Chapter 8: Experiments in the Simulated Robot Environment

The simulation and experiments described in chapter 7 were done with a high level of abstraction and a more realistic simulator was developed for the purpose of this thesis. This chapter presents a description of the simulator and the experiments conducted using the simulator. Section 8.1 summarises the main purpose of the simulator, while section 8.2 overviews the main components of the simulator. The simulation set-up, including a description of the robot population and environments used is given in section 8.3. Section 8.4 presents the experimental results.

8.1 Introduction

For the purpose of further experiments into the applicability of social networks as coordination tools in multi-robot teams, robot simulator software was developed. The main purpose of the simulator is to visualise the behaviour and movement of multi-robot teams during task execution. The robot simulator software itself is not a truly realistic simulation of the real-world, since many simplifications were made. A realistic robot simulation was considered at a certain stage (i.e. Webots [184]), but the limitations on social interaction models in readily availably robot simulations have necessitated the development of a simulator that focuses on multi-robot teams, albeit in a less realistic simulated environment.

8.2 Robot Simulator Overview

The robot simulator was developed in C++ in a Windows ™ environment. The robot simulator consists of the following components:

- The robot definition component, which encapsulates robots' behaviours.
- The society component, which maintains social links between robots.
- The environment component, which provides interaction with the simulated environment.
- The display component, which visualises task executions.

Each of the components will be described in more detail in the following sections. The main loop of the robot simulator is illustrated in algorithm 10.

```
For all Robots in team
   If Task not accomplished
      SetSensorReadings(Robot, Environment)
      Execute(Robot)
      ValidateAction(Robot, Environment)
      Display(Robot)
   EndIf
EndFor
```

Algorithm 10. The main loop of robot simulator

8.2.1 Robot Definitions Component

In order to keep the realism level high, robot-related functionality is kept and implemented independently in the robot definitions component. This facilitates a potential migration from the simulated robot to a real robot platform. The simulated robot architecture is based on a simplified INDABA agent architecture (refer to chapter 5), consisting of two layers:

- A controller layer that implements GoTo, AvoidObstacle, DetectObject and GrabObject behaviours.
- A combined sequencer and deliberator layer.

Such simplifications are justified by the fact that the task was one of the benchmark tasks for robotic teams: a simple foraging task. It was deemed to be unnecessary to implement a full inference engine based deliberator layer. Similarly, the interaction between the agents is handled by the society component of the robot simulator. Implementation of the interaction layer was therefore not deemed necessary. These simplifications should not impact on the "realism" factor in any manner.

The robot component receives input from the environment by reading sensor inputs from the environment. Based on the sensor inputs and the progress of task execution,

the combined sequencer and deliberator layer enables or disables behaviours in the controller layer. The desired robot actions are then sent to the environment component that evaluates their validity.

8.2.2 Display Component

The display component is used to monitor the execution of the allocated task. Minimum effort was spent on this component as it is of little relevance to the outcome of the experiments. A typical screenshot is presented in figure 35. White rectangles represent obstacles, small black rectangles represent "food", while the larger dark grey area represents "rough terrain". The symbol referred to by the white arrow on figure 35 indicates a robot.

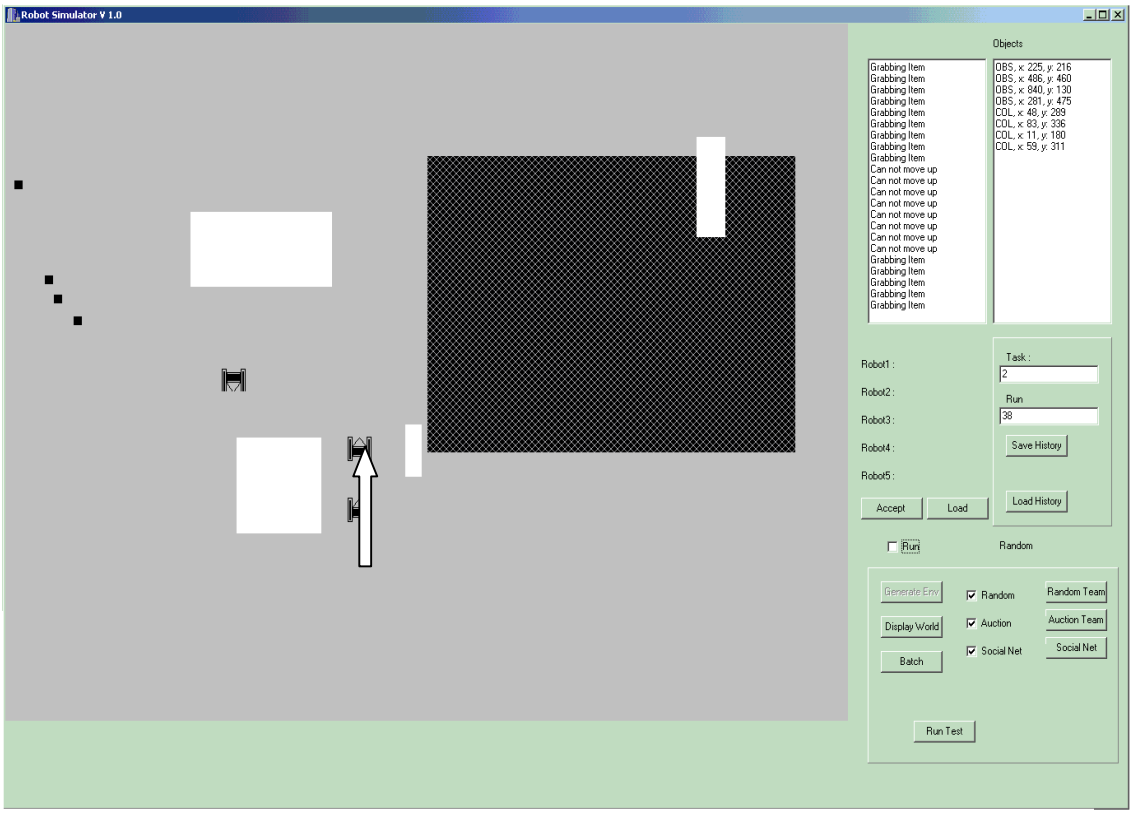

Figure 35. A Screenshot of Robot Simulator

8.2.3 Society Component

The main role of the society component is to facilitate team selection. In the INDABA architecture [157] the agent's interaction layer performs this task, as described in section 5.2. In the robot simulator, the society component allocates the tasks. This is done by finding the robot most suited to the task based on its performance history. If there is no performance history a robot is randomly selected. Then the strengths of social links are calculated using the definitions of kinship as given in section 7.1. The team allocation algorithm is given in pseudo-code in algorithm 11.

Announce Task T

For all Robots
If t(Robot, Robot, T)> t(leader, leader, T)
Leader = Robot
EndFor
If Leader found
For all Robot in society
Calculate s(leader, Robot, T)
EndFor
Select team based on highest s(leader, Robot, T)
Else
Select team based on auctioning
End Else

Execute Task

For all Robots in team
If Task successfully executed
Increase t (leader, Robot,T)
Else
Decrease t (leader, Robot,T)
EndFor

Algorithm 11. Task allocation and task success evaluation in simulated robot environment

8.2.4 Environment Component

The environment component simulates the real-world. It provides sensor readings based on a description of the environment, described in a set-up file which is loaded before execution begins. All the robot actions are validated within the limits and constraints of the environment, i.e. the obstacles and environment variables that influence the outcome of desired robot actions.

8.3 Simulation Set-up and Assumptions

For the purpose of this simulation, robot, environment and task attributes were selected to resemble potential real-world environments and problems. The setup is described with reference to the robots and environments used (refer to section 8.3.1) and the tasks to be performed (refer to section 8.3.2).

8.3.1 Robots and Environments

Each robot is defined by a predetermined set of attributes, as listed in table 22.

ROBOT ATTRIBUTE	POSSIBLE VALUES
LOAD	LOAD_SMALL
	LOAD_NORMAL
AVOIDANCE	AVOIDANCE_AVAILABLE
	AVOIDANCE_NOT_AVAILABLE
DRIVE	DRIVE_WHEEL
	DRIVE_TRACK
	DRIVE_LEG
SPEED	SPEED_LOW
	SPEED_MEDIUM
	SPEED_FAST
DETECTION RANGE	DETECTION_NORMAL
	DETECTION_LIGHT_ONLY
	DETECTION_ADVANCED
POWER	POWER_TETHERED
	POWER_SOLAR
	POWER_BATTERY

Table 22. Robot Attributes and possible values

It is important to note that the robots were defined by a set of attributes similar to that used in the experiments of chapter 7, but not identical.

The environments were defined in a similar manner, with environment attributes listed in table 23, where food refers to collectable items.

ENIVRONMENT ATTRIBUTE	POSSIBLE VALUES
TERRAIN	TERRAIN_NORMAL
	TERRAIN_ROUGH_AREA
LIGHT	LIGHT_NO_OBSTACLES
	LIGHT_SHADED_AREA
OBSTACLES	OBSTACLES_NONE
	OBSTACLES_FEW
	OBSTACLES_MANY
FOOD_DISTANCE	FOOD_FAR
	FOOD_CLOSE
FOOD_WEIGHT	FOOD_LIGHT
	FOOD_HEAVY
	FOOD_MIXED
FOOD_AVAILABILITY	FOOD_PLENTY
	FOOD_AVERAGE
	FOOD_SPARSE

Table 23. Environment attributes and possible values

Each robot action is expressed in terms of actuators, namely:

- Motor for left drive (on/off)
- Motor for right drive (on/off)
- Gripper motor (open/close)

The simulator does not prescribe a limit on the number of different robots and environments that could be created and used. However, for the purpose of experiments presented in this chapter, a total of 15 robots and 15 environments were created, all having randomly selected characteristics.

8.3.2 Tasks

While previous investigations focused on a simple foraging task using the same simulated robot environment [158], this thesis considers two tasks: scouting and then foraging.

For the purpose of these experiments, it is assumed that a robot is aware of its approximate position in relation to the environment. It is also assumed that a scouting robot has knowledge of the approximate position of the area where food is located. Although the robots have an idea of the approximate position of food, they do not have the precise position. Then, during the process of finding the best scout, if a robot reaches the approximate position, then that robot performs a spiral search to reach the food.

The following restrictions, similar but not identical to the rules given in section 7.1, are placed on the interaction between robots and the environment:

- If a robot's LOAD attribute is LOAD_SMALL, it cannot load food that has FOOD_WEIGHT attribute FOOD_HEAVY.
- If a robot's POWER attribute is POWER_SOLAR, it cannot move in an environment area that is in the shade.
- If a robot's POWER attribute is POWER_TETHERED, it is limited in range.
- The detection range is reduced in the shaded area and if a robot's DETECTION_RANGE attribute is DETECTION_LIGHT_ONLY, the robot cannot detect objects in shaded areas.
- A robot's progress is determined based on terrain, drive and speed (refer to section 7.1).

For each environment, the number of obstacles (depending on the value of the OBSTACLES attribute namely none, many (3-7 obstacles) or few (1-3 obstacles)) is randomly created. The same applies to food (depending on the value of the FOOD_AVAILABILITY attribute). Obstacles positioning is random, while food positioning is also random, but within the limits of the value of environment

parameter FOOD_DISTANCE. It is important to note that, due to the randomness of the environment attributes, in some cases a task could not be successfully executed at all. To illustrate, during the experiments conducted here, two of the environments were not successful due to the food being blocked by obstacles, so foraging was not possible. A consideration was given to recreating those two environment, but it was decided to use them unchanged. The reason for such decision is that, in real-world, it may happen that certain environments are just too difficult for robots to execute an allocated task [131].

For each environment, three different strategies have been used to select teams: (1) random team selection, (2) team selection based on auctioning and (3) team selection based on social networks. Using each strategy, two teams were formed, a scouting team and a foraging team. The scouting team consisted of only one robot, while three robots were in each foraging team. Each team was allowed to perform the task for a limited period of time (50 seconds).

If a robot did not complete the allocated task in the prescribed period of time (i.e. the robot did not find the food in the case of the scouting task, or did not return with the collected food to the home area in the case of the foraging task), it is considered that the robot failed to complete the task.

8.4 Simulation Results

In order to simulate a condition of market failure due to uncertainty, uncertainty about the task was introduced. Ucertainty was included by having incomplete information about environment attributes. For the purpose of this section, information was available for only FOOD_DISTANCE, FOOD_WEIGHT and TERRAIN attributes.

As mentioned in the previous section, each team had a limited size. If more than the allowed number of team members satisfied the auctioning bid, teams were selected randomly from these robots until the team size constraint has been met. For the purposes of this chapter, two experiments have been conducted.

For the first experiment, performance was evaluated when the same selection method is used for both the scouting team and foraging team. Performance was evaluated for the three different selection strategies (i.e. auction, social networks and random).

For the second experiment, the selection strategy used for the two teams were not necessarily the same. The strategy for selecting the scout was random selection. For the second experiment, three different studies have been done, where each study uses a different selection strategy for the foraging team.

Each experiment consisted of six simulations, where each simulation was run for forty-five execution cycles. Three execution cycles (each using a different selection strategy) were applied to each of fifteen randomly generated environments. It is important to note that each execution cycle builds on the social network, consisting of trust and kinship relationship, established during the previous execution cycle.

8.4.1 Results Using Same Selection Method

For the results reported in this section, the scout and foraging team selection utilise the same selection method. The results are illustrated in figure 36.

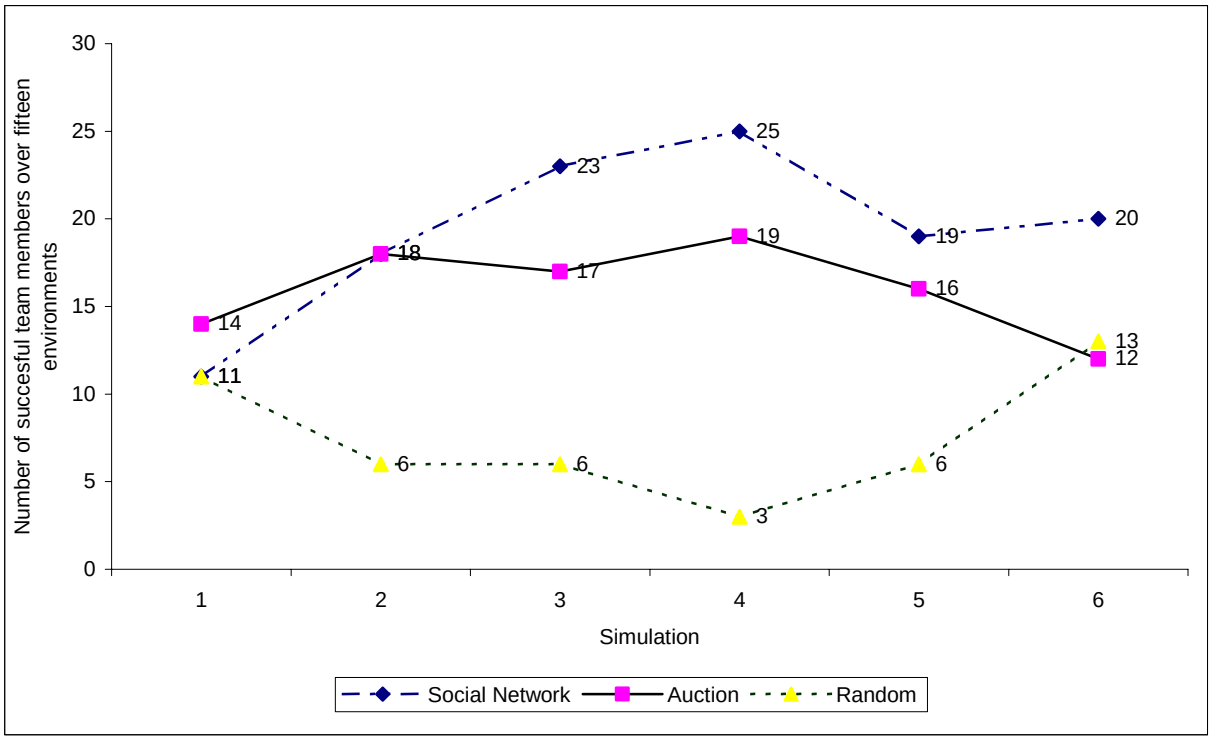

Figure 36. Comparative results of three selection methods over six simulations (same selection methods for both tasks)

It is important to note that for each of the three selection strategies, maximum number of successful agents per simulation is forty-five (three robots per team over fifteen randomly created environments). The social networks based team allocation mechanism on average performed better than the auctioning or random selection team allocation mechanisms.

In the first simulation, at which point there is no history, the social networks based team selection utilises auctioning. Therefor, there is no significant difference between the performances of the auctioning and social network team selection methods. Only after the third run, does the social networks approach start to outperform the auctioning method. The reason for this improvement in performance is due to the trust that was developed between the agents during the first two runs. In the third run, trust and kinship start to play the primary role in team selection. The performance of the auctioning system fluctuates, but on average does not improve. The random selection method performs far worse than the auctioning and social networks methods.

Fluctuations were observed in the performance of all three selection methods. It is important to note that these fluctuations are due to the uncertainty of the robot and food positioning, which are randomly selected for each environment (in case of food positioning) and for each task execution (in case of robot positioning). To quantify the uncertainty, in the experiment reported in this section robots were randomly positioned 2700 times (four robots (one scouting and three foraging robots) using three selection strategies for each of the fifteen environments in six simulations).

8.4.2 Random Scout Selection Method Simulation Results

This section reports results of using different selection method for the scout and foraging teams. Scout selection uses the random selection method, while the foraging team selection respectively utilise auctioning, social networks or random selection methods. The results are illustrated in figure 37.

Again the social networks based team allocation mechanism performed on average better than the auctioning and random selection team allocation mechanisms, but not

as well as for the previous experiments (refer to section 8.4.1). The lower performance can be attributed to the fact that whole task fails if the scout does not successfully complete its task. Random selection method does not guarantee that the best robot is selected for scouting task, while social networks based selection does (based on historical performance). The same selection method is therefore the recommended approach to fully utilise the advantages of the social networks based approach.

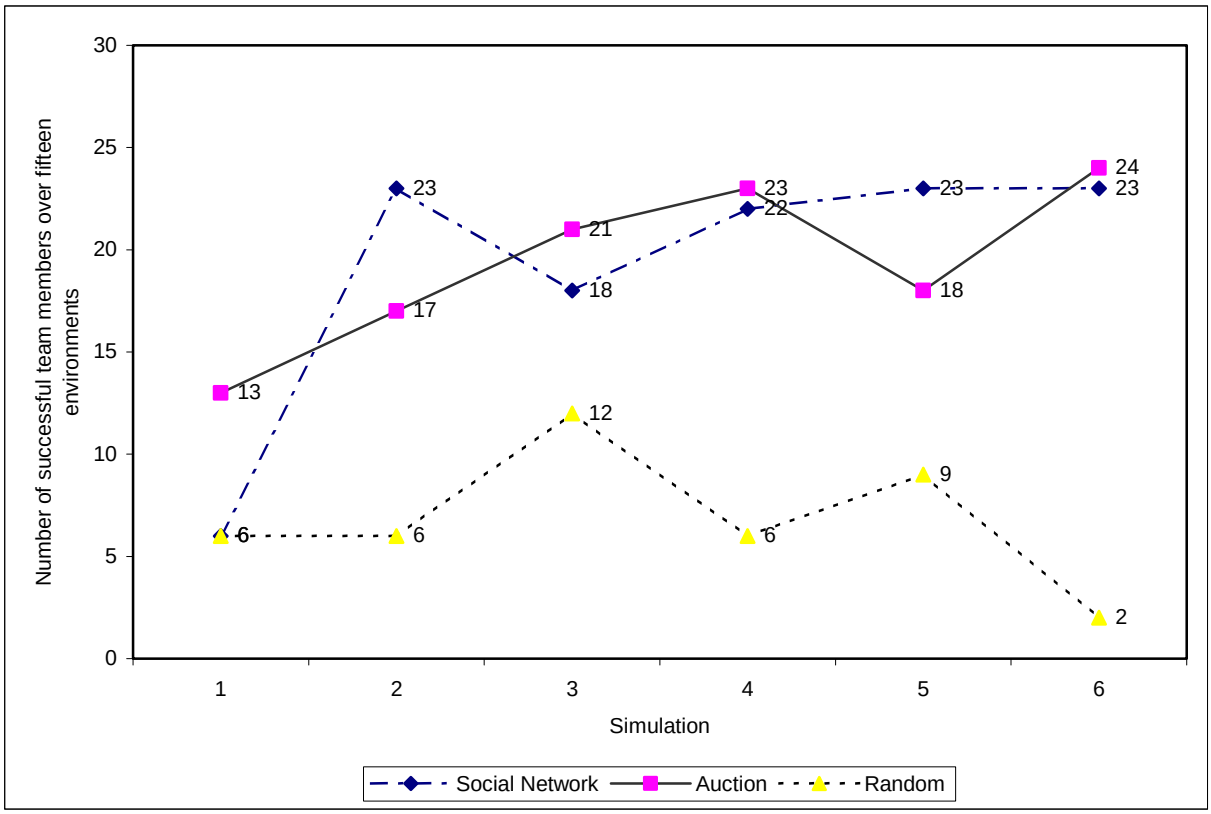

Figure 37. Comparative results of three selection methods over six execution cycles (inconsistent selection)

For both experiments it was noted that the social networks approach improved its performance over time, allowing for minor fluctuations. Social networks store information about relationships (kinship and trust) between members of the society. The stored trust relationship information is derived from the historical performance of the team members (refer to section 8.2.3 and algorithm 11). This information is used in the process of team selection, and leads to improvement in performance of the selected team.

The improvement in performance, due to the better team selection method based on kinship and trust relationships, can be seen as a learning capability.

8.5 Summary

The INDABA architecture was used as the agent architecture in a simulated robot environment, which was presented in this chapter. Scouting and foraging tasks were simulated in this environment and the results were presented and discussed. The social networks based approach was used as the main team selection mechanism. A comparison was made to alternative team selection strategies, namely auctioning and random selection. The results of conducted experiments indicate that social networks based approach performs better than alternative team selection strategies.

The next chapter presents a full INDABA implementation in a physical environment, using a readily available robotic platform.