pp. 28–39, Nov. 2006.
- [13] Y. Zhang, Z. l. Pei, J. h. Yang, and Y. c. Liang, “An Improved Ant Colony Optimization Algorithm Based on Route Optimization and Its Applications in Travelling Salesman Problem,” in *2007 IEEE 7th International Symposium on BioInformatics and BioEngineering*, pp. 693–698, Oct. 2007.
- [14] “Ant Family Castes.” <http://antark.net/ant-life/the-family/>. [Online; accessed 13-May-2018].
- [15] Y. Dobrev, “Castes of ants.” <http://www.antday.com/?lang=en&pageid=castes>. [Online; accessed 13-May-2018].

A Appendix State Machines

This section of the Appendix contains the state diagrams that are required for the Methodology section but are too large to include in the actual body of the report.

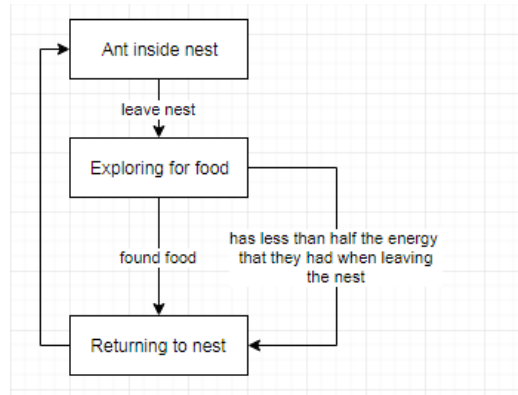


Figure 5: High Level State Diagram for the Ant

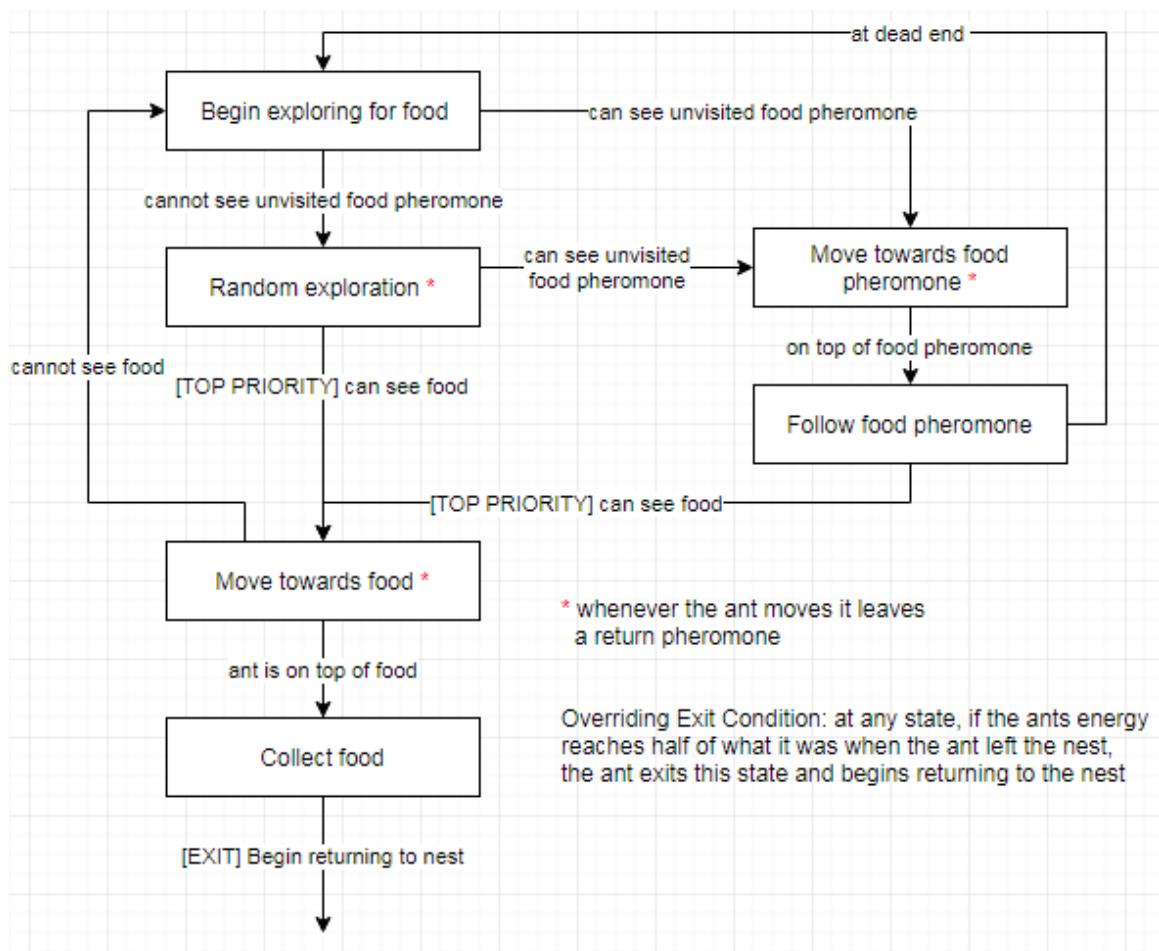


Figure 6: State Diagram for the Ant Exploring

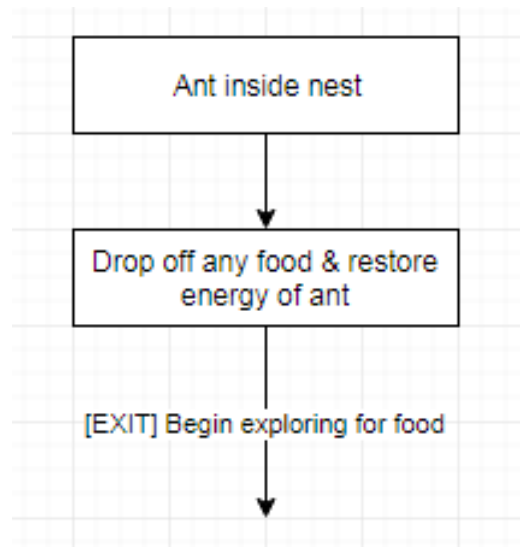


Figure 7: State Diagram for the Ant interacting with the nest

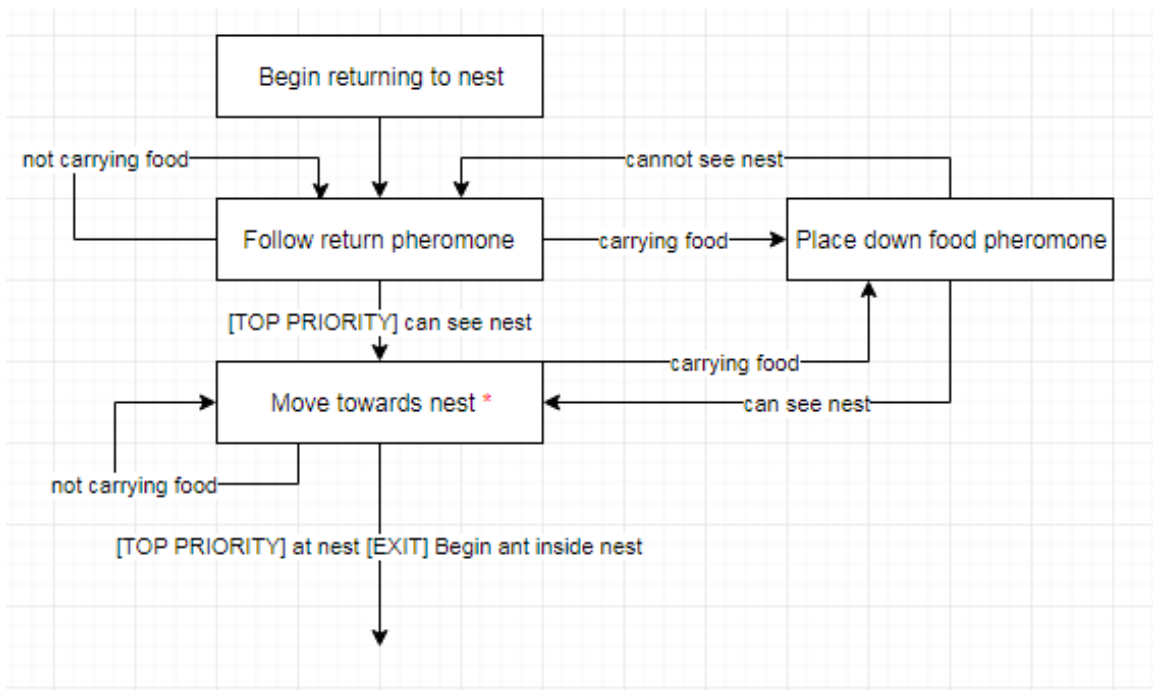


Figure 8: State Diagram for the Ant returning to the nest

B Appendix Computational Efficiency Profiling

This section of the appendix contains some of the relevant computational efficiency profiling screenshots.

Profile Summary

Generated 13-May-2018 22:35:06 using performance time.







Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
ant_simulation	1	87.474 s	0.242 s	
Environment>Environment.step	1000	86.185 s	13.649 s	
TerrainTile>TerrainTile.step	2500000	61.226 s	30.644 s	
Pheromone>Pheromone.step	5000000	30.581 s	30.581 s	
Colony>Colony.step	1000	11.306 s	0.318 s	
Ant>Ant.step	29000	10.985 s	0.408 s	
Ant>Ant.moveStep	22973	6.641 s	0.179 s	

Figure 9: Profile summary of our Matlab application

Children (called functions)


Function Name	Function Type	Calls	Total Time	% Time	Time Plot
Environment>Environment.step	class method	1000	86.185 s	98.5%	
legend	function	1	0.337 s	0.4%	
Environment>Environment.Environment	class method	1	0.247 s	0.3%	

Figure 10: Child functions of the ant-simulation, which contains the majority of the computational cost

Listings