NTNU
Innovation and Creativity

Automatic Configuration for Collective Construction

Automatic parameter setting for response threshold agents in collective construction

Jørgen Braseth

Master of Science in Computer Science
Submission date: June 2007
Supervisor: Keith Downing, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

Since their early days, the fields of swarm intelligence and swarm robotics have been, to a large degree, concerned with autonomous construction by simple agents. Most research, however, has been concerned with construction of simple geometric shapes, and assumed the ideas can be extended to include more advanced structures.

This thesis will design simple, swarming agents capable of constructing complex structures and investigate how these agents can be made to automatically and dynamically adjust their internal parameters to best suit the task of constructing these structures while using only indirect communication and simple rule-sets.

If successful, we will have shown that swarm intelligence systems are in fact capable of carrying out such complex tasks and that even complex tasks can be carried out without having perfect information about the task and the environment in which it is to be carried out.

Assignment given: 15. January 2007
Supervisor: Keith Downing, IDI

Abstract

Swarm Intelligence and swarm robotics is a field in which a lot of research is being done. A lot of this research is concerned with collective construction, construction performed by a set of simple agents working in a distributed fashion. Most of the work done focuses on constructing very simple structures however, leaving a gap in the research, the need for examining the construction of complex structures.

In this thesis, we take a step towards filling this gap. In our preliminary studies, we design a complex structure along with four agent types and evaluate which design is better suited for constructing a complex structure as we define it. We find that the agent design inspired by social insects and using internal thresholds to control its behaviour is the best performer, and set out to enhance this design. To make our agents better suited for distributed construction, we enhance them by giving them wall-following behaviour, along with significantly redesigning their internal threshold model to closer resemble the threshold models employed by social insects.

Performing simulations using versions of our improved agents varying in their threshold-setting methods, we show that different types of structures will test different aspects of agent design and that, to some degree, making the agents work faster will also make them less reliable. From our results, we conclude that our *Variable Limit* agent design, having dynamic internal thresholds, is the design best suited for the task of collective construction. Our results show that the use of dynamic thresholds will enable swarm intelligence systems to perform efficiently and reliably, while requiring only a minimal amount of information about their task and the environment in which the task is to be carried out.

Acknowledgements

I would like to thank my supervisor Prof. Keith Downing for agreeing to be my supervisor, extending help when needed and for supplying useful information throughout the process.

I would also like to thank Erik Axel Nielsen for fruitful discussions and all the people at the "Ugle" computer lab for helping me through the tedious parts by supplying ample amounts of coffee, conversations and barbeques.

Trondheim, June 2007

_____

Jørgen Braseth

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER

1 Introduction

We will not know unless we begin.

Peter Nivio Zarlenga

This chapter gives a brief overview of this thesis and its problem domain. We start by giving a motivation for our work and by giving a description of our specific problem. We end this chapter by outlining the organisation of the rest of the thesis. This thesis is intended for readers with an interest in swarm-intelligence and swarm-robotics. We assume no prior knowledge of the field, though basic knowledge of multi-agent systems and swarm-intelligence will help the reader attain a thorough understanding of our work.

1.1 Motivation

This thesis lies in the domains of swarm-intelligence and swarm-robotics. These closely related domains are concerned with the collective behaviour of multiple, independent agents. The term swarm-intelligence was first coined by Beni & Wang in 1989 in the context of cellular robotic systems and has been made increasingly popular by the advent of well-performing algorithms like particle swarm and ant colony optimisation.

The swarm robotics field has spawned from swarm intelligence because the possibility of having large numbers of simple agents solve complex problems lends itself easily to robotic systems and, when successful, such solutions allows for the mass production of simple robots in contrast with most of today's robotics systems, which often rely on a few expensive robots. Among the earlier uses envisioned for swarm robotics was NASAs proposed system for automatic construction of lunar bases. These days, swarm robotics is still very much concerned with collective construction, and it spans from dreams of automatically constructed mining stations on Mars to the production of nano machines for fighting disease.

A lot of reasearch has been done on collective construction, but very little work has been done into the construction of structures more complex than simple lines and other geometric shapes. This thesis tries to remedy this by taking some early looks at what effects more complex structures have on the ability of swarms to efficiently construct them, with a focus on how one can make the swarming agents in such a way that one can automatically and dynamically set the parameters of the system without the need for thorough analysis of the structure and the environment in which it is to be built.

1.2 Paper Organisation

The rest of this thesis is structured in the following way. We start in chapter 2 by outlining the background for our problem and some theory on the behaviour of social insects on which our thesis is based. Chapter 3 describes our preliminary study along with what work is to be done to improve on our preliminary results. In this chapter we also define the experimental setup and make some assumptions as to what we can expect from our results. Chapter 4 describes our results and discusses reasons and implications for our results, while chapter 5 draws some conclusions based on our results, along with outlining possible further work based on our research.

CHAPTER

2

Background

In the fields of observation,
chance favors only the prepared mind.

Louis Pasteur

This thesis is concerned with collective construction of complex structures by simple, distributed agents. To give our work a context, this chapter will outline some of the theory that form the basis of this thesis, along with related works.

2.1 Collective behaviour

Throughout the ages, cooperation has been one of the defining points of civilization. People have worked together in all things, from defending the cave to constructing skyscrapers. Because cooperation is such a "human" trait, we are fascinated when we observe it in other animals, and we tend to wonder how such simple beings can cooperate so well. In the following, we take a look at how humans employ cooperation in the design of our systems.

2.1.1 Stepwise cooperation

The assembly line, most commonly associated with the *automated* assembly line credited to William C. Klann, who introduced the concept to Henry Ford (see figure 2.1), is an old concept. By assigning one or more workers to each specific part of the production, an assembly line approach allowed for division of labour and the spesialisation of workers to increase productivity. This approach of using specialisation to improve performance in step-wise tasks is a common one, and is used in all parts of society, from the continued use in automated automobile production-lines, to the domain of computers where the stepwise cooperation approach is employd both in the hardware structure of computers,

having specialised components for various tasks like storage and computations, and in the utilising of modularity in software development.

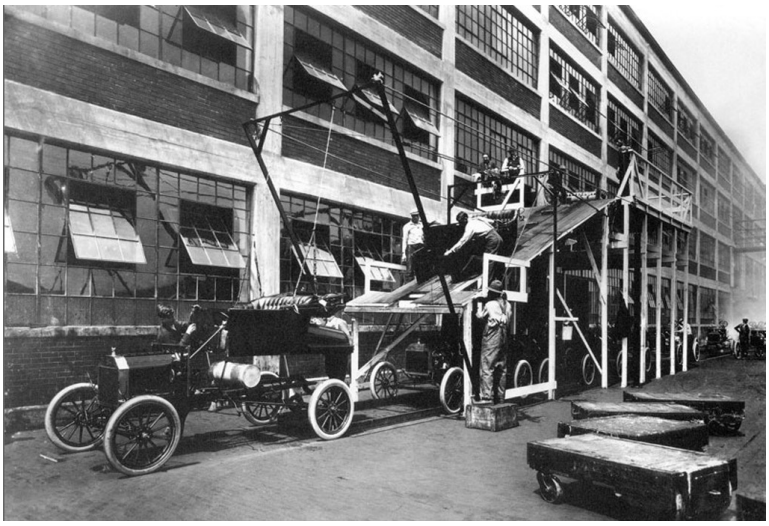


Figure 2.1: Ford Model T assembly line, 1913.

2.1.2 Schools, Flocks and Swarms

The stepwise approach, though a time-proven, effective approach, is not the only approach to performing tasks in groups. Furthermore, this method of labour is rarely observed in nature, though there are some examples, like the production of mRNA in cells [1]. A form of cooperation that we commonly observe in nature, however, is parallell, distributed work done by groups of animals like schools of fish, flocks of birds and swarms of insects. Earlier, these forms of behaviour was thought to be initiated and controlled by mediating individuals, but research has shown that, in most cases, the behaviours are results of simple rule-sets adopted by all individuals. The research into flocking and swarming behaviour has lead to several advances in computers. The research of birds' flocking behaviour has enabled the special effect makers of Hollywood to create realistic animal behaviour like the large groups of bats in the movie Batman Returns. Other uses of this kinds of behaviour include use of swarming behaviour by UAVs in military situations [2] and the use of cellular automata for encryption [3].

2.1.3 Centralised vs Decentralised Control

In multi-agent systems, whether the agents or homogeneous, doing parallell work or heterogeneous specialists working in an assembly-line fashion, a common decision that has to be made is how to coordinate the execution of tasks. This decisions commonly has to to do with employing centralised or decentralised coordination. The usual way of implementing central coordination is to have one agent or a set of agents control the work of the other agents, thereby ensuring reliability and efficiency in the system. A rather mundane example of such centralised control is the centralised traffic control commonly employed by railroads, having a central controller operating all railroad switches and signals to avoid accidents and to maximise the flow of trains [4]. The most commonly stressed problem with systems utilising centralised control is that the controlling unit becomes a single point of failure - if the central control breaks down, the system cannot function. Another problem concerning coordination is

that it creates a bottleneck, the efficiency of the system being limited by the capacity and ability of the coordinating agent.

To remedy the problems of centralised coordination, several multi-agent systems use decentralised control, relying on inter-agent communication in some form to control the behaviour of all agents. An example of this is the routing algorithm employed by the internet protocol (IP), which relies on routing tables describing the network topology being propagated through network nodes to maximise the flow of network traffic [5]. By eliminating the central coordinator, such systems do not have a single point of failure and, as such, can be expected to be much more reliable and fault-tolerant than central coordination systems.

2.1.4 Division of Labour

In any system having multiple entities performing multiple tasks, one has to have some form of divison of labour, either a basic "everyone does everything" approach or more commonly, setting certain individual to perform certain tasks. Plato attributed divison of labour to natural differences in individuals in his work *The Republic*, while Adam Smith, in his *An Inquiry into the Nature and Causes of the Wealth of Nations* considers division of labour to be the cause of economic progress.

Stating that division of labour raises efficiency of task execution should not cause much debate these days. There are, however, different methods of achieving labour division both in natural and artificial systems, two of which we will discuss currently.

Market Based Specialisation

In many multi-agent systems, groups of agents are able to perform all, or a subset of the tasks to be performed. In such systems, a popular approach is to employ market based job allocation, letting the agents decide between themselves which agent should perform which task. This is often done by adopting auction-based approaches, letting agents bid on tasks, assigning the tasks to the highest bidder [6]. This approach, though it is partially decentralised, still periodically requires one of the agents to coordinate and handle the bidding, to determine which agent is the highest bidder for a certain task.

Polyphenism

In contrast to the market based approach mentioned above, social insects employ polyphenism[1] to facilitate job allocation, relying neither on a central, coordinating individual nor direct communication between individuals. Polyphenism and the underlying mechanisms are explained in further detail in section 2.2.8.

2.2 Nature as Inspiration

For many years, scientists have been fascinated by how nature manages to solve problems; termites build huge hives, bees cooperate to find food and even mold seems to exhibit some sort of organised and complex behaviour. The fact that such simple creatures can form such complex and advanced behaviours has made computer and robotics researchers look to this domain for inspiration.

[1] The web-page EverythingBio defines polypheism as *"The occurrence of several phenotypes in a population which are not due to different genetic types. Maybe caused by environmental influences."*

Early in the nineties, a PhD Graduate named Marco Dorigo, inspired by ant foraging, invented the metaheuristic called Ant Colony Optimisation (ACO) and thereby made a major contribution to the field of Swarm Intelligence [7], a term coined earlier, in 1989, by Beni & Wang working on cellular robotic systems [8]. The advent of ACO and its varieties, and the fact that algorithms based on these ideas seem to work exceptionally well, has spurred further research into nature-inspired systems.

Below, we describe some examples of natural behaviours and topics that are interesting in the context of this project.

2.2.1 Ant Foraging

A large field of study, and also the inspiration for Dorigo's ACO algorithm, is the foraging behaviour of some species of ants. While foraging, the ants wander at random, looking for food. While walking, the ants lay a trail of pheromones. When the ant finds a source of food, it returns to the hive[2], still laying down pheromone. When a wandering ant happens on a trail of pheromone, it is inclined to follow this trail. The outcome of many ants exhibiting this simple behaviour is the creation of a "highway" connecting the hive to the foodsource, with ants traveling back and forth along the path, gathering the food. [7, 9, 10, 11]

2.2.2 Slime molds

Slime molds are special types of protists[3] that normally act in the form of amobea, individually looking for food and reproducing by means of dividing. When food gets scarce however, the slime molds' behaviour changes drastically - they stop reproducing and instead move towards each other, gathering in large clusters of tens of thousands of cells. These clusters[4] then move towards a location that is richer in food. When a cluster finds such a place, it creates spores, and thereby starts a new lifecycle of slime molds at the new location. Earlier, scientists assumed the clustering process was coordinated by a "pacemaker" cell, but recent studies has shown that all the slime mold cells are in fact homogeneous, and that the clustering is mediated by the individual cells' excretion of a certain pheromone and the subsequent movement in the direction of the strongest perceptible source of this pheromone.[11, 12]

Figure 2.2: Slime mold on O'ahu, Hawai'i. (Photo by Eric Guinther)

[2]The exact way of finding the way back, visual cues, pheromone, etc. varies between species.
[3]Complex-cell organisms that are classified as neither animals, plants, nor fungi.
[4]Called pseudoplasmodiums

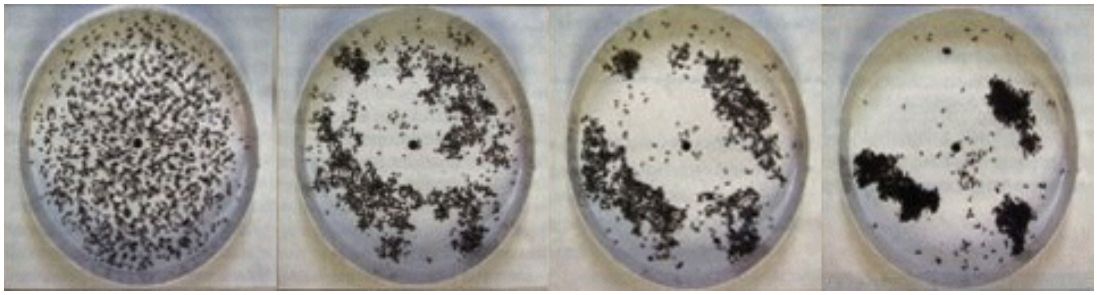


Figure 2.3: Corpse clustering done by a colony of *Messor sancta* ants, showing pictures at 0h, 3h, 6h and 36h respectively. (After Bonabeau, Dorigo, Theraulaz [9])

2.2.3 Corpse clustering

Corpse clustering is another type of clustering happening in nature. Several types of ants pick up and gather their dead breathren in piles, apparently without any form of cooperation, so how do they do it? This time, the answer is not necessarily pheromone, as simulations mentioned later in this paper have shown. Experiments with the *Messor Sancta* ant have shown that if subjected to an environment with randomly distributed body-parts of dead ants, the live ants start forming cemetery clusters within hours (figure 2.3). Similar behaviour has been shown in other species of ants, and is not restricted to only corpses, but is also evident in sorting of larvae and food in some species of ants and in the beginning phase of the construction of a hive by collecting dirt pellets in some termites. [9]

2.2.4 Hive building

Pheromone trails and clustering are not the only ways of guiding work in the world of insects. Insects also utilise other factors in the environment and are guided by these factors. Many ant species rely on temperature and humidity gradients while building and working in their hive, while the termite species *Macrotermes subhyalinus* use the queen's body shape and pheromones emitted by the queen to construct the royal chamber. The construction of said chamber has been shown experimentally to be bases on the gradient of the queen-pheromone - there exists a pheromone gradient window inside which a worker will deposit a dirt pellet if it is carrying one. By utilising this simple rule, the termites build a wall around their queen.[9]

2.2.5 Emergence

Websters online dictionary gives, as one of the definitions of emergence, "becoming apparent".[13] Emergent behaviour can be readily observed in everyday life, in the behaviour of traffic jams and the flocking of birds [11, 10]. It can be described as *complicated behaviour as the result of the collective acts of simple agents*, or in other words, the total behaviour is more than a sum of its parts. Emergence is also clearly visible in all of the examples given above - the ants create a path to the food, the slime mold gathers in a cluster, a royal chamber is built, etc. The notions of emergence and emergent behaviour are central in the world of social insects and are therefore also central in the artificial systems based on these natural systems. [9, 10]

2.2.6 Pheromones and Stigmergy

Several naturally occuring behaviours, some of the ones described above among them, are based on the use of pheromone as a mediator of the behaviour. The ants use pheromone to mark a path to the food source, while the mold uses pheromone to locate other mold-cells. The act of placing pheromone on the ground and relying on this trail for communication between the individuals in the population is a form of stigmergy[5] that enables the colony as a whole to show the behaviour of shortest-path finding and population-clustering in the two first examples above. The use of pheromones and stigmergy along with relying on emergent behaviour are main characteristics of most artificial systems based on swarm intelligence.

2.2.7 Templates

As mentioned above, insects seldom rely purely on the factors they themselves create, like pheromone trails and clusters of corpses, but also on external factors or "prepatterns" like temperature, moisture and pheromone gradients. These prepatterns in the environment can help organise and direct the workflow in a colony of insects. [9, 11] The notion of templates can also be utilised in artificial systems, to help direct work in the right direction. [9, 15, 16, 17]

2.2.8 DivisionOfLabour

As with people, animals, too, exhibit division of labour in task performance. In our context, the divison of labour in social insects, in the form of polyphenism, is especially interesting, as we will see in the following.

Polyphenism

Several types of social insect exhibit polyphenism in division of labour behaviour when executing tasks. While large colonies often have polymorph populations, that is, physically specialised individuals for doing various tasks, small colonies with apparently homogeneous individuals also show forms of division of labour [18]. In [9], the authors refer to to three basic forms of divison of labour shown in social insect societies, described briefly in the following.

Temporal polyethism This form of division of labour bases the division on the age of the individuals, grouping the individuals into *age casts*, where the different casts perform different tasks. An example of this kind of labour division is found in honey bees, where workers between their 6^{th} and 14^{th} day of their life produce royal jelly, forcing them to specialise in feeding the queen larvae. Later in their life, certain cells on their abdomen start producing wax, turning the workers into specialist comb-builders. [18]

Worker polymorphism Worker polymorphism is, perhaps, the most obvious form of labour division and is, among other places evident in the polymorphic ant genus *Pheidole*, where the colony is divided into two morphological subcastes: the minor and the major. The majors (also called soldiers) have a different morphology, namely larger mandibles, than their smaller minor relatives. This difference in

[5] An expression coined by Grasse in 1959 while studying termites as "Stimulation of workers by the performance they have achieved", it describes the indirect communication between individuals by modification of their environment.[9, 7, 14]

morphology makes the major ants more suited for defending the hive, while the smaller worker ants are better suited for carrying out other tasks, like feeding larvae and foraging.

Individual variability Temporal polyethism and worker polymorphism is not always enough to explain the work allocation done in social insects. In experiments done by E. O. Wilson [19] it was shown that in the case of the ant genus *Pheidole* mentioned above, if enough minors were removed from the colony, after a while, the majors start doing the jobs normally associated with the minors. This behaviour has been explained by the use of fixed respons thresholds by Bonabeau et al., as described in further detail below. To describe a group of individuals performing the same task, one usually speaks of a *behavioral caste*. [20, 9, 21]

Response Thresholds

To explain Wilsons findings in [19], Bonabeau et al. developed a simple model relying on fixed response thresholds [22]. In Bonebeau et al.'s model, every individual has a response threshold for every task, making them engage in a given task if their stimuli for that task is above their threshold limit. This model is sufficient to explain Wilson's findings, as removing minors from a task will cause the stimulus for this task to increase until, at some point, it exceeds the threshold for this task in the majors, causing them to engage in this task, though they usually are not inclined to do so. In [9] the authors formally define a response threshold as a function consisting of s, the stimulus for a given task, and θ, the threshold for said task. They stress the importance of such a function being designed such that the result $T_\theta(s)$ will be low for s « θ and high for s » θ, making the probability of an individual adopting a behaviour high if the stimulus for that behaviour is high, and low if not. The authors continue by suggesting two possible, normalised functions. They also point out that exponential response threshold functions are likely to occur in nature, and have been experimentally found to be plausible for the task of ant cemetery creation.

2.3 Related Work

A lot of work has been done based on social animal behavior. Some systems are inspired by how animals behave in the real world (e.g. [7, 9]), while others simulate the animals' actual behaviour to learn more about the animals and their ways of cooperation (e.g. [11, 9]).

2.3.1 Simulations

Handling the latter first, we take a look at two examples of simulation of social animals done by Resnick in [11] which have also been done in other settings. An important observation to make with these kinds of simulation is that they show that the agents of a system do not necessarily have to be smart for the system to behave intelligently, as anthrophomorphised by Aunt Hillary, the ant-hill that communicates with the anteater in Douglas Hofstadter's book, *Gödel, Escher, Bach: an Eternal Golden Braid*.

Termites and Woodchips

Inspired by how termites start the building of their nest, Mitchel Resnick and one of his students did a simulation of termites gathering wood chips (Figure 2.4). Implemented in the scripting language StarLogo, the termites were given a simple set of rules:

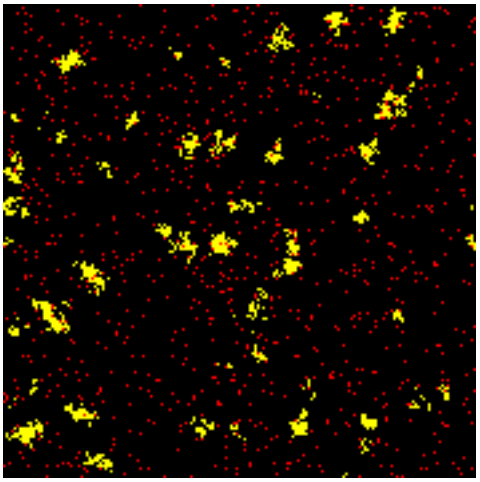


Figure 2.4: A simulation of termites gathering woodchips done in StarLogo. Red dots are termites, yellow dots are wood chips.

- If you are not carrying anything and you bump into a wood chip, pick it up.
- If you are carrying a wood chip and you bump into another wood chip, put down the wood chip you're carrying.

Implementing and running a simulation with termites doing a random walk and following these rules, Resnick showed that the termites would gather the chips into a gradually declining number of gradually larger sized piles. An important note to this simulation is that due to the simplicity of the rules, at any given point, a large amount of the wood chips are not on the ground, as they are being carried around by termites. Resnick rectified this by adding a rule protecting large piles, resulting in a larger number of piles than the previous experiment in most cases, but having, in the end, all chips placed on the ground.[11]

Slime Mold Simulation

Resnick, in [11], also simulates slime mold clustering behaviour. Using another simple set of rules and implementing the mold cells in StarLogo (figure 2.5). The simulation done by Resnick shows that the complex clustering behaviour displayed by the mold cells can be explained by simple pheromone-gradient following rules, combined with the excretion of pheromone by the cells.

Growing Complex Arcitechtures

Bonabeau, Theraulaz and Cogne [23] have done experiments to investigate the possibility of constructing complex structures in a 3D lattice, using only very simple agents behaving deterministically depending on the local configuration of bricks in their environment. Because the space of possible rule sets is so large, they employ a genetic algorithm to explore this space. Running some hundred generations of their genetic algorithms, they discovered rulesets producing arcitectures similar in style to natural wasps nests, and other well defined structures. Their results show that it is possible for agents relying only on a very limited set of rules and stigmergy (see section 2.2.6) to construct very complex architextures. Their applied technique is limited to investigating what kinds of rules lead to which structures, and does not, for now, support the design of rules based on a target structure.

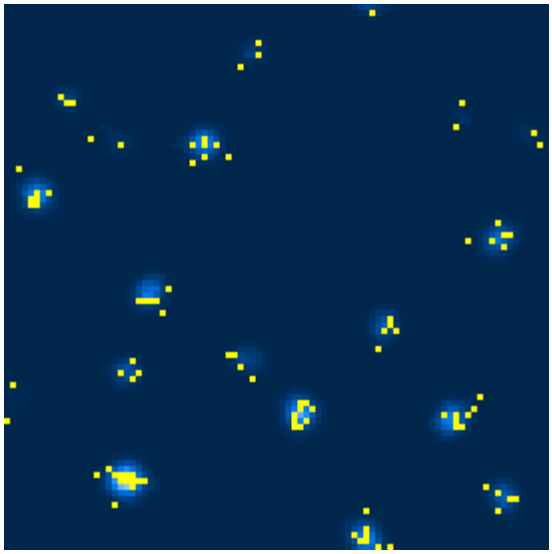


Figure 2.5: Simulation of slime mold cell-clustering done in StarLogo. Yellow dots are mold cells, pheromone intensity is shown in shades of blue

2.3.2 Inspiration

Though a lot of work is being done simulating the behaviour of social animals, an even larger field of research is the biologically *inspired* systems field. This field of research concentrates on designing algorithms and methods for solving problems based on phenomena observed in actual biological systems. Among the most prominent examples of such work are evolutionary computing, neural nets and ant colony optimisation, along with the research field of swarm robotics, some of which are discussed in this section.

Ant Colony Optimisation

Ant Colony Optimisation (ACO) was first proposed my Marco Dorigo in 1992, based on E. Bonabeau's work on virtual ant foraging, and is, like Bonabeau's work, inspired by the way ants forage for food, as described earlier in this paper. This metaheuristic was designed to solve combinatorial optimisation problems like the traveling salesperson (TSP) and the quadratic assignment problems (QAP) [24]. Based heavily on artificial pheromone trails and artificial "ants", often augmented with various local search algorithms, ACO has proven to perform as well as, and in some cases considerably better than other methods for solving static problems like TSP and QAP, while outperforming most of the alternatives in dynamic problems like network routing. [7, 24]

Data sorting and clustering

Other ant-based approaches have been made to explore the data in databases. These approaches are often based on ideas taken from corpse clustering (see section 2.2.3) and brood sorting (see [9] p.151-152). One of these applications is the work done by Lumer and Faieta on clustering similar customers of a bank in the bank's financial database, enabling the bank employees to compare similar customers, allowing them to determine e.g. whether or not a given customer might be able to repay his loan, even if that customer has never had a loan before [25]. Ant clustering algorithms for data-mining have also

Figure 2.6: Picture of cathedral termite mounds in the Northern Territory, Australia. (Taken by Brian Voon Yee Yap)

been explored by other researchers, and Dorigo et al. has devised a data clustering algorithm called ATTA for this purpose [26].

Swarm Robotics

The field of swarm robotics is a constantly growing one, and has been described on *Swarm-Robotics.org* as *"... the study of how large numbers of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment"* [27]. An often thought of use for such systems is the collective construction of some kind of structure, often based on the hive-building behaviour shown by termites in nature (figure 2.6). A simple example of such building behaviour is the wood chip piling mentioned earlier in this paper. Examples of more advanced construction is shown in [15, 16, 17], where a template (see section 2.2.7) is used to guide the construction of regular structures like lines and circles. In [17] the authors use a spatio-temporal template to gradually build a linear structure, while they in [15] use a simpler, static template to build a circular structure. In [16] they demonstrate the validity of their spatio-temporal template mechanism by constructing more advanced structures and posing the empirically supported claim that this method can be used to construct any planar structure.

CHAPTER

3

Research and Experimental Setup

The power to question is the basis of all human progress.

Indira Gandhi

To investigate the possibility of dynamically configuring our agents, we will perform several simulation runs. In the following, we describe our preliminary studies along with the improvements done to our system and the experimental setup for our simulations. We conclude this chapter by pointing out key aspects for the agents and the expectations we have for the experimental runs.

3.1 Preliminary Studies

In [28], the preliminary work for this thesis is described. The paper investigates the demands set by complex structures on swarm agents designed to construct complex structures. In the following, we briefly recap the process and results of the preliminary studies.

3.1.1 The chosen structure

In [28], a *complex structure* was defined using a set of requirements: it can not be describable by a simple polygon or a single geometric function such as a circle, a line, etc. and it must consist of at least one closed-loop wall contained by another. The final structure designed is shown in figure 3.1, consisting of two square "wheels" connected by four spokes.

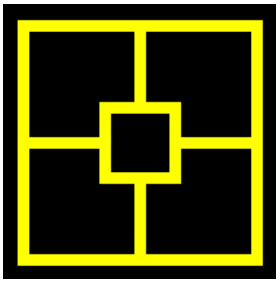


Figure 3.1: The target structure of the preliminary study

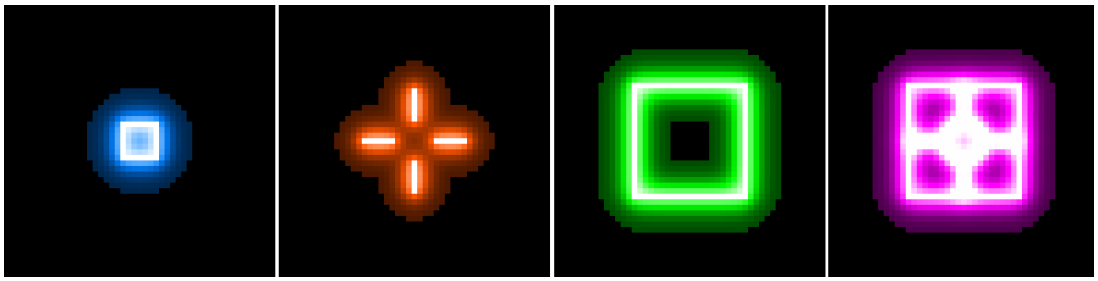


Figure 3.2: The pheromone templates for (from left to right) the inner wheel, the spokes, the outer wheel, and the total pheromone. The gradient shows the amount of pheromone in each grid-cell, lighter colors indicating higher amounts of pheromone. White cells have pheromone at or above the limit for the agents to place blocks.

3.1.2 Templates

To facilitate the construction of the chosen structure, we defined three templates, one for each of the parts of the structure. These templates were deployed by placing a simulated pheromone for each of the structure-parts in the environment and subsequently diffusing this pheromone, thereby creating a gradient pheromone field for the agents to follow. Figure 3.2 shows these pheromone fields as they appeared in our simulation environment.

3.1.3 Designing the Agents

In order to design a set of suitable agents for our system, five requirements where defined which any agent had to fulfill:

1. Allow parallel construction.
   This will ensure that no agent has to wait for another to finish before it can perform its task.
2. Minimise inter-agent conflicts.
   The agents minimise the hindering other agents' progress, e.g. by building a wall around an unfinished feature.
3. Maximise the portion of the structure completed.
   Optimally, the whole structure should be completed.
4. Finish within reasonable time.
   The agents' behaviour should make them utilise the environment to make the construction of

the structure finish within reasonable time. This means that, for example, a random picking up and dropping of blocks would not be acceptable even if it could potentially result in the desired structure.

5. Once finished, the structure must stay finished.
   Agents should not pick up blocks once they have been placed in a correct position.

Four agent species were designed, all fullfilling the requirements to varying degrees. The species designed where named *Uniforms*, *Specialists*, *Timebasers* and *Thresholders*, referring to each of their behavioural configurations, as described in more detail later.

Though the species vary in several ways, they all follow the same basic rules and behaviours and perform a common sequence of actions for each time-step. This sequence of actions can be divided into three steps: action, move and orient.

1. Action
   Some specifics of this step vary between the species, but common for all of the species is the decision in this step to pick up or drop a block if possible. First checking the pheromone level in front of the agent, the agent decided to either:
   - Pick up block
     If the cell is occupied by a block, the pheromone level is below the limit for block placement and the agent is not carrying a block
   - Drop a block
     If the cell is not occupied, the pheromone level is above the limit and the agent is carrying a block.
2. Move
   The agent moves one step forward if possible. If not, it stays in its current position, but turns in a random direction.
3. Orient
   The behaviour in this step is dependent on the species and state of the agent. If an agent is carrying a block, this step has a searching aspect to it. If an agent is not carrying a block, a "wiggle" is the only part of this step, randomising the direction of the agent by turning it to a random angle in the arange [-90°,90°] with respect to its current heading.

This simple three-step process repeatedly executed, along with the variations caused by the agent state and species enables the agents to construct the complete structure. The state diagram for the agents is shown in 3.3.

As mentioned, though the basic behaviour of all the agents were similiar, they varied in how job allocation was performed and whether or not the agents were made to be spesialists or generalists. The differences between the agents are described in the following.

Specialists vs Generalist

The first area in which the agents varied was the area of specialisation. As three parts were to be constructed, the choice was to either let each agent build only one part, i.e. setting, at creation, which part of the structure the agent was to build, or letting all agents be able to build all the parts, initialising them to one of the parts and letting them vary which part they would build as time passed. The *Specialists* species employed a static type-description, making each agent a fully specialised builder of its part of the structure. The agents of the three other species were all able to construct any part of the structure.

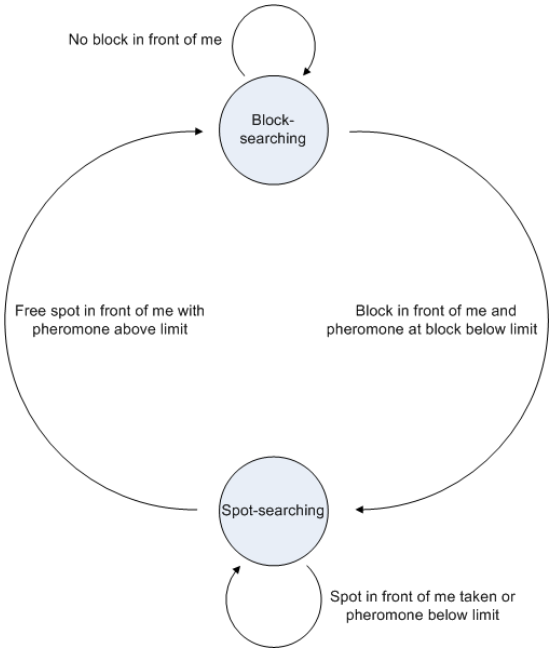


Figure 3.3: The state diagram for an agent

Job allocation

The second area in which the agents varied was job allocation. The *Uniforms* species employed a simple method of pheromone-search that made it ignore which part it was constructing at any time. The three other species implemented nature-inspired job allocation methods. The *Timebasers*, *Specialists* and *Thresholders* species adopted simple versions of the Temporal polyethism, Worker polymorphism and Individual variability job allocation methods(see section 2.2.8), respectively.

3.1.4 Preliminary results

Performing 20 simulation runs for each of the species, we were able to evaluate the species with relation to several aspects, among them reliability and efficiency. The *Uniforms* species was the poorest performing species, though this was expected, as it was designed to be a benchmark for the other species. The *Specialists* species had high efficiency at the start of each run, but suffered at low reliability and high completion time, as this species completely ignores the need for a partially ordered building sequence for the structure. Both the *Timebasers* and the *Thresholders* species, however, performed very well, having both high efficiency and high reliability once their parameters were set correctly. In the end, though, the *Thresholders* species was deemed the best approach as the parameters only need tuning with relation to the structure, in contrast to the *Timebasers* species with which one also needs to take the size of the environment into account for the parameters to be set right. Figure 3.4 shows the average completion times for the various species, clearly showing the good performance of the two stepwise agent-types, showing the *Thresholders*' slightly lower mean completion time, but slightly higher uncertainty. Note that the *Uniforms*' apparent good results in this figure is that this species only ever completed the outer part of the structure, and so this time is shown.

Because of its good results in the preliminary studies, along with its parameters relying only on the structure to be built, the *Thresholders* species was chosen to be improved upon in this thesis.

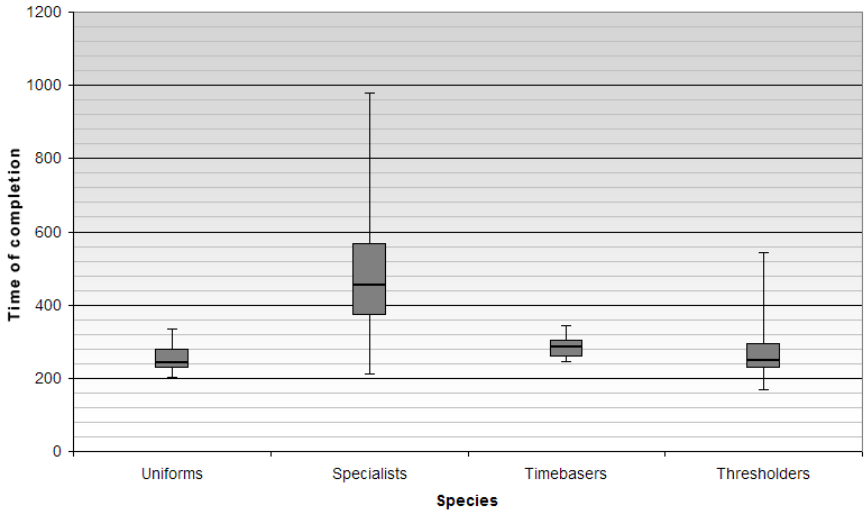


Figure 3.4: The average time spent completing a run for each of the agents of the preliminary studies.

3.2 Improving the design

As the goal of this thesis is to improve on the *Thresholders* species and examine if and how the parameters can be set in advance of construction, we first set out to improve on the very basic *Thresholders* species designed in the preliminary studies, to make it somewhat better suited for the task. To improve the performance of the agents, we implement two improving measures. First we include a simple-following behaviour in the agents, and secondly and most importantly, we drastically redesign the threshold functionality of the agents. Both these measures are described in the following.

3.2.1 Wall following

One of the main problems in the preliminary studies was the significant amount of reduction in construction-speed when the structure neared completion. We found that this was due to the agents' strict pheromone gradient following behaviour, which made the agents tend to move towards the center of the walls in the wheel-parts of the structure, resulting in the extremeties of these walls to have a significantly longer building time. To remedy this fault, the agents were given a simple wall-following behaviour that made them tend to turn right whenever they met a wall.

3.2.2 Advanced Response Thresholds

A second shortcoming of the original *Thresholders* species, was its simple state-switching behaviour. An agent could only change to the next type in a fixed series, it could never change back, or skip to a later stage. To make the agents more adaptable, a more advanced response threshold model was designed, making each agent keep track of the stimuli and threshold for each individual type, thereby enabling the agents to switch to the most apropriate type at any given time, instead of following a predefined order. This response threshold model is further explained in section 3.3.3.

3.3 Agent Design

We design four new agent types, all differing in their threshold functionality. These agents are the *Narrow*, *Wide*, *Complexity Dependent* and *Variable Limit* agents, they will all be described in the following, along with the similarities and differences between them.

3.3.1 Common Behaviour

Similar to the agents of the preliminary studies, the basic operation of agents can be divided into a stepwise process like the one described in section 3.1.3, consisting of the *action*, *move* and *orient* steps. In our new agents, however, a new step is added: *sense*. In this new step, the agent checks its surrounding environment and subsequently adjusts the stimulus for the different structure-parts. The agents then check to see whether or not they should change their behaviour, and if so, what behaviour they should choose to adopt. The new stepwise process, then, is described below:

1. Sense
    - Update stimuli.
      The stimuli for each of the structure-parts are updated, as described in section 3.3.2 below.
    - Change behaviour according to current stimuli.
      The agent might decide to change its behaviour, according to the values of its threshold functions, as described in section 3.3.6.
2. Action
   Identical to the agents of the preliminary study, as described in section 3.1.3
3. Move
   Identical to the agents of the preliminary study, as described in section 3.1.3
4. Orient
   Identical to the agents of the preliminary study, as described in section 3.1.3

In addition to this basic behaviour, all agents employ a simple wall-following behaviour as described above. The state diagram of the agents is identical to that of the preliminary study agents, as shown in figure 3.3

3.3.2 Stimulus

The question of when, how and how often to modify the stimulus for the agents is a hard one, and many possibilities are available. In nature, examples of such stimuli are larval pheromones in the air stimulating ants to feed their larvae and encountering dead hivemates stimulating honey bees to engage in corpse removal [9]. For our agents, we have chosen an approach close to the latter, having the stimulus for a certain part of the structure increase for an agent every time that agent happens upon a free space in its environment where there should be a piece of the given structure-part. The stimulus decreases every time an agent fails to place a block for a certain type because the spot it has found is already taken. This way of varying agent stimulus should ensure that once one part of a structure nears completion, the agents building said part will experience lowering stimulus, eventually causing it to change behaviour, thus making it start construction of parts of the structure that are in greater need of builders.

3.3.3 The Response Threshold Model

The new, improved response threshold model is based on the model defined by Eric Bonabeau et al. as described in section 2.2.8. We wanted an exponential function, making it similar to thresholds response functions observed in nature, and we wanted an easily calculable function. To this end, we have adopted one of the two functions from [9], function 3.1. The function is exponential, it is also normalised for s > 0. As the function only contains two variables: s, the stimulus, and θ, the threshold, it is relatively easy to calculate, and contains no additional parameters that need setting. Our function is plotted as a function of s for different values of θ in figure 3.5. All of our agents use function 3.1 as their threshold response function, only varying in how θ is chosen and whether or not it is modified over the course of the run. In practice, each agent calculates its response for all of its given stimuli and thresholds and then picks a behaviour at random through a roulett pick, giving the behaviours with high $T_\theta(s)$ a larger probability of being picked. This means a behaviour with no stimulus will never be picked, while a behaviour with s » θ will be likely to be picked.

$$T_\theta(s) = 1 - e^{(-s/\theta)} \tag{3.1}$$

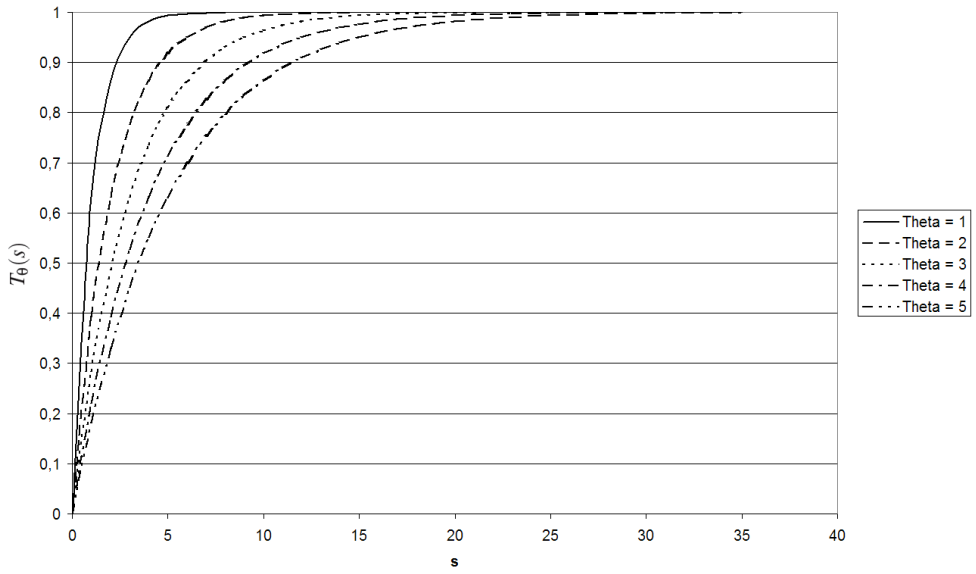


Figure 3.5: The threshold function for various values of θ

3.3.4 The Fixed Threshold Designs

As stated above, all our agents use function (3.1) as their threshold response function and vary only in the value of their threshold and in whether or not these thresholds are modified over time. Three of the designed agent-types have fixed threshold values, i.e. the thresholds stay the same for each of the parts throughout the run, while the last agent type varies its thresholds as time passes. The three fixed threshold agent types are the *Narrow*, *Wide* and *Complexity Dependent* agents, so called because their thresholds have values set close together, far apart or according to the number of blocks in each part of the structure respectively. The lower the "density" of these thresholds, the further to the right along the stimulus axis the high thresholds will be shifted, thus requiring higher stimuli for those behaviour

Type	θ
Narrow Fixed Limit	2 × part-number
Wide Fixed Limit	10 × part-number
Complexity Dependent Fixed Limit	part-size × part-number
Variable limit[1]	2 × part-number

[1] Initial values

Table 3.1: The threshold values for each of the four agent types. Part numbers refer to the part-numbering given in section 3.4

compared to the lower thresholds types than agents with high "density" in their thresholds. This is illustrated in figure 3.6, which shows $T_\theta(s)$ for the thresholds of the *Narrow* (fig. 3.6(a)) and the *Wide* (fig. 3.6(b)) agents in a five-part structure. The figure shows that the *Narrow* agents require much less stimulus to get high values for $T_\theta(s)$ than the *Wide* agents and, more importantly, the difference in needed stimulus for any given value of $T_\theta(s)$ between various values of θ is smaller for the *Narrow* agents, hence making the agents switch behaviours more easily.

3.3.5 Variable Threshold - Spesialisation

The variable threshold agents are initialised using the same values as the *Narrow Fixed Limit* agents. However, contrary to the fixed limit agents, these agents will adjust their threshold according to what job they are currently doing. Each agent continuously adjusts its thresholds as it performs a task, lowering the threshold for this task while raising the thresholds for all other tasks. By employing this strategy, agents will start specialising in the tasks they are performing. Employing such a specialisation-scheme should ensure the completion of each part to a higher degree than with the fixed threshold approach, as the specialised agents will take longer to switch away from a task, ensuring that the last few pieces of each part are also placed.

Table 3.1 shows the threshold values for the three fixed limit agent types along with the initial threshold values for the variable threshold agents.

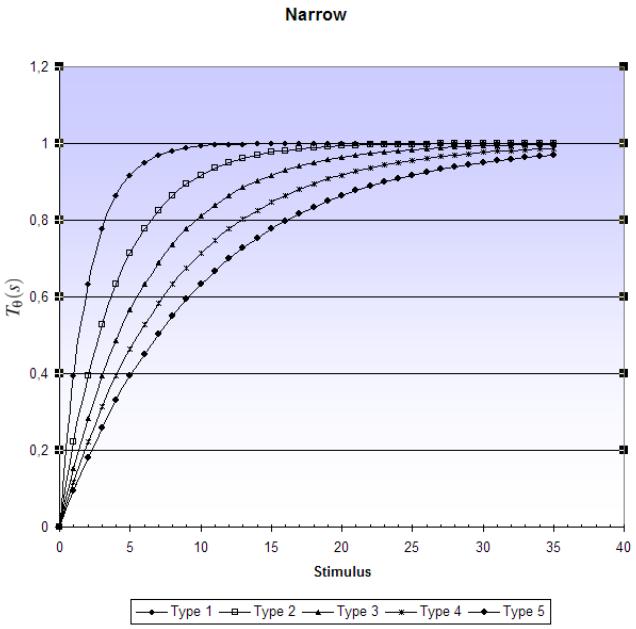
3.3.6 Changing Behaviour

Using the threshold response function described above, each agent can choose whether or not to change behaviour and, if it chooses to change behaviour, it uses its threshold response function to choose its new behaviour. To determine whether an agent should change behaviour, it performs a check vs the value of the response function for its current behaviour. The probability for an agent of type t to change behaviour at any point in time is defined by function 3.2.

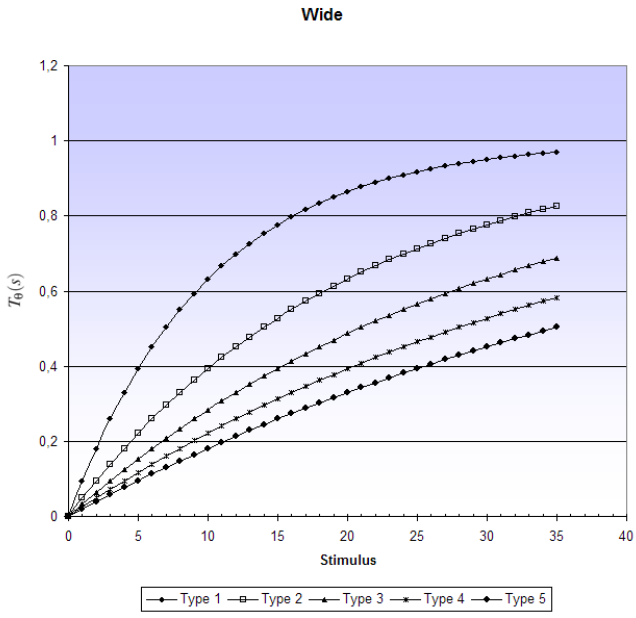
$$p = 1 - T_{\theta_t}(s_t) \tag{3.2}$$

This means that for an agent having $s \gg \theta$ for its current type, p will be close to 0, making it unlikely for the agent to change behaviour, while an agent with $s \ll \theta$ will be likely to change behaviour.

Once an agent has determined that it is to change its type, selecting the new behaviour is done in a roulette-pick, a method often used in evolutionary computing. First, the response function value is calculated for all the possible behaviours, then a new behaviour is picked at random with probabilities weighted so that behaviours with a high response function value are more likely to be chosen.



(a) The *Narrow* agents' threshold functions.



(b) The *Wide* agents' threshold functions.

Figure 3.6: The threshold functions of the *Wide* and *Narrow* agents for each behaviour type for a five-part structure.

3.4 Structure Types

To test the performance of our various agent types we designed multiple structures to be constructed, each adhering to the requirements for a stucture to be called complex. As defined in the preliminary studies and restated with justifications below, the demands set on a complex structure are:

1. It must not be a simple polygon, or describable by a single geometric function.

   Such simple structures do not require or benefit from the use of multiple templates

2. The structure must contain at least one closed-loop wall enclosed by another

   The presence of such closed loops might cause poorly designed systems to prevent the completion of the inner part, creating the demand for a stepwise or multiple template solution. This demand to the structure might also cause it to trap agents inside the structure during construction.

It is important to point out about such complex structures that in most cases, the order in which blocks are placed is not arbitrary, rather the structure can be divided into disjoint parts that need to be constructed in a partially ordered sequence. As an example, take the simple structure of one loop surrounded by another. In this case, agents cannot reach the inner part once the outer part is finished, demanding a strict construction-order of *inner part, then outer part* for this structure. To ensure our structures can be fully constructed we will divide each structure design into such disjoint parts. Furthermore, to enable the agents to search for positions in which to place their blocks, we will place artificial pheromone gradients in the environment.

3.4.1 Templates

To facilitate construction of the various parts of our structure designs, we employ pheromone templates like we did in the preliminary studies. These templates are created by placing pheromone in the area of construction for each of the structure parts and subsequently diffusing this pheromone, creating a pheromone gradient for the agents to follow while constructing the structures. In the following, our different structure designs, along with their pheromone templates, are described.

3.4.2 Wheels and Spokes

The first two structure designs to be tested are modifications of the structure designed for the preliminary studies and consists of square "wheels" connected by sets of "spokes". The two new designs consists of two wheels plus spokes and three wheels plus spokes, making a total of three and five parts respectively. The designs are shown in figure 3.7. Note that spokes on the same level, i.e. connecting the same wheels, are considered a single part of the structure. This type of structure should be reasonably hard, as its many internal compartments might easily trap and/or obstruct agents. To enable construction of this structure type, a pheromone template was placed in the environment, as shown in figure 3.8.

3.4.3 Nested Wheels

The next type of structure is a simpler version of the *Wheels and Spokes* design, consisting of a set of square wheels, but without any connecting spokes. The lower amount of compartments in this kind of structure compared to the *Wheels and Spokes* design, should mean it will be easier to construct as it has a lower chance of trapping agents. The *Nested Wheels* design will be constructed in two versions:

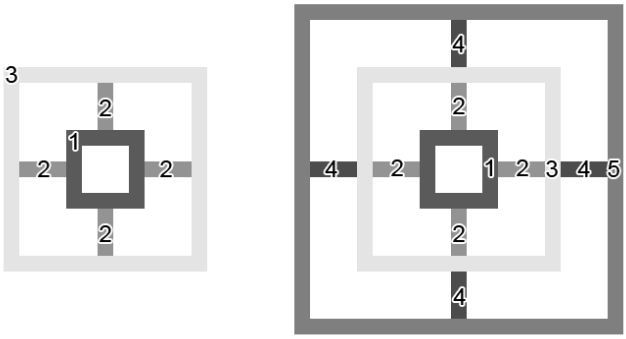


Figure 3.7: The 3-parts (left) and 5-parts (right) versions of the Wheels and Spokes design. Parts are numbered in preferred order of construction.

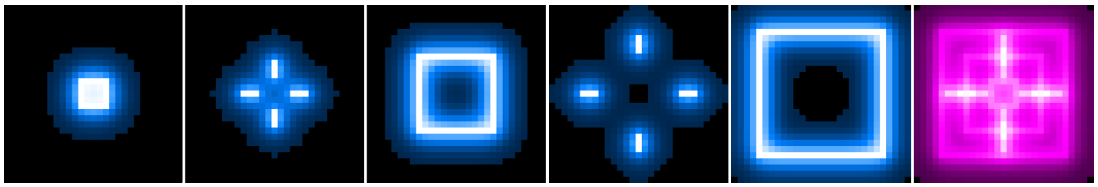


Figure 3.8: The pheromone template for the 5 parts version of the Wheels and Spokes design, showing (from left to right) the templates for parts 1, 2, 3, 4, 5 and the total pheromone.

three wheels and five wheels, as seen in figure 3.9. To facilitate the construction of the *Nested Wheels* design, pheromone templates were placed as shown in figure 3.10

3.4.4 Compact

The last design-type is the Compact design. This design consists, like the *Nested Wheels* design, of a set of square, wheel-like structures, but unlike the *Nested Wheels* design, the *Compact* design has no space between the parts in which the agents are able to travel. This type of design is seen in nature in the form of brood sorting in the ant species *Leptothorax albipennis* and the organisation of honey bee combs. The structure type is a highly researched one, and it has been shown to be quite hard both to complete construction of annular structures like honey combs and to do annular sorting like brood sorting in artificial systems [29, 30, 31].

This tight design, while greatly reducing the chance of trapping agents, should be the hardest of the designs to construct for three reasons. First, the fact that there is no room between the structure parts means premature construction of an outer layer will quickly prevent completion of the inner parts. Also, the fact that in such a structure, each part is very similar in size, the *Complexity Dependent* agents might have problems due to very similar thresholds. The final, and perhaps most severe, challenge posed by this design is that since all the parts are so close to eachother, following the wall of a completed part of the structure to find a free space in which to place a block will inadvertantly have the agent walking on top of where the next part to be built is located, dramatically increasing the stimulus for that part. This might cause the agents to change their behaviour too fast. The Compact design is shown in figure 3.11. The pheromone templates for the *Compact* design are shown in figure 3.12.

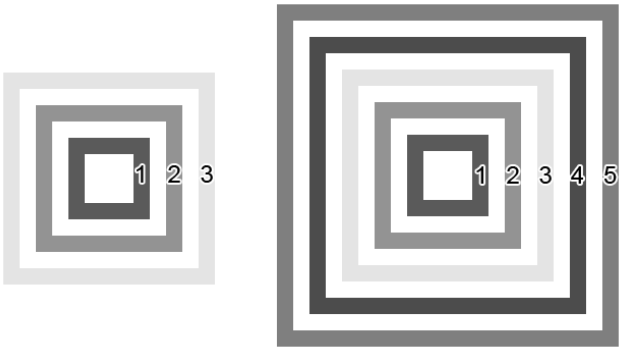


Figure 3.9: The 3-parts (left) and 5-parts (right) versions of the Nested Wheels design. Parts are numbered in preferred order of construction.

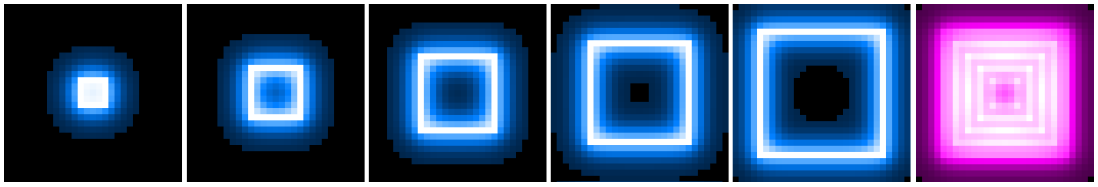


Figure 3.10: The pheromone template for the 5 parts version of the Nested Wheels design, showing (from left to right) the templates for parts 1, 2, 3, 4, 5 and the total pheromone.

Figure 3.11: The Compact design. Parts are numbered in preferred order of construction

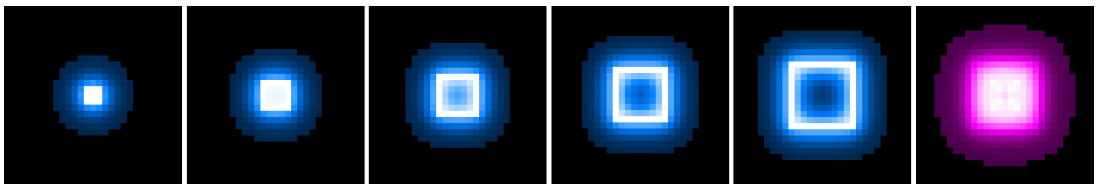


Figure 3.12: The pheromone template for the Compact design, showing (from left to right) the templates for parts 1, 2, 3, 4, 5 and the total pheromone.

3.4.5 Escape Hatches

With the exception of the *Compact* design, all of our structure types have, as one of the main complicating factors, the ability to trap agents, effectively reducing the number of agents that are constructing the structure over time. Outlined in [28] was the possibility of adding "doorways" to remedy this problem. To investigate the effect of this structural change, we will also construct the five-part *Wheels and Spokes* design modified to include such doorways, now renamed "escape hatches". Figure 3.13 shows the modified *Wheels and Spokes* design. The pheromone templates of the *Wheels and Spokes* design are placed as shown in figure 3.14. Note that the only differense in the templates from the *Wheels and Spokes* design is the removal of pheromone in the positions of the escape hatches, thereby preventing the agents from placing blocks in these locations. It is important to note, however, that by adding these escape hatches, the structure is no longer a *complex structure* as defined above, since the structure does no longer consist of any nested loops, thus failing to fulfil one of the two demands set for *complex structures*. The structure is still fairly complex, however, and the effect of the addition of such escape hatches should be investigated.

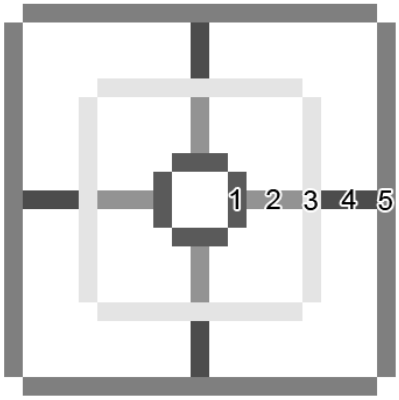


Figure 3.13: The 5-parts Wheels and Spokes design with added escape hatches. Parts are numbered in preferred order of construction.

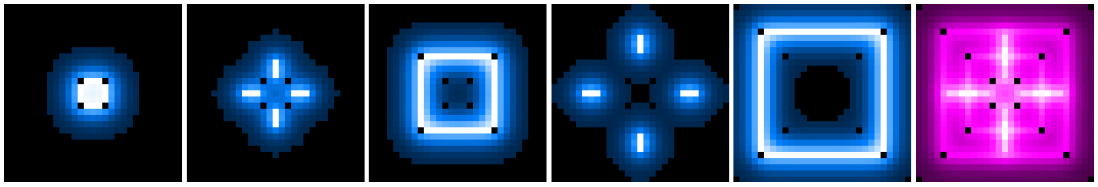


Figure 3.14: The pheromone template for the 5 parts version of the Wheels and Spokes design, showing (from left to right) the templates for parts 1, 2, 3, 4, 5 and the total pheromone.

3.4.6 Random Resource Placement

All of the designs described so far assumes that all available building blocks are initially placed outside the area of the structure. We will also test how a random placement of structure blocks in the environment at start might affect the performance of the system. To do this, we will have all cells in our

environment have a 10% chance of having a block placed on them initially. We will look at the construction of the five-part *Wheels and Spokes* structure with these randomly placed building blocks. One can expect this to raise the initial progress somewhat, as it will increase the probability of a randomly wandering agent finding a block. We will still, however, assume unlimited building resources along the edge of the area, as this thesis focuses on the construction of the structure, not the finding of resources and, as such, the agents have not been made to efficiently search for resources.

3.5 Experimental setup

To test the performance of our agents, we will let each agent type construct each of the structure types described above. In addition to testing all our new agents, we will also test the performance of the *Thresholders* species of the preliminary studies on all of our structure types, making this a benchmark to which we can compare our new agents. In the following we describe the initial values and parameters of our test environment, along with assumptions made about the agents and their simulated environment.

3.5.1 The StarLogo Simulation Environment

To run our simulation, we will use the StarLogo simulation environment. The StarLogo language was developed by Mitchel Resnick and colleagues at MIT. Based on the Logo language, a dialect of LISP, the StarLogo scripting language, used in the StarLogo environment (figure 3.15) allows for programming of the behaviour of distributed agents[1]. StarLogo also enables programming behaviour for the toroidal grid environment in which the agents live, in principle making this environment a cellular automaton in its own right. The environment is mainly used to simulate the behaviour of distributed agent systems, and has native methods for behaviours such as moving and orienting the agents, along with diffusion of variables over the patches of the environment, making it the ideal testbed for our thesis. For an excellent introduction to StarLogo and some interesting experiments done in this environment, see [11].

3.5.2 Initial Conditions

For each of our simulation runs, some parameters will be common for all the runs. These common parameters are listed in table 3.2. The stimulus increase/decrease parameters are the amounts of stimulus added/removed when manipulating an agents stimulus as described in section 3.3.2. In addition to these common parameters, the agents' threshold values are also set according to agent type as shown in table 3.1. The benchmarking agent will be initiated with the parameter values that gave the best results in the preliminary studies. In addition to these parameters, all agents will be initiated with some stimulus for the part labeled "1" in each of the structures to avoid initial type-oscillation as, otherwise, they would not have any stimuli on which to base their behavioural decisions.

3.5.3 Assumptions

Some assumptions are made about the agents and about the simulated world in which they operate. Specifically, the following four assumptions are made:

1. Blocks along the edge of the map are immediately replaced when picked up.

[1] Called "turtles" in the StarLogo terminology.

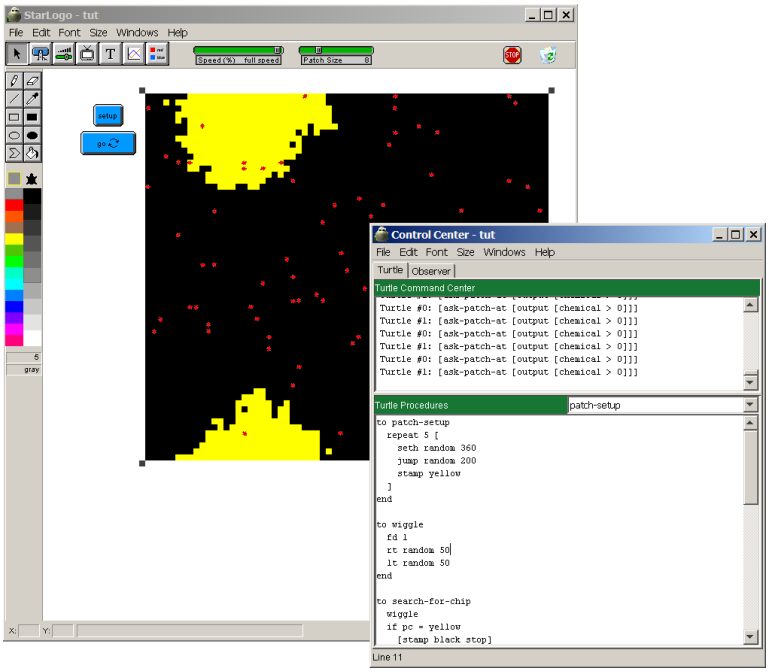


Figure 3.15: The StarLogo simulation environment.

Parameter	Value
World size	29 x 29
Number of agents	50
Number of runs per structure type per agent type	20
Stimulus increase	+1
Stimulus decrease	-2

Table 3.2: The common parameters for all simulations.

This assumption enables us to ignore the problem of searching for blocks for construction, as this is not relevant to our research.

2. Agents cannot pass through walls.

   By denying agents the ability of walking through walls, we create the demand for completing the structure part in the correct order.

3. Agents *can* pass through each other.

   This assumption is made to lessen the complexity of the agents' pathfinding behaviour, as they do not have to find alternate routes to pass other agents.

4. If a test-run lasts for more than 2500 timesteps, the run is assumed failed due to stagnation.

   This stop-condition is mainly a limit designed to enable batch simulation. The value was arrived at by doing test-runs for all the designs and agent types to ensure it gives the agents sufficient time to finish the structure if they are able to do so.

3.6 Key Aspects and Expectations

To ensure success in constructing all of the structure types described in the previous, there are four key aspects our agents need to take into account.

- Stability
  To ensure each part is completed, the agents should not change behaviours too fast. Oscillating agent behaviour would be a sign of low stability.
- Adaptability
  To ensure all parts are in fact constructed, the agents need to adapt to their changing stimuli and change behaviour as the need arises.
- Efficiency
  The agents have to construct the structure in as little time as possible. As the agents are given a time-limit, albeit a rather generous one, they need to switch behaviour fast enough to ensure the whole structure will be built within the given time-frame. A high efficiency will be evident by low construction-completion times.
- Reliability
  For a system like this to be useful, one needs to be confident in its ability to perform the task given. High reliability will ensure high construction success rates.

While the first two aspects, stability and adaptability are innate qualities in each agent design, efficiency and reliability are qualities thata can only be quantified through experiments. One might say that stability and adaptability are the cause, while efficiency and reliability are the effect. Concerning the various agent types designed, we see that they all cater to all of these needs, though some of the agent-types are more focused to some of the needs. The *Wide* agent-type is designed with stability in mind, as the large differense in thresholds for each of the structure-parts should have the agents retaining their behaviour for a long time compared to, say, the *Narrow* agents. The *Narrow* agents, on the other hand, sacrifice stability for efficiency and adaptability as they will likely change behaviour faster, thus potentially finishing the structure in a shorter time. The *Complexity Dependent* agents are a compromise between the two other fixed limit agents, having lower thresholds for the parts of the structure that are small, and which therefore create less stimulus in our model. This approach should

give a good balance between efficiency and stability. The *Variable Limit* agents are designed to focus both on efficiency and reliability. They are initiated in the same way as the *Narrow* agents, making them have high initial efficiency and adaptability. The continuously changing of the agents' threshold limit, however, makes the agent specialise in different parts of the structure, and will make some agents lower their adaptability and efficiency while raising their stability, thereby raising the probability of the part they are currently constructing being completed.

Looking at the structures designs, they all require the agents to show all of the above mentioned aspects. Some of the structures, however, favour various aspects. Specifically, the designs vary in how they emphasise the importance of Stability and Adaptability. Since the *Compact* structure has such a strict requirement for near-serial construction of the structure-parts, this design favours stability, and will most likely punish agents having too high a focus on efficiency, like the *Narrow* agents. On the other hand, the *Nested Wheels* design is much more forgiving in relation to behaviour-switching and allows for high parallelisation of construction, possibly favouring efficiency-focused agent designs. The *Wheels and Spokes* structure does not focus on any specific key aspect. The focus of the structure designs are summarised in table 3.3.

	Stability	**Adaptability**
Wheels and Spokes	Medium	Medium
Nested Wheels	Low	High
Compact	High	Low

Table 3.3: Focus set on different key aspects by the different structure designs.

From these assumptions, we should expect the *Narrow* agents to perform well in relation to construction time in all of the structure designs, but possibly having a lower successrate, especially for the *Compact* design. On the other hand, the *Wide* agents can be expected to have a high reliability in all cases, while taking longer to finish each structure. The *Complexity Dependent* and *Variable Limit* agents should show results somewhere in between the *Wide* and the *Narrow* agents, hopefully exhibiting both high reliability and high efficiency, though the *Variable Limit* agents might be expected to show a higher degree of reliability because of agent specialisation. A summary of these assumptions made of the agents are shown in table 3.4.

	Stability	**Adaptability**	**Efficiency**	**Reliability**
Wide	High	Medium	Low	High
Narrow	Low	High	High	Low
Complexity Dependent	Medium	Medium	Medium	Medium
Variable Limit	Medium	Medium	Medium	High

Table 3.4: Expected performance in relation to the key aspects for each of the agent types.

CHAPTER

4

Results

Look to the ant, thou sluggard! Consider her ways and be wise.

Proverbs 6:6

To evaluate our agent designs, we ran simulations for all agents on all structures as described in the previous chapter. In this chapter, we report the results of all runs and comment on the findings. We start out by defining the metrics with which to evaluate the performance of our agents.

4.1 Performance Metrics

To enable us to evaluate the performance of our agents and to enable us to compare the various designs, we need a set of metrics. The four key aspects described in section 3.6 make a good set of characteristics for comparison so we will base our metrics on these four aspects. Thus, our agents will be evaluated and compared with relation to their *stability, adaptability, efficiency and reliability*.

The latter two of these aspects are easily quantifiable, completion time and success rate being the obvious measurements for efficiency and reliability respectively. The other aspects, namely stability and adaptability are harder to quantify and must be evaluated by looking at how the agent designs perform in construction of the various structure designs. As some of the agents still had some problems finishing the last parts of our structures, we will also state the efficiency measures at the time of 95% completion for all runs.

The unit of measure for efficiency will be blocks placed per timestep (BpT) and the unit for reliability will be the percentage of runs in which the agent succeeded. Our key aspects and the methods and unit of measure for each aspect are listed in table 4.1. All measures and evaluations will be done both over all runs and for each structure design individually.

Aspect	Method	Unit
Stability	Subjective evaluation	N/A
Adaptability	Subjective evaluation	N/A
Efficiency	Total blocks placed / Time spent	Blocks per Timestep (BpT)
Reliability	Success rate	% of runs

Table 4.1: The key aspects of agents and the chosen method of evaluation for each aspect.

4.2 Simulations

In this section, the results of all simulations are reported for each of the agent types. Important characteristics of each agent will also be pointed out. Detailed information about each run can be found in appendix A

4.2.1 The Benchmarking Agents

In addition to the new agent designs, the best agent design of the preliminary studies has been tested on the new structure designs. The results of this agent are provided to have a benchmark for the improved agent designs to be compared to. These agents have a very simple threshold for changing behaviour, deciding to change to constructing the next part of the structure once they have failed a certain amount of times placing blocks for the part they are currently constructing. Below, we report important values and averages from the experimental runs using the old agent designs. The results for each of the runs of the benchmarking agent are reported in table 4.2 and will be reported in more details in the following.

Design		Final BpT	BpT at 95%	Reliability
Wheels and Spokes	3 parts	0.43	0.48	100%
	5 parts	0.26	0.44	50%
Nested Wheels	3 parts	0.28	0.41	90%
	5 parts	N/A	0.24	0%
Compact		0.32	0.18	5%
Escape Hatches		0.30	0.42	85%
Random Blocks		0.28	0.48	50%
Overall		*0.31*	*0.38*	*54%*

Table 4.2: The average efficiency and reliability values for the benchmarking agents.

Wheels and Spokes

The *Wheels and Spokes* design is based on the structure-design of the preliminary studies, in fact, the 3 part design is identical apart from the size of the different parts, which are slightly smaller in the new design. As shown in table 4.2, the old agent design has a 100% successrate, along with high efficiency as should be expected for the 3-part variant of the design, while showing lower values for the 5 part design. The typical end result for a failed run is shown in figure 4.1, showing two blocks missing from the design with the access to these locations blocked by the completed surrounding structure.

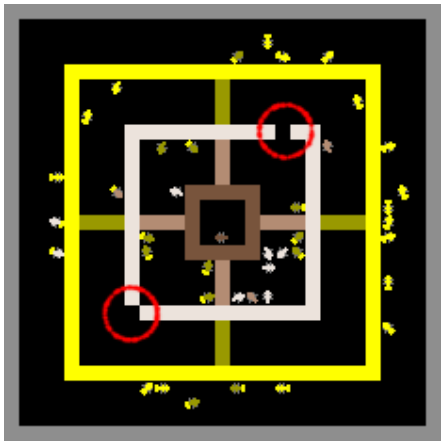


Figure 4.1: A typical failed run for the benchmarking agents. Circles outlining the missing pieces.

Nested Wheels

The nested wheels design created problems for the old agent design, as the space between structure parts create long corridors for the agents to traverse, creating a high rate of block-placing failure for the agents, thus making them change their behaviour, and also causing the agents to be trapped between parts of the structure. Though the agents managed to construct the whole structure in 18 of the 20 runs for the 3 part structure, the 5 part structure proved harder to finish, rendering the agents unable to finish any of the runs. Figure 4.2 shows a typical example of a failed 5 part *Nested Wheels* run, where all agents are either prevented from accessing the area in which construction is needed, or are unable to switch to the correct behaviour.

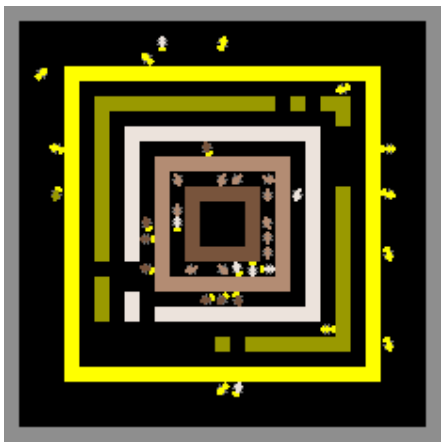


Figure 4.2: An example of a failed *Nested Wheels* run using the old agent design.

Compact

The *Compact* structure design proved a hard one for the old agent type to complete, making them successful in only one of the 20 runs, as reported in table 4.2. The reason for the high rate of failure

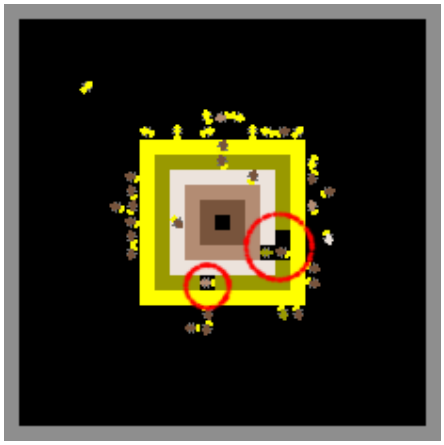


Figure 4.3: The creation of "pockets" while constructing the *Compact* structure.

for this design was the creation of "pockets" in the structure as shown in figure 4.3, caused by agents changing behaviours too fast, and subsequently trapping other agents in such pockets.

Escape Hatches

Adding escape hatches to the structure to be completed was a suggestion for improving performance put forward in the preliminary studies, as it would mean there would always be a path to any cell on the grid. However, as is seen in table 4.2, the old agent design still succeeded in finishing the structure only in 85% of the runs. This failure was caused by these agents' lack of a wall-following behaviour. As we shall discuss in section 4.3, however, the design of the templates might also be a cause for the failures of both the old agent design and the new ones in several cases.

Random Blocks

By placing some blocks randomly in the environment at startup, we expected to see slightly lower completion times, as the early parts of the structure should be completed faster due to the increased probability of finding blocks for the agents. As seen in table 4.2, our expectations seem confirmed, seeing that the reliability is the same as for the normal 5 part *Wheels ans Spokes* runs, while the efficiency is slightly higher.

Overall Performance

Overall, the old agent design shows good results for the 3 part *Wheels and Spokes* structure for which it was originally tuned, while exhibiting low reliability and efficiency for most of the other test-cases. Considering this agent type's adaptability and stability, we can plainly see from the results that it shows high adaptability and stability for the 3 part *Wheels and Spokes* design, while being unable to adapt to the other designs, usually changing behaviours too fast, or causing unability to reach the necessary locations. This performance is expected, as this is the agent design we seek to improve on.

4.2.2 Narrow

The first version of our new agents to be tested is the *Narrow* fixed threshold limit agent. This agent has similar threshold values for the various structure parts, and was expected to display high efficiency while possibly lacking reliability. As expected, the agent performed well with relation to efficiency, while having some poor results concerning reliability, most notably when constructing the *Compact* design. The results of the *Narrow* agents are reported in table 4.3.

Design		**Final BpT**	**BpT at 95%**	**Reliability**
Wheels and Spokes	3 parts	0,51	0.54	100%
	5 parts	0.49	0.54	90%
Nested Wheels	3 parts	0.48	0.51	100%
	5 parts	0.48	0.50	100%
Compact		0.07	0.08	75%
Escape Hatches		0.47	0.51	100%
Random Blocks		0.50	0.57	60%
Overall		*0.43*	*0.46*	*89%*

Table 4.3: The average efficiency and reliability values for the *Narrow* agents.

Wheels and Spokes

As this design focuses neither on relability nor efficiency in the agents, as discussed in section 3.6, it was expected that the *Narrow* agent design would have high efficiency with an average reliability score. Looking at the results reported in table 4.3, we see that our expectations were correct and that while showing very high efficiency results, this agent design suffers from its low focus on reliability by failing 10% of the test runs on the 5 parts variant of the *Wheels and Spokes* design.

Nested Wheels

The nested wheels was expected to put little emphasis on stability, while awarding high adaptability and, indeed, as table 4.3 shows, the *Narrow* agent design, being one of high adaptability and low stability, performs very well, having high values for both efficiency and reliability in both the 3 parts and the 5 parts variants of the structural design.

Compact

In contrast to the *Nested Wheels* design, this design was expected to award stability and punish high adaptability and, again, our expectations proved to be correct. The *Narrow* agent design has a low 75% reliability score and also exhibits low efficiency for this structure, having the lowest score of the new agent designs for both efficiency and reliability. The reason, like with the old design, for the failures of construction is because of the construction of pockets in the structure as shown in figure 4.3.

Escape Hatches

The addition of escape hatches to the *Wheels and Spokes* design effectively removes all focus set by the design on stability in the agents, as it is no longer possible to obstruct the access to any cell in the

simulation environment. Since the *Narrow* agents are highly focused on adaptability, it is no surprise that they perform very well, in fact showing the best results of all the agent types, concerning both efficiency and reliability, in constructing this design.

Random Blocks

By adding random blocks to the environment, we expected to see higher efficiency values due to faster construction of the earlier parts of the structures. As table 4.3 shows, this was a reasonable assumption. More interesting is the rather drastic decrease in reliability by 30% compared to the regular 5-part *Wheels and Spokes* structure. As we shall see below, this decrease in reliability was evident in all of our new designs, and we will discuss the possible reasons for this in section 4.3.

Overall Performance

As expected, this agent design shows high efficiency and low reliability. In fact, the *Narrow* agent design has the highest efficiency values for all structural designs except the *Compact* design, which punishes highly adaptible agents.

4.2.3 Wide

This agent design is mostly identical to the *Narrow* design, only differing in the difference between the thresholds of the various structure parts. We expected this agent to show high reliability with low efficiency due to its high thresholds for the later parts of the structure which should cause it to choose to build these parts only after considerable stimulus. As shown below, however, though it does show high reliability, efficiency is also quite high. The results of the various runs for the *Wide* agent design are shown in table 4.4.

Design		**Final BpT**	**BpT at 95%**	**Reliability**
Wheels and Spokes	3 parts	0.47	0.49	100%
	5 parts	0.45	0.48	100%
Nested Wheels	3 parts	0.43	0.45	100%
	5 parts	0.42	0.48	100%
Compact		0.10	0.11	100%
Escape Hatches		0.40	0.45	100%
Random Blocks		0.45	0.52	75%
Overall		*0.39*	*0.43*	*96%*

Table 4.4: The average efficiency and reliability values for the *Wide* agents.

Wheels and Spokes

As this structure design requires the agents to focus neither on adaptability nor stability to a high degree, it does not punish the *Wide* agent design for its low adaptability or its high stability, thus the agents show both high reliability and quite high efficiency values for both variants of this design.

Nested Wheels

This design rewards adaptability while not setting especially high demands for stability, thus making the *Wide* agent design show only average values, as seen in table 4.4.

Compact

The fact that this structure design punishes high adaptability and demands high stability makes it ideal for the *Wide* agent design and, as is evident from table 4.4, the *Wide* agent design exhibits a 100% reliability along with the second highest efficiency values of all the agents for this structure design.

Escape Hatches

This structure design removes the need for stability, which is the *Wide* agents' main focus. This causes the agents to show no significant difference in the results compared to the regular 5-part *Wheels and Spokes* design.

Random Blocks

Again, we see that the addition of random blocks at startup causes the system to show a significantly lower reliability than normal, though the efficiency is slightly higher than for the regular *Wheels and Spokes* runs. The results, show that the *Wide* agents exhibit in constructing this design, as with most of the other structure types, a slightly higher reliability and slightly lower efficiency than the *Narrow* agents.

Overall Performance

Overall, the *Wide* agent design performs mostly as expected, showing high reliability at the cost of slightly lower efficiency. The efficiency and reliabiliy values for the *Wide* agent design is shown in table 4.4 and shows that, on average, the *Wide* agent design performs very well concerning reliability and also, quite unexpectedly, concerning efficiency.

4.2.4 Complexity Dependent

In contrast to the *Wide* and *Narrow* agents, which set the thresholds using only the prefered order of construction for the various structure parts, the *Complexity Dependent* design also considers the size of the parts when setting their thresholds. This was expected to raise the efficiency somewhat, as it should enable construction of smaller parts at lower levels of stimulus, thereby enabling higher adaptability with relation to these smaller parts. As we shall se below, however, this agent design excels neither in efficiency nor reliability in constructing most of the structure designs. The results for this agent design are shown in table 4.5.

Wheels and Spokes

For this structure type, the *Complexity Dependent* agent design is among the poorest performers of the new designs for both structure variants, even being outperformed by the benchmarking agents for the 3-part variant. This design also fails to complete the 5 parts variant in 10% of the experimental runs.

Design		Final BpT	BpT at 95%	Reliability
Wheels and Spokes	3 parts	0.40	0.43	100%
	5 parts	0.42	0.45	90%
Nested Wheels	3 parts	0.41	0.44	100%
	5 parts	0.42	0.45	95%
Compact		0.12	0.12	100%
Escape Hatches		0.41	0.45	100%
Random Blocks		0.48	0.50	75%
Overall		*0.38*	*0.40*	*94%*

Table 4.5: The average efficiency and reliability values for the *Complexity Dependent* agents.

Nested Wheels

For the *Nested Wheels* design, this design also performs quite poorly, being the only of the new agent designs to fail any of the runs and still showing quite low efficiency values for both variants of the design, as reported in table 4.5.

Compact

Surprisingly, while being a very poor performer for the *Wheels and Spokes* and *Nested Wheels*, the *Complexity Dependent* agent design is actually the best suited of the designs for handling the *Compact* design, exhibiting the highest efficiency of all agents, along with a 100% reliability. This might be due to its high thresholds for each type in this structure design, as we shall discuss in section 4.3.

Escape Hatches

For this design, the *Complexity Dependent* design shows neither very high, nor very low results, having average efficiency values along with a 100% reliability, as do all the new agent designs.

Random Blocks

Once again, we see the reduced reliability caused by random initial block placement, though the *Complexity Dependent* agents, otherwise being poor performers, do actually show efficiency similar to the other new agents while having a 75% reliability, being the same as that of the *Wide* agent design.

Overall Performance

The *Complexity Dependent* agents show only mediocre results in both reliability and efficiency values, being outperformed by the *Wide* and *Narrow* agents in both regards. As shown in table 4.5, the average reliability is at a reasonably high 94%, and this, combined with excellent results for the *Compact* structure design are the *Complexity Dependent* agents' merits.

4.2.5 Variable Limit

The *Variable Limit* agents, while being initiated to have thresholds identical to the *Narrow* agents, has been augmented to allow for dynamic variation of each of the thresholds within agents, causing them to specialise in constructing parts of the structure by lowering the threshold for any part they are currently constructing while raising it for any other parts. We expected this to make the *Variable Limit* agents exhibit high reliability while being initiated to the values of the *Narrow* agents was expected to give high efficiency. As we will see in the following, at least one of these assumptions were correct. Table 4.6 report the results of the *Variable Limit* agents.

Design		**Final BpT**	**BpT at 95%**	**Reliability**
Wheels and Spokes	3 parts	0.49	0.54	100%
	5 parts	0.40	0.45	100%
Nested Wheels	3 parts	0.47	0.50	100%
	5 parts	0.39	0.43	100%
Compact		0.09	0.10	100%
Escape Hatches		0.38	0.43	100%
Random Blocks		0.44	0.51	80%
Overall		*0.38*	*0.42*	*97%*

Table 4.6: The average efficiency and reliability values for the *Variable Limit* agents.

Wheels and Spokes

For this structure design, the *Variable Limit* agents show very good results for the 3-part variant, while showing distinctly low efficiency values for the 5-part variant. The poor efficiency results is one that is recurring for this agent design in all of the 5-part structures and is due to over-specialisation, as we shall discuss in section 4.3. Though this agent designs shows a low efficiency, it has a 100% reliability rating for both of the *Wheels and Spokes* variants.

Nested Wheels

Again, like with the *Wheels and Spokes* design, the efficiency of the *Variable Limit* agents is second only to the *Narrow* agents for the 3-part variant, while being the poorest achiever, efficiency-wise, of all the new agents in constructing the 5-part variant. Table 4.6 shows that, again, this agent design performs well with relation to reliability for both the 3-part and the 5-part variants of the design.

Compact

Like the *Narrow* agents which they resemble initially, the *Variable Limit* agents have a low efficiency rating for the *Compact* structure design, as shown in table 4.6. Contrary to the *Narrow* agents, however, is the *Variable Limit* agents' high reliability rating, matching our expectations for a high reliability, medium efficiency agent design.

Escape Hatches

The *Variable Limit* agents show surprisingly poor performance in constructing the 5-part *Wheels and Spokes* structure with added escape hatches, having the lowest rating of all the new agents and even approaching the benchmarking agents in efficieny. This, as mentioned above, is again related to these agents' over-specialisation problem, as will be discussed in section 4.3.

Random Blocks

Though again, these agents suffer from over-specialisation and, thus, low performance, the *Variable limit* agents have the highest reliability value of all the agents tested in constructing the 5-part *Wheels and Spokes* structure with random initial block placement. The difference in reliability compared to the other designs are small though, and this will be discussed in section 4.3.

Overall Performance

The overall performance of the *Variable Limit* agents is the best of all the agents for reliability, though they suffer greatly in the efficiency rating due to over-spesialisation.

4.3 Analysis and Discussion

To be able to draw any conclusions from our results, as summarised in table 4.7, we need to take a closer look at why the differences in reliability and efficiency occur, and evaluate how large and how important these differences are.

Agent Design	Final BpT	BpT at 95%	Reliability
Benchmark	0.31	0.38	54%
Narrow	**0.43**	**0.46**	89%
Wide	0.39	0.43	96%
Complexity Dependent	0.38	0.40	94%
Variable Limit	0.38	0.42	**97%**

Table 4.7: The overall results for all agent designs. High values in boldface.

One interesting aspect of the results is the very high performance of the benchmarking agents with relation to construction of the 3-part *Wheels and Spokes* structure. This agent, which has only one parameter to adjust its adaptability and stability, a high threshold giving stability, while a low threshold gives adaptability, actually outperforms one of our new agents - the *Complexity Dependent* design. Of course, the *Complexity Dependent* design performs much better than the benchmarking agents for the other structure designs, but these results do suggest that the original agent design can probably be tuned to build any of the other structures with good performance as well. The investigation of such a claim is left for further work however, and we see that the benchmarking agents are vastly outperformed by all the new agents in the other structural designs.

The *Complexity Dependent* agent design, though it showed poor performance with relation to efficiency for the 3-part variants of the structures and had some reliability issues when building the 5-part variants, still performed roughly as expected, showing average results for both reliability and efficiency, as predicted in section 3.6.

The agent design showing the highest efficiency was, as expected, the *Narrow* design, having the highest efficiency of all agents for all structure designs except the *Compact* structure, along with some mediocre reliability results, as predicted in section 3.6. The high efficiency is due to the agent design's high adaptability which causes the agents to change behaviour quickly when the need arises. The high adaptability, however, is also the cause for this design's lower reliability, as it sometimes causes agents to change behaviour too quickly, keeping them from finishing the inner structure parts.

Also expected and predicted in section 3.6 was the *Wide* and the *Variable Limit* agents' high reliability. The *Variable Limit* agents' slightly higher reliability than that of the *Wide* agents (97% vs 96%), while in lines with our expectations of the *Variable Limit* agents having a higher reliability, is inconclusive. With our limited test-data consisting of only 140 runs for each agent type in total, the 1% difference in reliability is due to only 1 run more failing for the *Wide* agents than for the *Variable Limit* agents, this being while constructing the 5-part *Wheels and Spokes* structure with random block placement, a structure for which the reliability was low for all of the agent designs. As the reliability for both of these agent types was 100% for all of the other structure types, the two design must be considered to have equal reliability.

All of our new agents show a reasonably high efficiency for all structure types, all agents designs having an average efficiency between 0.38 and 0.41 blocks placed per time-step. In comparison, the benchmarking agents have an average efficiency of 0.31 blocks/time, placing them 22% lower than the worst performer of our new agents, the *Variable Limit* agents. This high efficiency is caused by our agents' relatively high adaptability through their improved response threshold model. The high adaptability is illustrated by example in figure 4.4, showing the percentage of the agents showing each behaviour, along with the current completion percentage for the corresponding structure part as a function of time for a successful run of the *Narrow* agent species. This figure is representable for all of our new agent types and show quite clearly how the agent population adjusts its distribution of behaviours with relation to the completion of each structure part.

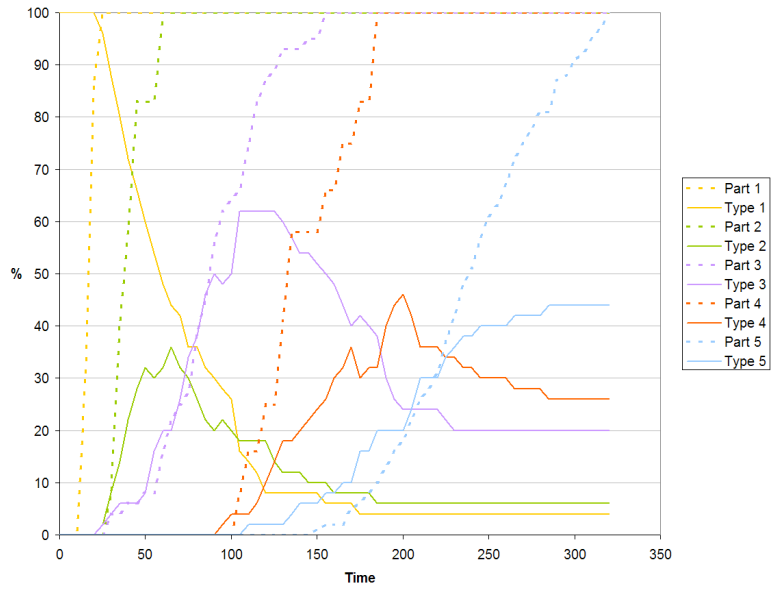


Figure 4.4: The percentage of agents showing each behaviour type and the completion percentage of each structure part as a function of time for a successfull run by *Narrow* agents.

Though all of our new agents show relatively high efficiency, the *Variable Limit* agents are the worst

performers and seem to have a lower efficiency the more parts the structure has. Investigating this, we found that the slow construction by this agent type is caused by the threshold adjustments for the agents in this design. As agents, while constructing a part, will lower the threshold for this part while raising the threshold for all other parts, the latter parts of the structures will be caused to have very high thresholds, making the agents slower in adopting the behaviours for these later parts of the structure. A maxiumum threshold limit might be introduced to remedy this.

Over the course of the experimental runs, only two sets of results were far from what was expected, and we need to see what might have caused these unexpected results. These two abnormalities were the low reliability for all of our agent designs when having random blocks placed in the environment initially and the fact that though the *Complexity Dependent* agents performed very poorly in most of the experimental runs, they performed exeptionally well when constructing the *Compact* structure. We will look at the latter first, as it is easily explained.

As we expected, and this was also confirmed through our experiments, the *Compact* structure design puts heavy focus on reliability while punishing high adaptability. Ideally, the agents should build the whole of each part, then switch to the next part, as this will ensure the completion of the whole structure and avoid the creation of "pockets" as described earlier. The reason, then, for the *Complexity Dependent* agents' success is simply the fact that as this is the only agent design that considers structure-part size when setting its threshold, creating higher thresholds than any of the other agent designs for this structure, and thus being higly stable, which is what is needed for the construction of this structure type.

The second abnormality, the low reliability of all our new agents in constructing the 5-part *Wheels and Spokes* structure with random blocks placement initially is a problem that is harder to pinpoint. One likely reason for this lower reliability might be that the obstruction the random blocks create force agents to take detours to arrive at their goal locations and in so doing expose the agents to higher stimuli for other structure parts than they would be exposed to in a "clean" environment. Doing some test runs monitoring the stimulus of the agents, however, show no abnormal stimulus values for the agents. Another explanation is that the randomly placed blocks raised the probability of blocking off parts of the structure, as less blocks are required to finished some parts of the structure. To prove or disprove this claim, further examinations is needed. We leave this as further work.

Though the *Narrow* agents did show the highest efficiency values of all the agents, taking a look at the runtimes for all agents over all runs, as shown in figure 4.5, we see that the difference in runtimes is rather small, all agents having a median runtime of about 300-400 timesteps over all runs with similiar standard deviations. As all agents, even our benchmarking agents from the preliminary studies have such similar completion times, reliability is really the most interesting factor for our agents.

As we see from figure 4.6, showing the reliability of all agents, the benchmarking agents are clearly outperformed by our new agents, so our goal of improving on the original design has definitively been achieved. Looking at the reliability of our new agents, we see that the *Narrow* agent design is clearly outperformed by the other designs, while the *Wide* and the *Variable Limit* agents top the ranking. As discussed in above, the difference between these two designs relating to reliability is not large enough to be decisive on our limited set of test-data, so they must both be considered to have identical reliability. If one looks at the efficiency of the two, however, we see that the *Wide* design has less spread in the values of its finishing times, suggesting that this agent design is more reliable with relation to expected efficiency. The fact that the *Wide* agent design performs at least as well as our *Variable Limit* agents suggests that the possibility of specialisation of the agents that the *Variable Limit* design allows for is not needed in our setting, and, as such, the simpler, fixed limit design is sufficient for construction of complex structures as defined in this thesis. Some test runs using threshold values with a much wider range than the *Wide* agents improved neither efficiency nor reliability, though neither did it lower any of these values significantly, suggesting that having thresholds set too wide might be preferable to having too narrow thresholds. Though these results suggest that the *Wide* design is a better option than the *Variable Limit* design in our setting, the *Variable Limit* agents have one major advantage over the *Wide*

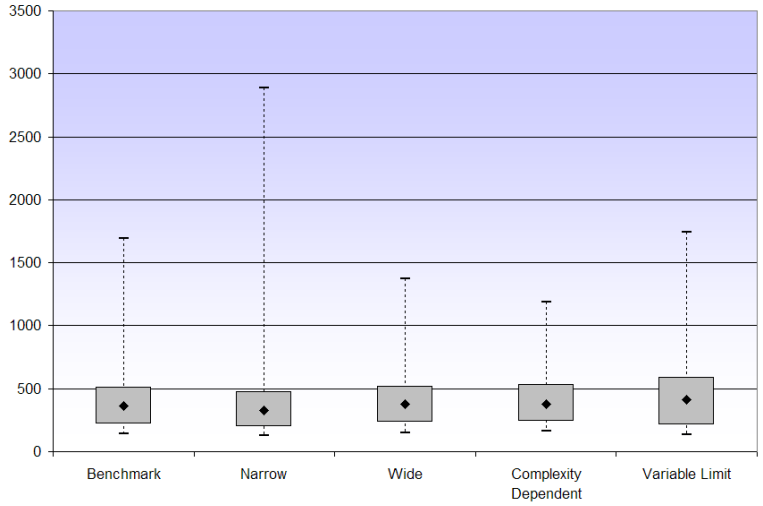


Figure 4.5: The runtimes for all agents on all designs.

agents: while the fixed limit *Narrow* and *Wide* agents are really only the same agents with different threshold values, the *Variable Limit* agents do not rely on the initial threshold values to be successful, as the agents adjust their own threshold. This means that in construction of an unknown structure, while the fixed limit designs rely on complexity analysis of the structure for the thresholds to be set correctly, the *Variable Limit* agents can be placed in the environment directly and relied upon to adjust their own thresholds to correct values.

One problem concerning our template design was uncovered in the course of running the experiments. To ensure the stimulus for the spokes part of the structure would be high enough, the decision was made to make the spokes at the same level of our *Wheels and Spokes* structure share a pheromone template, as illustrated in figure 3.8. As the agents follow the pheromone gradient by going "uphill", this caused the agents to be unable to finish the spokes on some occasions, as all spokes-building agents were at the location of finished spokes, unable to find the unfinished spoke(s) as their rule-set does not allow them to go downhill in the pheromone gradient to reach the other spokes. Investigating our results, we found that out of all the failed runs on structures that contained the spokes as a structural part, 93% of the failed runs failed as a result of the agents not being able to complete one of the spokes parts of the structure. Seeing these results, it is probable that by creating a single pheromone template for each of the spokes, we could raise the reliability of our agents considerably when constructing the *Wheels and Spokes* structure variants.

Overall, though all our new agents are significantly better than the benchmarking agents from the preliminary studies, comparing our new agents, they show similar performance with relation to efficiency, while reliability varies to a larger degree between agent designs.

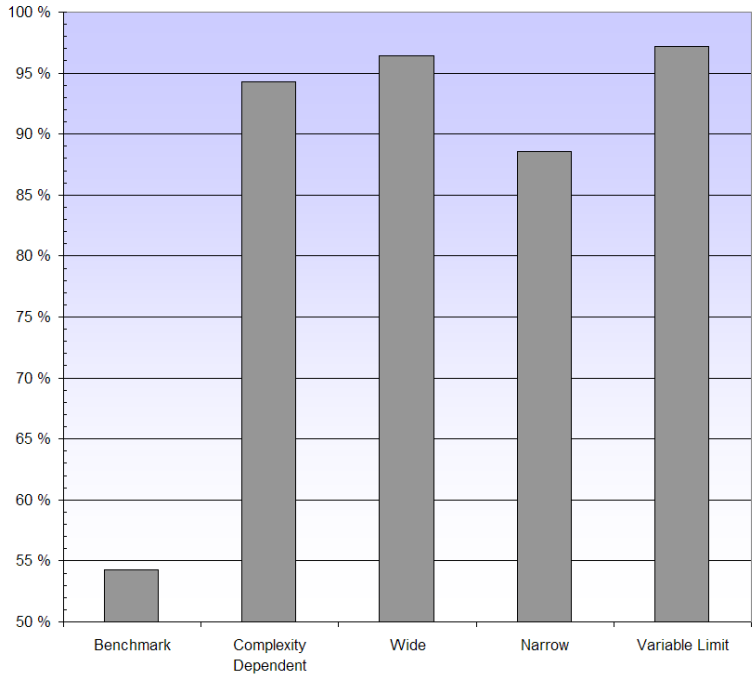


Figure 4.6: The reliability of each agent design for all structure designs

CHAPTER

5

Conclusions and Further Work

I approve of theorizing if it lays its foundation in incident, and deduces its conclusions in accordance with phenomena.

Hippocrates

In this chapter, we will draw conclusions based on our experimental results. Furthermore, we will consider what contribution we have made to our field of study and make some suggestions for further work in the field of distributed construction based on our work.

5.1 Conclusion

In this thesis, we have tested how the parameters of distributed agents can be set dynamically and automatically in order to enable them to construct complex structures. To facilitate these experiments, we have defined a set of criteria for complex structures, along with designing structures to fulfill these criteria. We have also designed a set of agents by first investigating basic agent designs based on job allocation methods found in nature.

After finding a well-performing agent design we set out to improve this design and to investigate how to set the agents' parameters dynamically and automatically to enable them to construct our various complex structures. The investigation of such methods is of practical interest as coordination of tasks for distributed agents lends itself to several problems, among them distributed robotics, distributed data-handling in computer systems and nano-technology. Earlier work in this field has mostly been limited to construction of simple structures [15, 16, 17] and investigation into how certain rules may lead to certain structures [32, 33]. By employing artificial pheromone templates along with job allocation methods observed in nature, we have achieved coordination of a large group of agents without the use of mediating agents or inter-agent communication. By drawing on job allocation methods observed in social insects, we have designed basic rule-following agents able to be placed in an environment and construct large, complex structures without any need for information about the environment in advance.

In our preliminary studies, we arrived at a basic threshold-based agent that was able to construct a complex structure without relying on any other factors than the structure itself, letting us ignore the environment in which it operated. In this thesis we then seeked to improve on the performance of this agent, along with investigating how one could design such an agent in such a way that one could set its parameters in advance, having only minimal knowledge about the structure to be built and have a group system of such distributed agents reliably and efficiently construct the given structure. To improve the performance of our agents we gave them basic wall-following behaviour and also gave them an advanced response threshold model, based on work by Bonabeau et al., to control job allocation.

In the experimental runs for our agents, we found that, for agents with a fixed threshold for each type, having a large difference in threshold values gave a high reliability for our agents, while a smaller difference gave higher efficiency. This resulted in our *Narrow* agents performing very well with relation to efficiency, while failing more runs than any of our other agent designs. We found that the *Wide* agent design gave high reliability and also reasonably high efficiency, while our two more advanced threshold-setting methods, used in the *Complexity Dependent* and *Variable Limit* agents, did nothing to improve efficiency, though the reliability of the *Variable Limit* agents was at least as high as that of the *Wide* agents.

Our results show that in the construction of complex structure, it is possible to employ fairly simple agents, relying neither on direct, inter-agent communication, nor a coordinating or mediating agent, to reliably finish structures of variable size and complexity by relying on job allocation methods found in nature. Furthermore, our results show that further enhancing the agents by allowing for specialisation does not necessarily improve performance in this settings, though it *will* lower the amount of analysis one has to perform before deploying the agents to the environment. This, in our opinion, makes the *Variable Limit* agents the obvious first choice of our designed agents for distributed construction, as it shows excellent reliability, quite high efficiency and does not need any adjustment prior to deployment.

Although this thesis is concerned with distributed construction of complex structures, some of our results lend themselves to other areas of swarm intelligence. Our main goal for this thesis, the automatic configuration of agent parameters is an important in most applications of swarm intelligence and, in deed, most computer systems, as nearly all systems need some form parameter "tweaking" in order to function well. By employing internal stimuli and thresholds, we have shown that we can, to some degree, ignore a lot of the surrounding environment when setting our parameters. Furthermore, the dynamic modification of the internal thresholds enable the agents to specialise over time. This specialisation would be interesting to employ in swarm based systems performing sorting of data, as the principle of locality might make specialising agents an effective approach, as related data tends to be placed close together.

Another advantage of dynamic threshold modification, as employed by our *Variable Limit* agents, is the possibility of creating a set of homogeneous agents, letting them "evolve" into heterogeneous agents over time. This would make swarm intelligence systems easier to use, as one would not need to design more than one type of agent.

One issue concerning our system and one which is certainly an issue for all swarm intelligence systems, is the issue of ensuring reliability. By using distributed, non-communicating agents, we create high parallellability and unlimited scalability, but we also make it very hard to ensure perfect reliability, as these kinds of systems, by their non-deterministic nature, are impossible to predict on a low level. This means that if one is to design a guaranteed reliable system based on swarm intelligence, one might need a method, either centralised or distributed, for monitoring and coordinating the progress of the system. Drawing on techniques from the field of distributed decision making might be a good choice for ensuring reliability in swarm intelligence systems.

In conclusion, we have shown that distributed agents can be relied upon to construct complex structures, and that relying on variable internal stimulus response thresholds is a viable way of providing adaptive

adjustment of parameters for such multi-agent systems. On a more general note, our results show that swarm intelligence systems can be relied upon to perform advanced tasks both efficiently and with high reliability. In so doing, this paper joins a long line of papers proving that looking to nature for inspiration, relying on millions of years of careful adaptation to common problems, is not just one way to go, but might be *the* way to go in a multitude of situations.

5.2 Further Work

Though this thesis goes a long way in finding a method of designing agents for collective construction, some problems are highlighted in the course of our experiments, while other problems are deliberatly overlooked to simplify the experiments. For a system of distributed agents to be a viable option for construction these problems will have to be dealt with. We outline these problems, along with possible extensions of our work below.

A problem that was discovered in the course of our work was, as described in section 4.3, the problems caused by pheromone templates consisting of several, disjoint maximum locations, making construction of the parts they represented harder for the agents. Though, in our case, the problem could be remedied simply by dividing the template into several smaller templates, an important question remains: what is the optimal template design for a given structure? This, along with the method of representation and placement of pheromone gradients is an important issue to be handled.

Another problem evident in our results was the over-specialisation by the *Variable Limit* agents, which causes them to have lower efficiency the more parts are added to the structure. Investigations into how this effect can be limited or avoided would be of great use in increasing the performance of this agent design.

Interesting extensions to our experiments might be the extension into constructing structures in 3D space, along with performing the experiments using physically manifestations of our agents in the form of simple robots, like the ones designed in the *Swarm Bots* project directed by Dr. Marco Dorigo, to further validate our findings.

BIBLIOGRAPHY

[1] B. D., "The mrna assembly line: transcription and processing machines in the same factory," *Current Opinion in Cell Biology*, vol. 14, pp. 336–342, June 2002.

[2] J. J. O. H. Van Dyke Parunak, Sven A. Brueckner, "Swarming coordination of multiples uavs for collaborative sensing," *2nd AIAA "Unmanned Unlimited" Conf. and Workshop and Exhibit*, Sept. 2003.

[3] P. P. C. S. Nandi, B. K. Kar, "Theory and applications of cellular automata in cryptography," *IEEE Transactions on Computers*, vol. 43, pp. 1346 – 1357, 1994.

[4] "Wikipedia article: Centralised traffic control." http://en.wikipedia.org/wiki/Centralised_Traffic_-Control.

[5] D. Medhi and K. Ramasamy, *Network Routing: Algorithms, Protocols, and Architectures*. Morgan Kaufmann, 2007.

[6] Y. Shoham and K. Leyton-Brown, *Multi Agent Systems*. In Press, 2005.

[7] M. Dorigo and T. Stützle, *Ant Colony Optimization*. The MIT Press, 2004.

[8] "Wikipedia article: Ant colony optimization." http://en.wikipedia.org/wiki/Ant_colony_optimization.

[9] M. D. E. Bonabeau and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[10] S. Johnson, *Emergence*. Penguin Books, 2001.

[11] M. Resnick, *Turtles, Termites, and Traffic Jams*. The MIT Press, 1997.

[12] "Wikipedia article: Slime mould." http://en.wikipedia.org/wiki/Slime_mold.

[13] "Websters online dictionary." http://www.websters-online-dictionary.org.

[14] "Wikipedia article: Stigmergy." http://en.wikipedia.org/wiki/Stigmergy.

[15] R. L. Stewart and R. A. Russel, "Emergent structures built by a minimalist autonomous robot using a swarm-inspired template mechanism," *Proceedings of The First Australian Conference on Artificial Life*, 2003.

[16] R. L. Stewart and R. A. Russel, "A generalised technique for building 2d structures with robot swarms," *Proceedings of The Second Australian Conference on Artificial Life*, 2006.

[17] R. L. Stewart and R. A. Russel, "Building a loose wall structure with a robotic swarm using a spatio-temporal varying template," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[18] C. Anderson and D. W. McShea, "Individual versus social complexity, with particular reference to ant colonies," *Biol. Rev. (2001)*, pp. 211–237, 2000.

[19] E. O. Wilson, "The relation between caste ratios and division of labor in the ant genus pheidole (hymenoptera: Formicidae)," *Behavioral Ecology and Sociobiology*, pp. 89–98, 1984.

[20] T. De Wolf, L. Jaco, T. Holvoet, and E. Steegmans, "A Nested Layered Threshold Model for Dynamic Task Allocation," in *LNCS 2463, Ant Algorithms*, Lecture Notes in Computer Science, pp. 290–291, 2002. http://iridia.ulb.ac.be/˜ants/ants2002/.

[21] E. B. et al., "Fixed response thresholds and the regulation of division of labor in insect societies," *Bulletin of Mathematical Biology*, pp. 753–807, 1998.

[22] J.-L. D. Eric Bonabeau, Guy Theraulaz, "Quantitative study of the fixed threshold model for the regulation of division of labour in insect societies," *Proceedings: Biological Sciences*, vol. 263, no. 1376, pp. 1565–1569, 1996.

[23] E. Bonabeau, G. Theraulaz, and F. cois Cogne, "The design of complex architectures by simple agents," Working Papers 98-01-005, Santa Fe Institute, Jan. 1998. available at http://ideas.repec.org/p/wop/safiwp/98-01-005.html.

[24] L. M. G. Marco Dorigo, Gianni Di Caro, "Ant algorithms for discrete optimization," *Artificial Life*, vol. Volume 5, Number 2, 1999.

[25] E. Bonabeau and G. Théraulaz, "Swarm smarts," vol. 282, pp. 72–79, Mar. 2000.

[26] J. Handl, J. Knowles, and M. Dorigo, "Ant-based clustering and topographic mapping," *Artif. Life*, vol. 12, no. 1, pp. 35–61, 2006.

[27] "Swarm-robotics.org." http://www.swarm-robotics.org/.

[28] J. Braseth, "The effect of complex structures in collective construction," 2006.

[29] A. B. S.-F. Matt Wilson, Chris Melhuish and S. Scholes, "Algorithms for building annular structures with minimalist robots inspired by brood sorting in ant colonies," *Autonomous Robots*, vol. 17, pp. 115–136, Sept. 2004.

[30] V. Hartmann, "Evolving agent swarms for clustering and sorting," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, (New York, NY, USA), pp. 217–224, ACM Press, 2005.

[31] A. H. Vik, *Advances in Artificial Life*, ch. Evolving Annular Sorting in Ant-Like Agents, pp. 594–603. Springer Berlin / Heidelberg, 2005.

[32] G. Theraulaz and E. Bonabeau, "Coordination in Distributed Building," *Science*, vol. 269, pp. 686–688, Aug. 1995.

[33] G. Theraulaz and E. Bonabeau, "Modelling the collective building of complex architectures in social insects with lattice swarms," *Journal of Theoretical Biology*, vol. 177, pp. 381–400, Dec. 1995.

APPENDIX

A

Simulation results

(a) 3-part *Wheels and Spokes*

Run	Completion Time
1	160
2	185
3	210
4	215
5	145
6	185
7	160
8	175
9	155
10	155
11	180
12	175
13	180
14	150
15	165
16	165
17	220
18	155
19	170
20	195

(b) 5-part *Wheels and Spokes*

Run	Completion Times
1	99%
2	1695
3	98%
4	99%
5	98%
6	475
7	445
8	505
9	98%
10	390
11	480
12	99%
13	900
14	99%
15	670
16	99%
17	99%
18	515
19	98%
20	470

(c) 3-part *Nested Wheels*

Run	Completion Times
1	320
2	325
3	260
4	235
5	420
6	265
7	250
8	275
9	625
10	365
11	305
12	98%
13	375
14	255
15	265
16	97%
17	320
18	275
19	780
20	330

(d) 5-part *Nested Wheels*

Run	Completion Times
1	95%
2	97%
3	91%
4	94%
5	91%
6	89%
7	94%
8	90%
9	93%
10	97%
11	95%
12	94%
13	94%
14	94%
15	91%
16	96%
17	91%
18	94%
19	95%
20	87%

(e) *Compact*

Run	Completion Times
1	**375**
2	**96%**
3	**98%**
4	**98%**
5	**95%**
6	**97%**
7	**99%**
8	**95%**
9	**97%**
10	**95%**
11	**97%**
12	**98%**
13	**95%**
14	**95%**
15	**98%**
16	**95%**
17	**96%**
18	**98%**
19	**97%**
20	**96%**

(f) *Escape Hatches*

Run	Completion Times
1	515
2	360
3	710
4	695
5	380
6	500
7	99%
8	535
9	885
10	660
11	335
12	350
13	410
14	550
15	500
16	600
17	370
18	99%
19	99%
20	605

(g) *Random Blocks*

Run	Completion Times
1	665
2	99%
3	99%
4	705
5	575
6	99%
7	360
8	98%
9	99%
10	445
11	99%
12	510
13	99%
14	675
15	465
16	1240
17	99%
18	98%
19	99%
20	320

Table A.1: The completion times of the *Narrow* agents for all runs. Percentages show the percentage of the structure completed for non-successfull runs.

(a) 3-part *Wheels and Spokes*

Run	Completion Times
1	175
2	170
3	190
4	185
5	210
6	195
7	180
8	210
9	190
10	200
11	170
12	165
13	215
14	180
15	185
16	200
17	210
18	190
19	195
20	190

(b) 5-part *Wheels and Spokes*

Run	Completion Times
1	415
2	400
3	410
4	380
5	395
6	99%
7	405
8	420
9	415
10	450
11	380
12	385
13	380
14	425
15	380
16	365
17	380
18	440
19	345
20	99%

(c) 3-part *Nested Wheels*

Run	Completion Times
1	215
2	230
3	255
4	235
5	225
6	215
7	265
8	220
9	220
10	240
11	235
12	255
13	250
14	235
15	205
16	210
17	245
18	250
19	240
20	220

(d) 5-part *Nested Wheels*

Run	Finish Times
1	520
2	645
3	550
4	540
5	510
6	540
7	625
8	530
9	480
10	555
11	580
12	99%
13	630
14	535
15	550
16	520
17	610
18	620
19	610
20	525

(e) *Compact*

Run	Completion Times
1	1145
2	935
3	880
4	1065
5	895
6	945
7	855
8	1120
9	995
10	1065
11	975
12	1100
13	1010
14	1130
15	1065
16	1150
17	880
18	955
19	1115
20	1185

(f) *Escape Hatches*

Run	Completion Times
1	410
2	365
3	405
4	385
5	355
6	370
7	405
8	400
9	360
10	375
11	405
12	370
13	435
14	365
15	360
16	370
17	415
18	370
19	360
20	325

(g) *Random Blocks*

Run	Completion Times
1	375
2	345
3	355
4	98%
5	335
6	305
7	99%
8	325
9	98%
10	370
11	370
12	310
13	410
14	315
15	99%
16	415
17	335
18	340
19	355
20	99%

Table A.2: The completion times of the *Complexity Dependent* agents for all runs. Percentages show the percentage of the structure completed for non-successfull runs.

(a) 3-part *Wheels and Spokes*

Run	Completion Times
1	160
2	175
3	150
4	160
5	155
6	155
7	160
8	175
9	160
10	165
11	160
12	180
13	150
14	155
15	180
16	160
17	150
18	175
19	155
20	165

(b) 5-part *Wheels and Spokes*

Run	Completion Times
1	350
2	405
3	330
4	390
5	390
6	350
7	415
8	400
9	365
10	375
11	375
12	350
13	360
14	380
15	385
16	395
17	345
18	365
19	400
20	415

(c) 3-part *Nested Wheels*

Run	Completion Times
1	200
2	215
3	200
4	240
5	220
6	235
7	250
8	205
9	230
10	225
11	185
12	215
13	215
14	240
15	220
16	230
17	225
18	220
19	240
20	255

(d) 5-part *Nested Wheels*

Run	Completion Times
1	540
2	560
3	595
4	525
5	510
6	535
7	1370
8	430
9	550
10	550
11	480
12	585
13	565
14	540
15	580
16	540
17	505
18	475
19	510
20	490

(e) *Compact*

Run	Completion Times
1	1115
2	1175
3	1080
4	1080
5	1210
6	1350
7	1060
8	1140
9	1145
10	1185
11	1270
12	1345
13	1130
14	1135
15	1140
16	1105
17	1080
18	1195
19	1075
20	1080

(f) *Escape Hatches*

Run	Completion Times
1	430
2	365
3	350
4	470
5	360
6	335
7	305
8	395
9	385
10	405
11	380
12	395
13	360
14	385
15	350
16	420
17	440
18	555
19	320
20	360

(g) *Random Blocks*

Run	Completion Times
1	345
2	99%
3	325
4	440
5	99%
6	350
7	99%
8	355
9	405
10	99%
11	320
12	385
13	400
14	370
15	500
16	340
17	360
18	330
19	340
20	99%

Table A.3: The completion times of the *Wide* agents for all runs. Percentages show the percentage of the structure completed for non-successfull runs.

(a) 3-part *Wheels and Spokes*

Run	Completion Times
1	170
2	145
3	150
4	165
5	130
6	150
7	170
8	150
9	145
10	155
11	155
12	145
13	170
14	145
15	135
16	145
17	155
18	145
19	150
20	130

(b) 5-part *Wheels and Spokes*

Run	Completion Times
1	320
2	99%
3	345
4	325
5	305
6	370
7	2890
8	99%
9	335
10	320
11	305
12	99%
13	305
14	350
15	380
16	385
17	345
18	335
19	305
20	315

(c) 3-part *Nested Wheels*

Run	Completion Times
1	205
2	195
3	210
4	190
5	200
6	185
7	215
8	215
9	185
10	200
11	195
12	195
13	175
14	195
15	205
16	225
17	200
18	205
19	240
20	195

(d) 5-part *Nested Wheels*

Run	Completion Times
1	550
2	480
3	600
4	485
5	530
6	465
7	490
8	500
9	550
10	515
11	525
12	520
13	505
14	465
15	475
16	455
17	540
18	470
19	480
20	440

(e) *Compact*

Run	Completion Times
1	1090
2	2755
3	2210
4	97%
5	1595
6	2135
7	66%
8	99%
9	1530
10	2470
11	99%
12	1095
13	995
14	1175
15	98%
16	2155
17	1320
18	2690
19	1595
20	1275

(f) *Escape Hatches*

Run	Completion Times
1	290
2	295
3	360
4	325
5	310
6	325
7	310
8	305
9	365
10	350
11	325
12	335
13	330
14	310
15	300
16	340
17	385
18	430
19	290
20	335

(g) *Random Blocks*

Run	Completion Times
1	290
2	99%
3	325
4	510
5	325
6	99%
7	275
8	99%
9	97%
10	350
11	285
12	325
13	98%
14	99%
15	320
16	330
17	99%
18	365
19	320
20	99%

Table A.4: The completion times of the *Narrow* agents for all runs. Percentages show the percentage of the structure completed for non-successfull runs.

(a) 3-part *Wheels and Spokes*

Run	Completion Times
1	160
2	145
3	155
4	170
5	185
6	135
7	140
8	200
9	135
10	145
11	145
12	135
13	160
14	175
15	145
16	135
17	175
18	150
19	135
20	150

(b) 5-part *Wheels and Spokes*

Run	Completion Times
1	460
2	450
3	450
4	400
5	380
6	365
7	425
8	500
9	435
10	430
11	385
12	440
13	345
14	415
15	435
16	395
17	425
18	415
19	410
20	505

(c) 3-part *Nested Wheels*

Run	Completion Times
1	220
2	200
3	190
4	245
5	180
6	225
7	220
8	170
9	205
10	200
11	230
12	215
13	170
14	200
15	195
16	180
17	215
18	210
19	210
20	205

(d) 5-part *Nested Wheels*

Run	Completion Times
1	630
2	650
3	590
4	615
5	665
6	665
7	665
8	625
9	500
10	610
11	550
12	605
13	610
14	700
15	675
16	625
17	580
18	570
19	560
20	495

(e) *Compact*

Run	Completion Times
1	1260
2	1480
3	1170
4	1275
5	1300
6	1160
7	1395
8	1555
9	1225
10	1215
11	1220
12	1215
13	1185
14	1700
15	1195
16	1305
17	1185
18	1290
19	1745
20	1185

(f) *Escape Hatches*

Run	Completion Times
1	410
2	385
3	445
4	390
5	435
6	615
7	445
8	415
9	395
10	350
11	425
12	370
13	410
14	380
15	365
16	370
17	345
18	425
19	400
20	395

(g) *Random Blocks*

Run	Completion Times
1	99%
2	440
3	355
4	405
5	405
6	360
7	99%
8	415
9	99%
10	390
11	410
12	325
13	315
14	320
15	430
16	360
17	375
18	99%
19	435
20	425

Table A.5: The completion times of the *Variable Limit* agents for all runs. Percentages show the percentage of the structure completed for non-successfull runs.