Chapter 7: Experiments in an Abstract Simulated Environment

The focus of this thesis is on an architecture for robotic systems and on the use of social networks as a coordination mechanism. The development of a realistic multi-robot simulated environment is not a trivial task and a decision was made to verify the validity of the INDABA architecture and coordination mechanism through a simpler, abstract multi-robot simulator. The abstract simulator set-up and its scope limitations are presented in section 7.1. The abstract simulator has two main algorithmic components. The first of the two is the task allocation and team formation algorithm, presented in section 7.2. The second algorithmic component is the task execution and task evaluation algorithm, presented in section 7.3. The simulations of uncertainties about task details are presented in section 7.4. The majority of simulations that were conducted for the purpose of this thesis are presented, together with the results and a discussion of these results in section 7.5. A summary of the results, as presented in section 7.6, concludes this chapter.

7.1 Scope Limitation and Simulation Set-up

While previous investigations focussed on social networks and coordination in abstract terms [154][156], this chapter the focuses on an abstract simulator. The main purpose of the abstract simulator used in this thesis is to provide a platform for simulations to explore the proposed social networks based approach to task allocation. The simulated environment implements only the two upper layers of the INDABA framework (see chapter 5), namely the interaction and the deliberator layers. The other two layers, sequencer and controller, are grossly simplified in this simulation. Such approach is justified since only the upper two layers of INDABA are relevant to coordination in general and to the task allocation problem in particular. In the experiments presented in this chapter, a population of fifty agents was randomly created. All agents are defined by the same set of attributes, with the attribute values

randomly selected from the domain of each attribute. The agent attributes and possible attribute values are given in table 7.

AGENT ATTRIBUTE	POSSIBLE VALUES
LOAD	LOAD_SMALL
	LOAD_NORMAL
FALSE FOOD SENSOR	PRESENT
	NOT_PRESENT
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 7. Simulated agent attributes and possible attribute values

Environments are defined in a similar manner to agents. Each environment is defined by the same set of attributes that represents physical characteristics of the environment. Environments were also created randomly, by assigning random values to the environment attributes. The environment attributes and valid attribute values are given in table 8.

ENIVRONMENT ATTRIBUTE	POSSIBLE VALUES
TERRAIN	TERRAIN_NORMAL
	TERRAIN_ROUGH_AREA
LIGHT	NO_SHADED_AREAS
	SHADED_AREAS
FOOD_DISTANCE	FOOD_FAR
	FOOD_CLOSE
FOOD_TYPE	FOOD_LIGHT
	FOOD_HEAVY
	FOOD_MIXED

Table 8. Simulated environment attributes and possible attribute values

The following rules define interactions that may occur between the environment and agents:

- If the robot LOAD attribute is LOAD_SMALL, it cannot load food that has FOOD_WEIGHT attribute FOOD_HEAVY.

- If the robot has as a POWER attribute value of POWER_SOLAR, it cannot move in an environment area that is in the shade.
- If the robot has POWER attribute POWER_TETHERED, it is limited in range.
- The detection range is reduced in the shaded area and if the robot DETECTION_RANGE attribute has value DETECTION_LIGHT_ONLY, the robot cannot detect objects at all.
- The number of steps required for robot movement is a function of the environment terrain (if it is in TERRAIN_ROUGH_AREA), drive and speed. For example, if a robot is in rough terrain area, and if the robot's drive attribute is track or wheel, then the number of required steps is increased by 30% and 60%, respectively. The values of 30% and 60% are arbitrarily chosen. In future work, these values will be changed according to observed decrease in performance of a real robot.
- If a robot is not equipped with FALSE_FOOD_SENSOR and the environment FOOD_TYPE is FOOD_MIXED, it has a 30% chance of picking up false food and failing the task.

For the purpose of simulations in the abstract simulated environment, two types of social relationships were implemented. The implemented social relationships were based on the concepts of trust and kinship, which are the fundamental concepts in the formation of social networks in complex animal societies. These concepts are defined next.

7.1.1 Kinship

Kinship is defined as the similarity between the simulated robots. A simple metric that quantifies kinship, $d(R_1, R_2)$, between robots R_1 and R_2 is calculated as

$$d(R_1, R_2) = \Sigma_{i=1,..,N} d_i (A_{1i}, A_{2i}) / N \qquad (7.1)$$

where N is the number of robot attributes, $d_i (A_{1i}, A_{2i})$ is a normalised metric function in the range [0, 1] between the i-th attribute of robots R_1 and R_2, and A_{1i} and A_{2i} are

the values of the *i*-th attribute of robots R_1 and R_2 respectively. Function $d_i\ (A_{1i}, A_{2i})$ quantifies the difference between the attributes A_{1i} and A_{2i}.

7.1.2 Trust

Let task T be defined by attributes $T_1, \ldots, T_m$, where m is the total number of attributes that define task T. Trust is then defined as a 3-tuple, $t\ (R_1, R_2, T)$, where R_1 and R_2 are robots, and T is a particular task. Tuple (R_1, R_2, T) quantifies the reliability of robot R_1 in relation to R_2, based on the historical performance related to task T that has involved both robots; in other words, how much trust R_1 has in R_2 in helping to complete task T. Trust is calculated as a ratio between the number of successful task executions and the number of task execution attempts. Note that in the approach presented in this thesis, trust is a symmetric function, i.e. $t\ (R_1, R_2, T) = t\ (R_2, R_1, T)$. The value is normalised to the range [0, 1]. Note that $t\ (R, R, T)$ represents the historical performance of robot R in relation to task T; in other words, the trust that robot R has in its own abilities.

For the purpose of simulations presented in this chapter, the strength of a social link, $s\ (R_1, R_2, T)$, between two agents is defined as :

$$s\ (R_1, R_2, T) = kd\ (R_1, R_2) + (1-k)t(R_1, R_2, T) \qquad (7.2)$$

where $k \in [0,1]$.

7.2 Task Allocation and Team Formation Algorithm

The algorithm starts with task details propagation. During this phase, the known task details are propagated to all agents. For the purpose of this thesis, task details for the simulations presented in this chapter consist of three parts:

- ENVIRONMENT_DETAILS, where known environment details are propagated. The environment details are implemented as a set of attributes. The attributes have discrete values, as given in table 6.
- TASK TYPE, namely scout or forage.

- CONSTRAINT, which represents a time constraint, as the maximum number of steps that each robot is allowed to execute.

The first task to be executed is that of scouting. To illustrate the propagation of task details, consider that a contracting agent sends task details, as described above, to all available robots with a request to the bid for a scouting task. If an agent is not available, the agent will not respond to the bid. Each of the available robots evaluates its own suitability (affinity) to the task, based on task attributes and previous experience (as explained below). This part of algorithm is based on general team leader selection algorithm 7.

The evaluation of task affinity is a two-step process, based on each robot's history. Firstly, the environment, as given by ENVIRONMENT_DETAILS is identified. Identification is done by means of encoding the environment according to its known attributes to produce an environment identifier. Secondly, the environment identifier and TASK_TYPE are used as indices in a table that stores the robot's previous experience in the environment identified by the environment identifier, related to the task identified by TASK_TYPE. The experience quantifier, expressed as a ratio between successful task executions and total number of task execution attempts, is then returned as the robot's bid. After all available agents have entered a bid, the scouting task is allocated to the agent with the highest bid.

The role of a scout is to explore the environment and to determine the values of the environment attributes as given in table 6. It is important to note that the scout might not be able to get accurate information about the environment. For example, if a path from the start point to the food concentration area does not cross a "shaded" area, the scout will not set the LIGHT environment attribute to SHADED_AREAS.

Once the scout completes its task, task details based on the observed environment are then propagated to all agents and the algorithm repeats the bidding process as described above and in more general terms, as given in section 6.6.4.2. The robot with the highest bid is selected as the team leader and given the responsibility of seeing the task to completion. If a number of robots have the same bid value, one is randomly

chosen. If the task exceeds the capability of the team leader, then the team leader selects a team that can execute the task. The team selection algorithm is based on algorithm 8.

The team leader forms a team according to the social relationships between the team leader and members of the team. Team selection starts by calculating all social relationships (trust and kinship). Trust is obtained by using a trust function, as described in section 7.1. Kinship can be obtained either by requesting a full set of attributes from a potential team member and calculating the kinship value, as given in equation 7.1, or by using the values that were predetermined and stored in each robot's kinship table. For the purpose of this thesis, predetermined kinship values were used. Once trust and kinship are calculated, the team is then formed by selecting those agents with strongest social relationships to the team leader. The task allocation and team formation process is described in algorithm 10.

Potential recognition

If not Auctioning agent

Receive task attributes

Identify environment

Send bid based on history and availability

Else

Send all agents task details

Collect bids

Award task to the highest bidder (team leader)

End

Team Formation

If team leader

For all agents

Evaluate strength of social link between agent and team leader

End For

Select the team members according to the strength of social links

EndIf

Algorithm 10. Potential recognition and team formation processes

The foraging task is a loosely coupled task, in the sense that each robot can execute its own task without relying on other robots, and each member of the team executes the same task in parallel.

Once the team is selected, the task is executed. Upon completion (successful or unsuccessful), the success of the task is evaluated. The implementation of the task execution and evaluation algorithms is discussed next.

7.3 Task Execution and Task Evaluation Algorithm

Task execution is simulated in this chapter. For the purpose of this simulation, tasks were simulated and evaluated according to the rules outlined in section 7.1. The simulated task execution is discussed next.

7.3.1 Task Execution

In order to simulate interaction with the environment, each robot is provided with a full set of environment attributes (not the potentially inaccurate set of environment attribute values as observed by the scout) and simulated execution takes place using the full environment set of attributes. Each robot action requires a number of steps. The number of steps that it will take the agent to execute a task is calculated, based on the robot's attributes and environment and interaction rules (as described in section 7.1.). For example, the number of steps for a robot to move from coordinate (x,y) to coordinate (x+1, y) is dependent on the robot's attributes SPEED and DRIVE, and the environment at coordinate (x+1,y). An execution cycle is the completed simulated execution of an allocated task, after which the success of the task is determined. The number of allowed steps limits the lifetime of an execution cycle. If a robot has not yet completed the task when the number of allowed steps is exceeded the robot is considered to have failed in its task.

7.3.2 Task Evaluation

As mentioned, if a robot exceeds a predetermined threshold, then the robot fails in its allocated task. The threshold is selected as the average number of steps required for successful task executions, averaged over a randomly chosen observation period (10 simulations, each consisting of 100 execution cycles) and a team of five robots, randomly chosen from the population of fifty robots.

If the execution of a task was successful, all the successful robots are rewarded. The strength of the social links between the successful agents that have participated in the same team will be reinforced, by raising the level of trust between the successful team members. For each successful robot R_j in the team, its trust rating $t\ (R_j\ ,\ R_k\ ,\ T)$ are improved by means of increasing its ratio between number of successful task executions and number of attempts of task execution. The increase is an additive increase. In other words, the number of successful task executions is increased by one. The trust rating $t\ (R_j\ ,\ R_k\ ,\ T)$ is related to task T and other successful members of the team R_k , where $k = 1,..,\ n$, and n is the number of successful team members. Task success evaluation is a function that awards agents that have successfully completed their allocated tasks, with or without the help of other agents. Each agent R_j also keeps its own trust $t\ (R_j\ ,\ R_j\ ,\ T)$ which represents its own historical performance, relative to a particular task's details. This in turn tracks an agent's affinity to a particular task type.

7.4 Simulating Task Details Uncertainty

The view adopted in this thesis is that uncertainty is unavoidable in real-world robotic applications. All experiments done for the purpose of this thesis include varying degrees of uncertainty. Basically, there are two causes of uncertainty that affecti the experiments, namely uncertainty due to environment variations and uncertainty due to the initial robot positioning. These causes of uncertainty are described next.

7.4.1 Uncertainty due to Environment Variations

As described in section 7.1, each environment is defined by a set of attributes. It is important to note that even if two environments have the same attribute values, as defined in ENVIRONMENT_DETAILS (section 7.1), the two environments are not necessarily the same. Even though environment attribute values may be the same, the location of rough terrain, shaded areas and food may differ due to random creation of these aspects. Random creation of environments brings a level of task uncertainty. Uncertainty due to environment variations is the main contributor to uncertainty in the experiments that are presented in section 7.5.

7.4.2 Uncertainty due to Initial Robot Positioning

Uncertainty due to initial robot positioning has a lesser influence on task execution than uncertainty due to environment variations. Uncertainty due to the initial robot positioning is caused by randomly initialising each robot's position on different coordinates of the home area (which is defined as the lower right corner of the environment). Random initial positions have an influence on the number of steps required to reach the food area and also contributes to uncertainty about task execution.

To illustrate the influence of uncertainty due to initial robot positioning to successful task completion, two simulations were done. The first simulation was done with a constant initial position for all robots, while the second simulation was done with variable initial robot positions but within the limits of the home area. Each simulation consisted of 100 execution cycles. The team size was limited to six team members.

The results are presented in figure 14. In the case of a constant initial position for robots, optimal team selection was achieved early in the simulation - in the first 20 execution cycles. In the case of variable initial robot positions, team selection was achieved later in the simulation. The results were never as good as in the case of the constant initial position. However, in case of variable initial robot positions, once a

team was selected, the selected team was more resilient to change in the initial positioning.

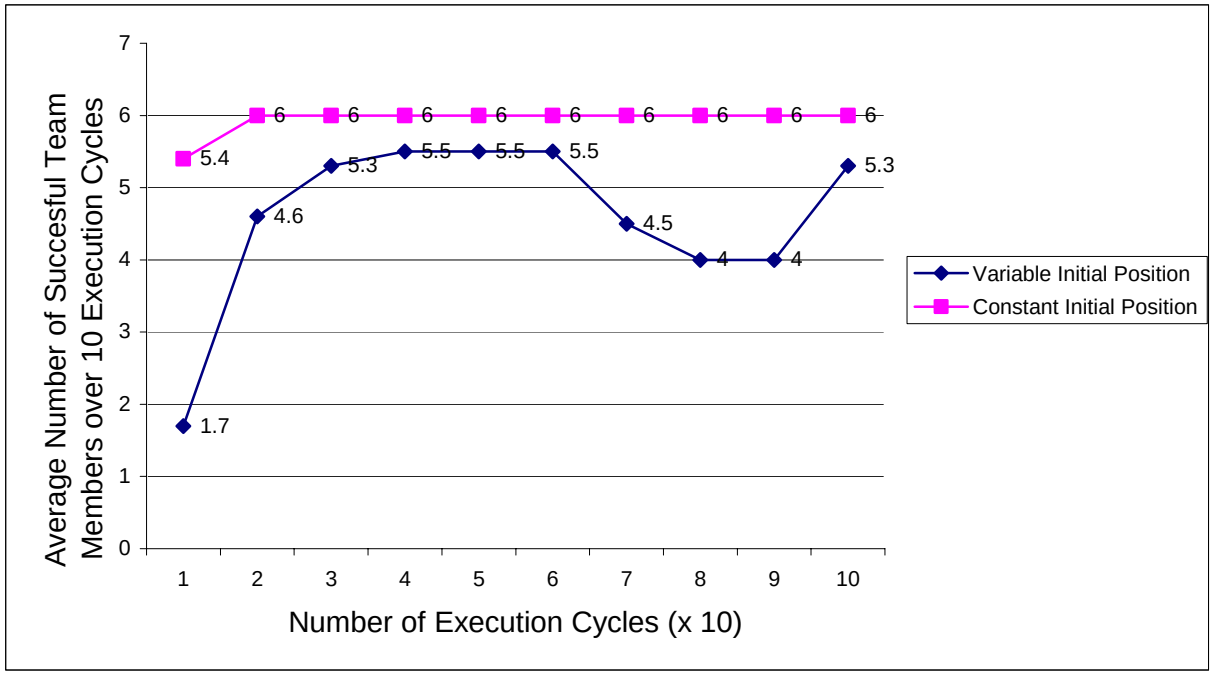

Figure 14. The Effect of Uncertainty due to the Initial Robot Positioning

7.5 Experimental Results

In order to verify the validity of the social network based approach for task allocation, various experiments were performed. When interpreting the experimental results, various aspects of the social network based approach were considered, ranging from learning to interpretation of the results in the light of social sciences.

The effects of uncertainty are visible in the results presented in this section. Even when the team is stable (consisting of the same team members), there are some fluctuations in the results. The fluctuations are there because of the introduced uncertainty, as described in section 7.4.

The experiments are divided into the following categories:

- Experiments that compare the social networks based approach to an auction based approach in relation to a single environment type.

- Experiments that compare the social networks based approach to an auction based approach in relation to multiple environment types.
- Experiments that explore the specialisation ability of robots with regard to a particular task.
- Experiments that explore learning capabilities of the social networks based approach to coordination.
- Experiments that explore the influence of changes in ratio between trust, kinship and history on the selection process.
- Experiments that explore the effects of probabilistic selection on the overall performance.
- Experiment that investigates formation of subgroups within a social network.

In all of the above enumerated simulations the same population of fifty randomly-created agents was used. The population, with its attribute values, is given in table 9. Each row represents one of the agents in the population.

LOAD	DRIVE	SPEED	DETECTION	AUTONOMY	FOODSENSOR
SMALL	WHEEL	FAST	NORMAL	TETHERED	YES
NORMAL	WHEEL	LOW	LIGHT ONLY	SOLAR ONLY	NO
SMALL	LEGS	FAST	LIGHT ONLY	SOLAR ONLY	NO
SMALL	WHEEL	FAST	NORMAL	SOLAR ONLY	YES
SMALL	TRACK	LOW	LIGHT ONLY	BATTERY	YES
NORMAL	TRACK	LOW	LIGHT ONLY	SOLAR ONLY	YES
NONE	TRACK	LOW	LIGHT ONLY	TETHERED	YES
NONE	WHEEL	LOW	NORMAL	BATTERY	YES
NONE	LEGS	MEDIUM	NORMAL	SOLAR ONLY	NO
NORMAL	TRACK	FAST	NORMAL	BATTERY	NO
NONE	LEGS	FAST	LIGHT ONLY	SOLAR ONLY	NO
NONE	WHEEL	MEDIUM	LONG RANGE	TETHERED	YES
NORMAL	LEGS	FAST	LONG RANGE	BATTERY	NO
NONE	LEGS	LOW	LIGHT ONLY	TETHERED	YES
NORMAL	LEGS	FAST	NORMAL	BATTERY	YES
NORMAL	WHEEL	MEDIUM	NORMAL	BATTERY	NO
NONE	WHEEL	FAST	LONG RANGE	SOLAR ONLY	NO
NONE	TRACK	LOW	LONG RANGE	TETHERED	NO
NORMAL	TRACK	LOW	LONG RANGE	TETHERED	YES
SMALL	LEGS	FAST	LONG RANGE	TETHERED	NO
NONE	LEGS	FAST	NORMAL	BATTERY	NO
NONE	LEGS	MEDIUM	LIGHT ONLY	SOLAR ONLY	NO
NORMAL	LEGS	MEDIUM	NORMAL	SOLAR ONLY	YES
NORMAL	LEGS	FAST	NORMAL	BATTERY	YES

NONE	TRACK	LOW	LONG RANGE	BATTERY	NO
NONE	TRACK	MEDIUM	LONG RANGE	BATTERY	NO
SMALL	WHEEL	FAST	NORMAL	BATTERY	NO
NONE	LEGS	LOW	LONG RANGE	BATTERY	YES
NONE	TRACK	MEDIUM	NORMAL	BATTERY	YES
NONE	WHEEL	LOW	NORMAL	SOLAR ONLY	YES
NONE	WHEEL	MEDIUM	LIGHT ONLY	BATTERY	NO
NONE	LEGS	MEDIUM	NORMAL	SOLAR ONLY	YES
NORMAL	WHEEL	FAST	NORMAL	BATTERY	NO
NORMAL	TRACK	FAST	LONG RANGE	TETHERED	YES
NONE	TRACK	LOW	NORMAL	TETHERED	YES
NORMAL	WHEEL	MEDIUM	LIGHT ONLY	SOLAR ONLY	NO
SMALL	WHEEL	LOW	NORMAL	TETHERED	YES
SMALL	TRACK	LOW	LONG RANGE	BATTERY	YES
SMALL	WHEEL	LOW	NORMAL	SOLAR ONLY	YES
NORMAL	TRACK	LOW	LIGHT ONLY	BATTERY	YES
NONE	WHEEL	FAST	LONG RANGE	SOLAR ONLY	YES
NORMAL	WHEEL	LOW	NORMAL	BATTERY	NO
SMALL	WHEEL	FAST	LIGHT ONLY	BATTERY	NO
NONE	LEGS	FAST	LONG RANGE	SOLAR ONLY	NO
NONE	WHEEL	MEDIUM	LIGHT ONLY	SOLAR ONLY	NO
NONE	WHEEL	MEDIUM	NORMAL	TETHERED	YES
NONE	TRACK	MEDIUM	LIGHT ONLY	TETHERED	NO
NORMAL	WHEEL	MEDIUM	LIGHT ONLY	BATTERY	YES
NORMAL	WHEEL	MEDIUM	NORMAL	BATTERY	YES
NORMAL	LEGS	LOW	NORMAL	BATTERY	YES

Table 9. Agent population created and used for experiments in this chapter

Each experiment used the same agent population. Each experiment was done using a simulation that consists of a number of execution cycles, usually 100 execution cycles (unless stated otherwise). An execution cycle consists of execution of a scouting task and a foraging task. Each of the tasks in turn consists of the five cooperative problem-solving cycle components as given in section 5.5.4.

It is important to note that each execution cycle can be seen as a small-scale simulation, as it independently simulates a complete task execution. Each execution cycle is subject to uncertainty due to the initial robot positioning (refer to section 7.4.2) and usually to uncertainty due to environment variations (refer to section 7.4.1). Even if the environments are of the same type, they are randomly created with random positioning of food items and obstacles. However, the execution cycles are not totally independent simulations. Historical performance, as embedded in trust relationships, is passed from one execution cycle to the next one. In other words,

through social relationships, the society maintains a performance history for each agent.

For each experiment, the simulation was defined using a set of parameters. The default simulation parameters are given in table 10. In all simulations, initial trust between agents is set to zero. In other words, there is no history between the agents. In the case of a scouting task, there is only one agent who executes the scouting task. In the case of the foraging task, the default (and maximum) team size is six members. The maximum number of successful team members is therefore six. The default environment type is variable, in other words the simulation is not restricted to a single environment type.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	100
Environment Type	Variable
Random Initial Positioning	Yes
Foraging Team Size	6

Table 10. Default simulation parameters

The results of simulations are presented and discussed next. It is important to note that the data points in figures 14-21 represent the average value over 10 execution cycles.

7.5.1 Performance Comparison to an Auction Based Approach (Single Environment Type)

For the first experiment, the performance of the new social networks based approach to task allocation was compared to the performance of a simple auctioning mechanism. Only uncertainty due to environment variations was introduced in this simulation. The same environment type is used. Table 11 provides a summary of the simulation parameters.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	200 – 300
Environment Type	Single
Random Initial Positioning	Yes
Foraging Team Size	6

Table 11. Default simulation parameter for a performance comparison to an auction based approach (single environment) simulation.

The auctioning mechanism is aware of all ENVIRONMENT_DETAILS attribute values, with the exception of the LIGHT attribute. The auctioning mechanism selects agents according to environment attribute and rules, as described section 7.1. Uncertainty is simulated by omitting the LIGHT attribute. Figure 15 gives the results after 200 execution cycles.

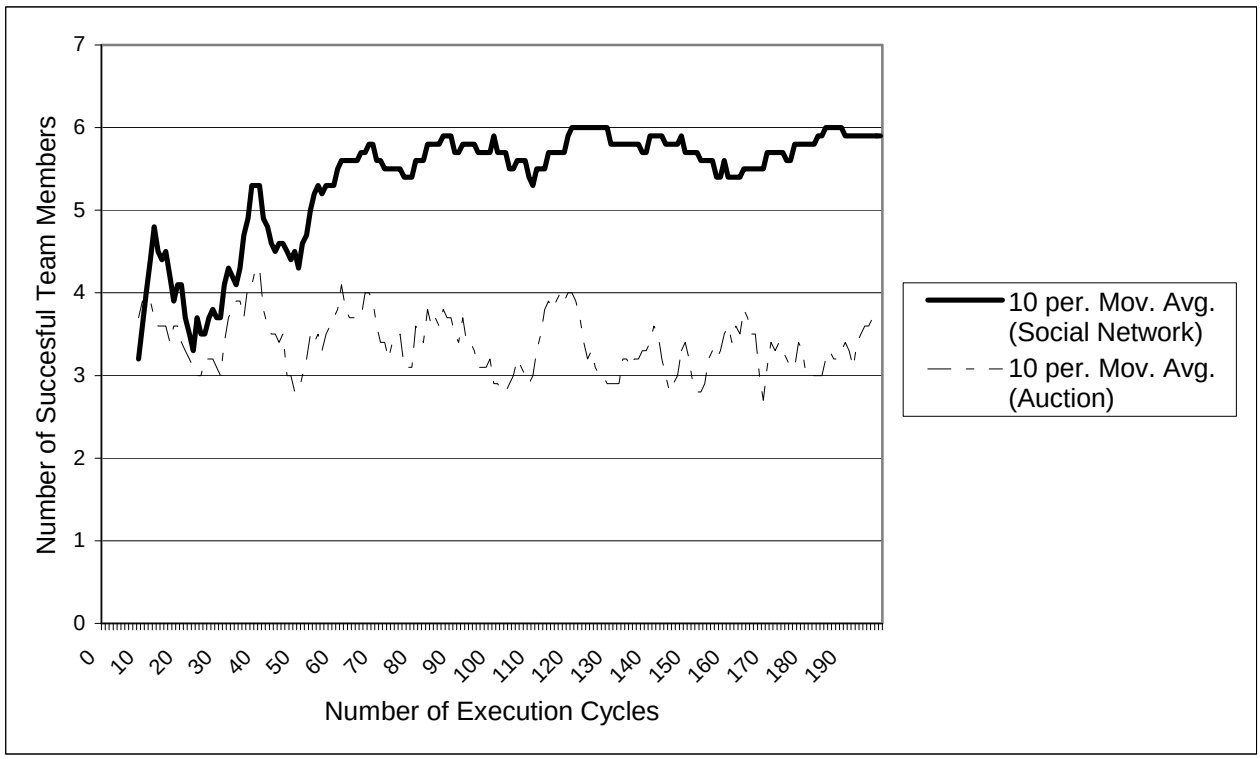

Figure 15. Performance comparison between social networks based approach and auctioning approach on single environment

The main difference between the simple auctioning and the social networks based approach is in the prior knowledge used. The social networks based approach has no prior knowledge and performance is initially less than the performance of the auctioning mechanism. The experiments showed that auctioning mechanism performance remains more or less stable, while the performance of the social networks approach improves over time. Social networks approach during certain execution cycles reached the optimal performance (all team members were successful). To confirm the stability of social networks approach, an additional 100 execution cycles were executed, with the results given in figure 16.

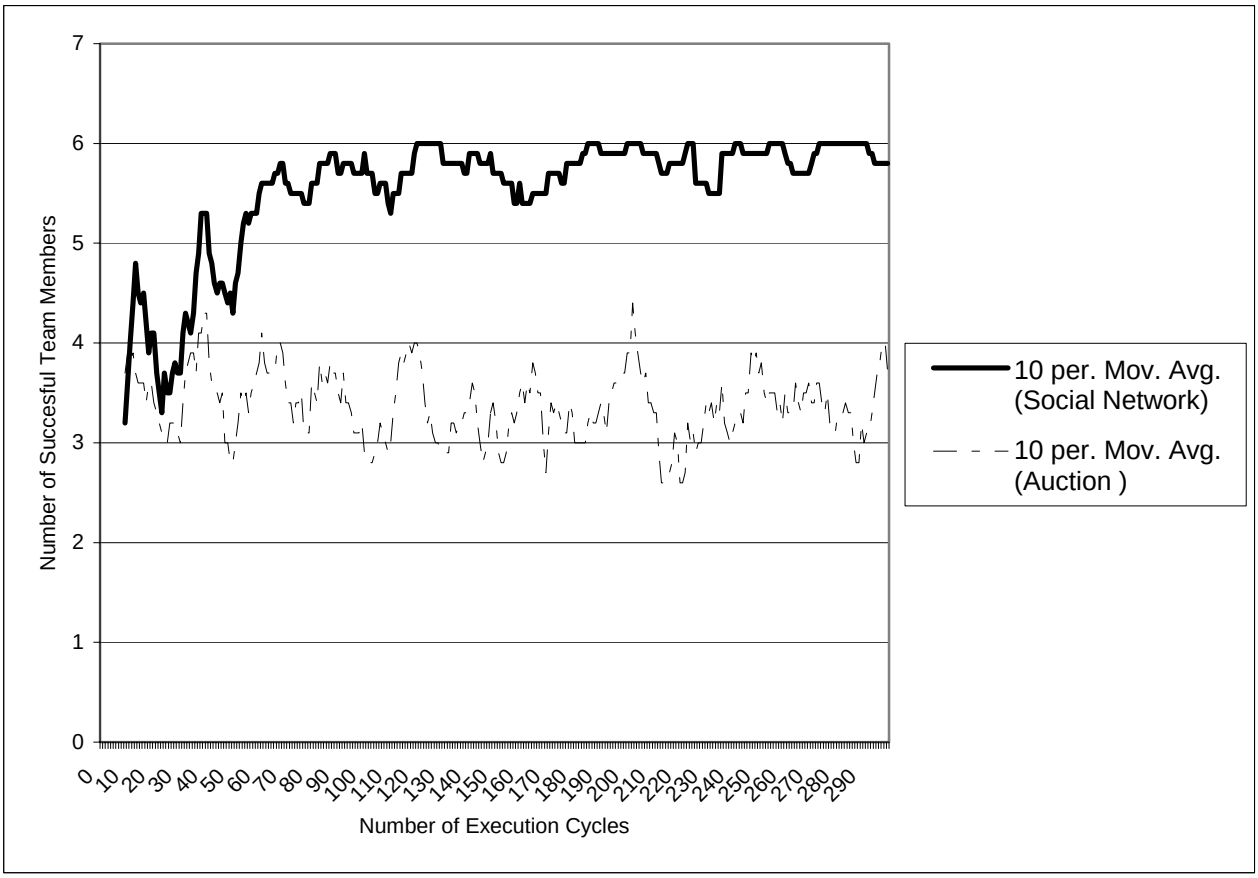

Figure 16. Performance comparison between social networks based approach and auctioning approach on single environment (300 execution cycles)

Based on these experiments, it is safe to conclude that the social networks based approach to task allocation performs significantly better than the auctioning mechanism in conditions of limited uncertainty about the task details.

7.5.2 Performance Comparison to an Auction Based Approach (Multiple Environment Types)

The results presented in section 7.5.1 are encouraging and the aim of the next experiment is to investigate if the social networks based approach performs equally well over a greater diversity of environments. In this simulation, the performance of the social network based approach to task allocation was again compared to the performance of a simple auctioning mechanism for task allocation. Uncertainty due to environment variations was introduced in this experiment (as in the previous experiment, refer to section 7.5.1). It is important to note that environments change from execution cycle to execution cycle. For each execution cycle, an environment was created by randomly choosing its attribute values. The aim of the simulation is to investigate if the social networks based approach generalises and if it can handle different environment types. A summary of simulation parameters is given in table 12.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	200 – 300
Environment Type	Variable
Random Initial Positioning	Yes
Foraging Team Size	6

Table 12. Parameters for a performance comparison to an auction based approach (multiple environment types) simulation.

The implementation of the auctioning mechanism is the same as described in section 7.5.1. The results after 200 execution cycles are illustrated in figure 17.

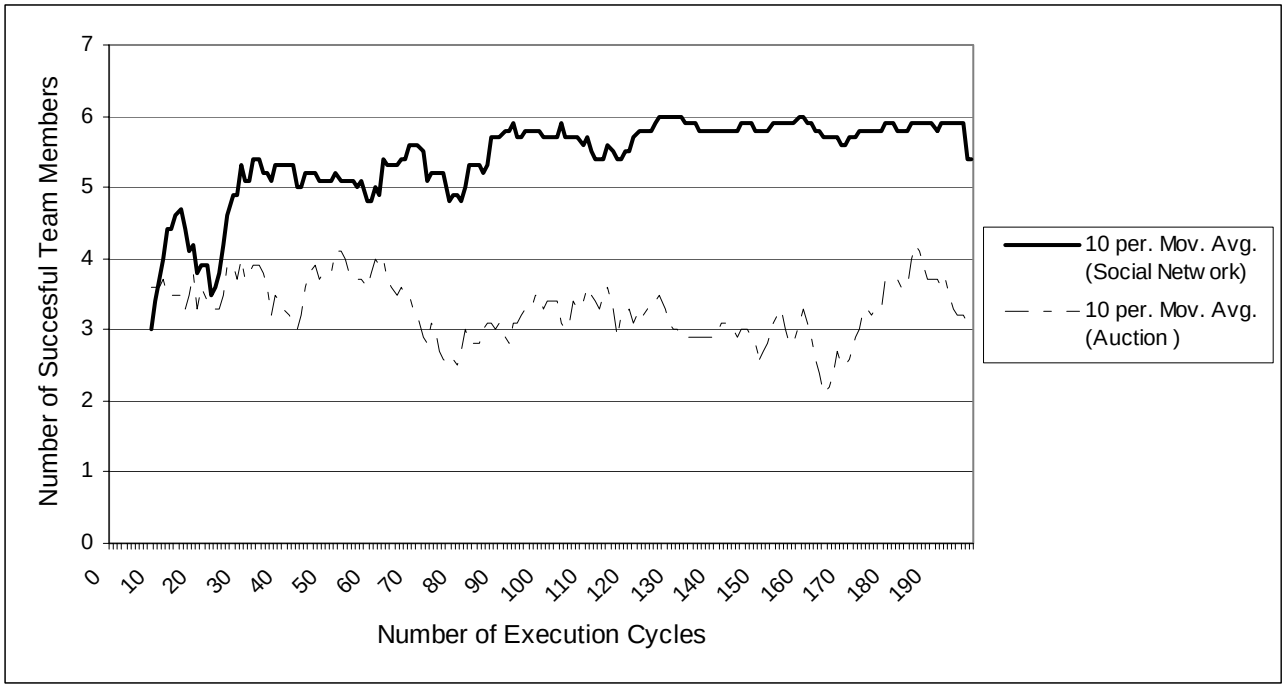

Figure 17. Performance comparison between social networks based approach and auctioning approach on multiple environments (200 execution cycles)

The drop in performance in the last ten execution cycles (refer to figure 17) warranted further investigation and the number of execution cycles was increased to 300 in order to check if it was just a random fluctuation or if it was an indication of a downward trend. As can be seen in figure 18, it was just a random fluctuation.

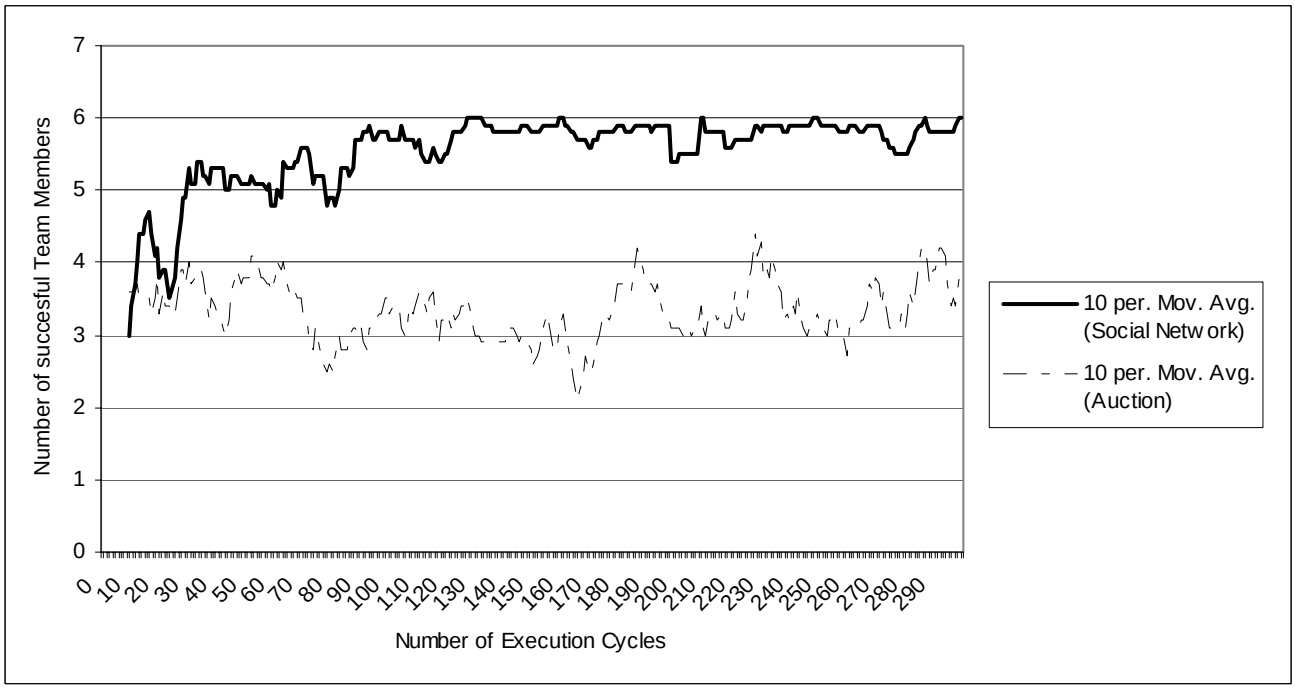

Figure 18. Performance comparison between social networks based approach and auctioning approach on multiple environments (300 execution cycles)

The observed performance leads to the conclusion that the social networks based approach to task allocation performs significantly better than auctioning mechanism in conditions of limited uncertainty over the task details and over multiple environment types.

7.5.3 The Influence of Probabilistic Selection

The social networks based approach uses a straightforward team selection method: the members with the highest scores (bids) are selected. As this is not the only possible team selection method, this section investigates the performance of different team selection methods for the social networks based approach. A probabilistic selection, based on roulette wheel selection, was implemented and compared with standard rank-based selection. Uncertainties due to environment variations and initial positioning were introduced in this experiment. The simulation was executed for 200 execution cycles over single and multiple environment types. A summary of the simulation parameters is given in table 13.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	200
Environment Type	Single, Variable
Random Initial Positioning	Yes
Foraging Team Size	6

Table 13. Simulation parameters used for the investigation of probabilistic selection influence.

Figures 19 and 20 respectively present the results for a single environment and for multiple environments.

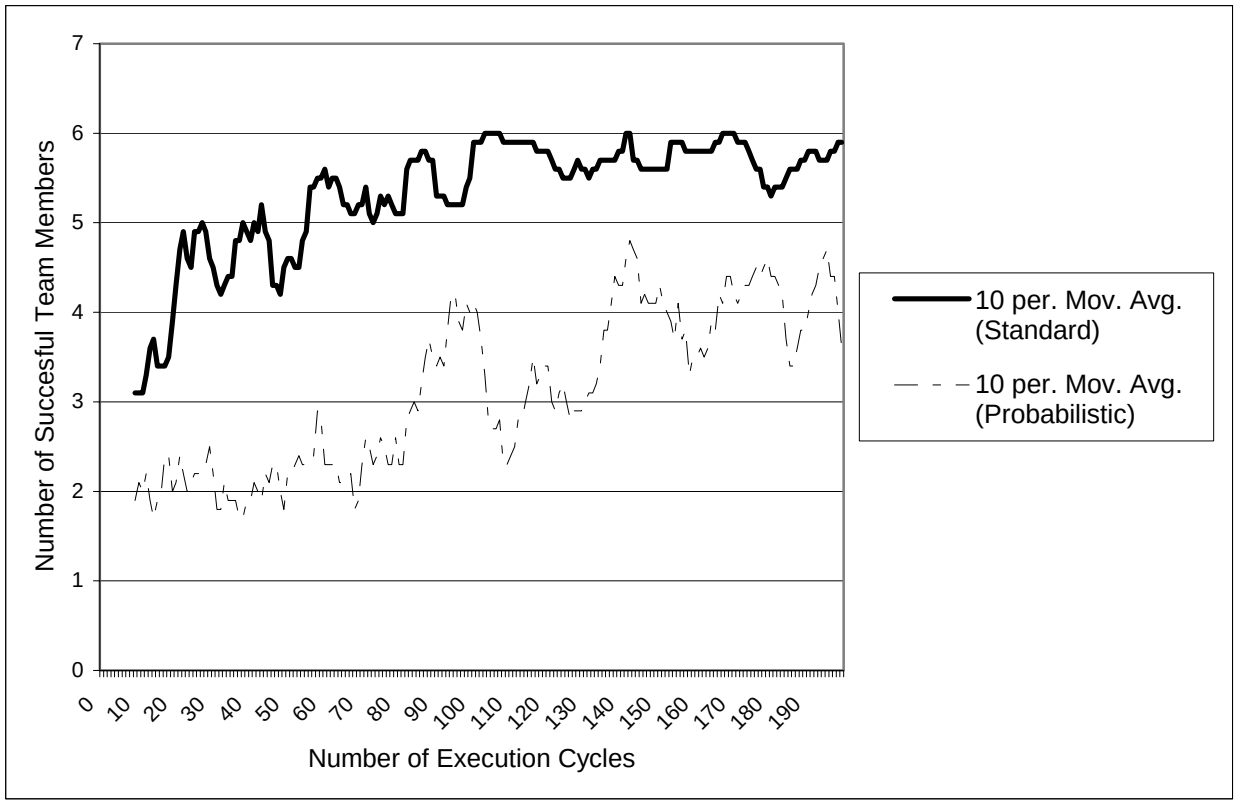

Figure 19. Performance comparison between standard and probabilistic selection on single type environment (200 execution cycles)

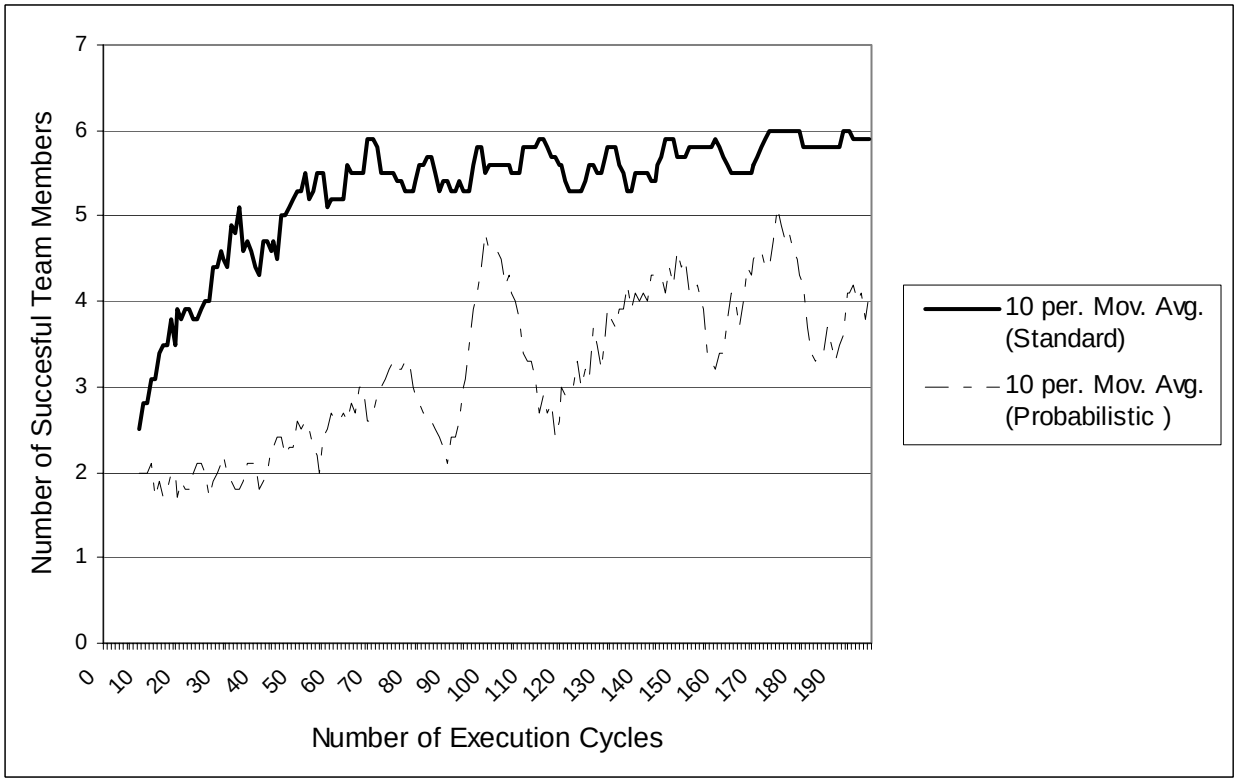

Figure 20. Performance comparison between standard and probabilistic selection over multiple environment types (200 execution cycles)

It is important to note that, although slower than standard rank-based selection, the performance using the probabilistic selection also steadily improves over time. The advantage of probabilistic selection is in the fact that more agents get tested over time for their suitability to a task. This approach might be feasible for future applications that allow for a greater training period and where the availability of agents might be scarce, but in environments were all agents are available and there is a time constraint, the standard approach is more desirable.

7.5.4 Learning Using Social Networks Approach

One of the most useful features of social networks is their ability to store information [179]. The experiments presented in this section concentrate on increasing the number of execution cycles while keeping uncertainty limited to the randomness in the initial positions of the robots. Two simulations were done to investigate the ability of social networks to learn. For the purpose of these experiments, the ability to improve on performance is seen as a learning ability. The performance is defined as the number of successful team members. For the first experiment, a single environment was used, while the second experiment used variable environments. The results are presented and discussed next.

7.5.4.1 Learning over Single Environment

For the simulation in the first experiment, the number of execution cycles has increased in steps of 10, from 0 to 100. Only uncertainty in this experiment is due to initial positioning. A summary of the simulation parameters is given in table 14.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	100
Environment Type	Single
Random Initial Positioning	Yes
Foraging Team Size	6

Table 14. Simulation parameters used for the investigation of probabilistic selection influence

The results are presented in figure 21. The results show that the social networks approach improves performance over time. It is important to note that optimal performance is reached after 8 execution cycles. As previously noted, the improvement in performance is seen as learning ability. The learning characteristic of social networks has been observed in organisations [102].

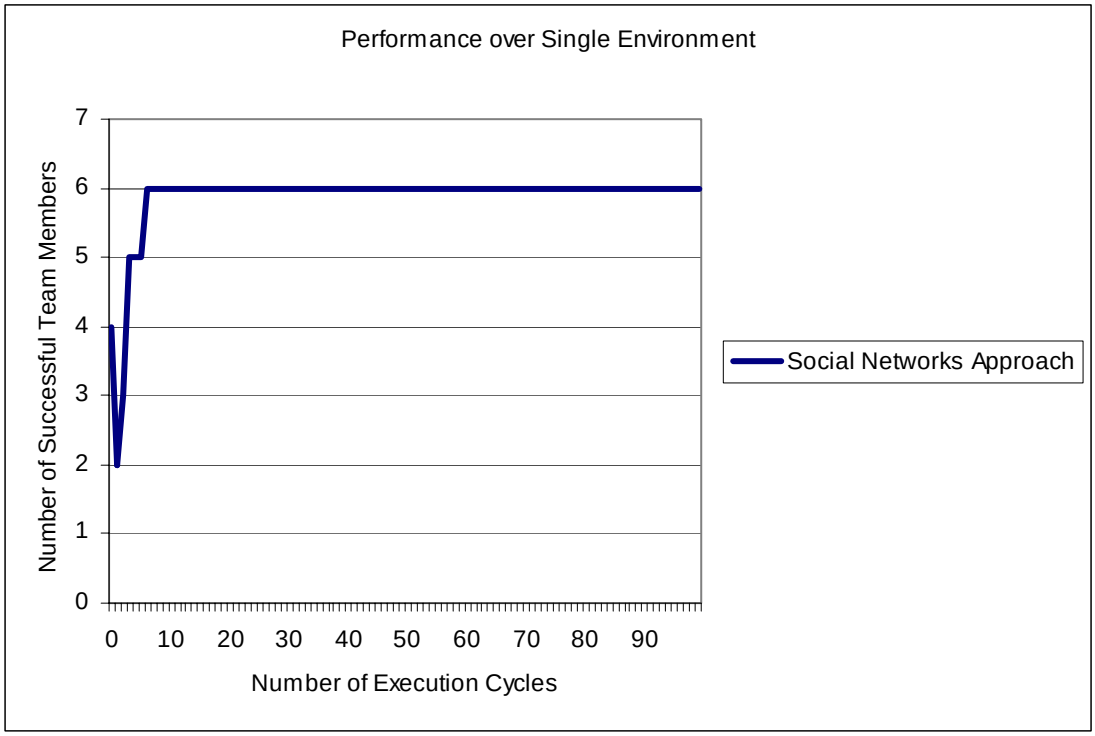

Figure 21. Observed improvement in performance (single environment)

7.5.4.2 Learning over Variable Environments

For this simulation, the influence of variable environment types on learning was investigated. As in previous experiment, uncertainty in this experiment is due to initial positioning. A summary of the simulation parameters is given in table 15.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	200
Environment Type	Variable
Random Initial Positioning	Yes
Foraging Team Size	6

Table 15. Simulation parameters used for the investigation of probabilistic selection influence

The results are presented in figure 22. Initially, after 100 execution cycles, the performance was not as good as for the previous experiment (refer to section 7.5.4.2). The number of execution cycles was then increased to 200, after which performance was approached the optimal performance.

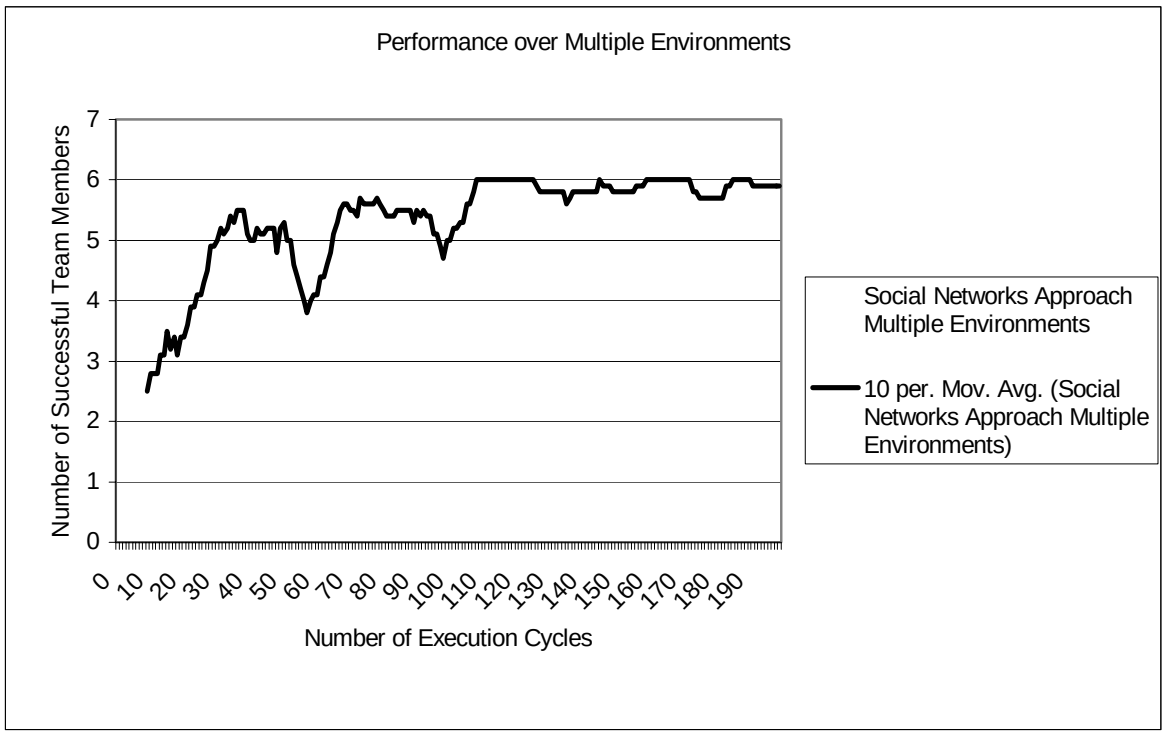

Figure 22. Observed improvement in performance (multiple environments)

Learning in multi-robot teams is a highly desirable feature due to the inherent challenges of real-world robotic applications. Some of the challenges facing real-world multi-robot teams are: uncertainty in sensing the environment (due to imperfect sensors), limited amount of historical performance (for training purposes) and difficulties in non-symbolic learning mechanisms, to name but a few. More on multi-robot team learning can be found in [143].

7.5.5 Agent Specialisation

The set of simulations presented in this section concentrate on the effect of social networks on agent specialisation. Specialisation is often observed in learning capable multi-robot architectures. For example, Balch reports that in such, learning capable, multi-robot team, individual robots automatically specialise in different roles in a team [13].

For the purpose of this paper, only one scout was allowed to assume the task of exploring the environment. The environment had the following attributes: TERRAIN_NORMAL, SHADED_AREAS, FOOD_FAR, FOOD_HEAVY. The success of the scout was measured by the ability of the scout to find the food area in a given number of steps. It is interesting to observe how various robots first attempted the scout role but in the end only one robot emerged as the best scout. Initially, any robot can be selected for either scouting or foraging tasks. If the robot is successful in the scouting task, the robot is kept as a scout. If a robot's performance in executing the scout task is less then an arbitrary threshold (defined as the ratio between the successful and attempted number of task executions), it is replaced by another robot. For the purpose of this thesis, the threshold was set to 0.8. The threshold can be interpreted as a minimum performance criterion. In other words, a scout with a success rate greater than 80% is an acceptable scout. The simulation executed for 100 execution cycles, although stability was reached after 48 execution cycles. A summary of the simulation parameters is given in table 16.

Simulation Parameters	Default Values
Execution Cycle	Scouting
Number of Execution Cycles	100
Environment Type	Variable
Random Initial Positioning	Yes
Foraging Team Size	N/A

Table 16. Simulation parameters used for the investigation of probabilistic selection influence

After 48 execution cycles, one of the agents, characterised by the attribute value vector (SMALL, LEGS, FAST, LONG RANGE, BATTERY, NO) (refer to table 7) emerged as the best scout. Table 17 illustrates the attribute values of the 24 agents that were considered for the scouting task. The first column represents the number of execution cycles (out of total of 100) that an agent was selected for the scouting task.

TRIES	LOAD	DRIVE	SPEED	DETECTION	POWER	FALSE FOOD
1	NONE	WHEEL	MEDIUM	LIGHT ONLY	SOLAR	NO
1	SMALL	TRACK	LOW	LONG RANGE	TETHERED	YES
1	NORMAL	LEGS	LOW	LONG RANGE	BATTERY	NO
3	NONE	TRACK	FAST	LONG RANGE	SOLAR	YES
1	NONE	LEGS	FAST	LONG RANGE	SOLAR	YES
1	NONE	WHEEL	FAST	LIGHT ONLY	SOLAR	NO
1	SMALL	WHEEL	FAST	NORMAL	TETHERED	YES
1	NONE	LEGS	FAST	LONG RANGE	SOLAR	NO
1	SMALL	WHEEL	LOW	NORMAL	TETHERED	YES
22	NORMAL	WHEEL	MEDIUM	NORMAL	BATTERY	YES
1	NONE	TRACK	FAST	LIGHT ONLY	SOLAR	NO
2	NONE	TRACK	FAST	LONG RANGE	SOLAR	NO
1	NORMAL	LEGS	LOW	LIGHT ONLY	TETHERED	NO
1	NONE	WHEEL	FAST	LIGHT ONLY	SOLAR	YES
1	NORMAL	WHEEL	FAST	NORMAL	BATTERY	NO
2	NORMAL	TRACK	LOW	LONG RANGE	SOLAR	NO
1	NONE	TRACK	MEDIUM	LONG RANGE	TETHERED	NO
1	SMALL	TRACK	MEDIUM	LIGHT ONLY	BATTERY	YES
1	NONE	TRACK	MEDIUM	NORMAL	BATTERY	YES
1	NORMAL	TRACK	LOW	LONG RANGE	TETHERED	NO
1	SMALL	LEGS	FAST	LIGHT ONLY	SOLAR	NO
1	NONE	WHEEL	MEDIUM	LONG RANGE	TETHERED	YES
1	NONE	TRACK	LOW	LONG RANGE	BATTERY	NO
52	SMALL	LEGS	FAST	LONG RANGE	BATTERY	NO

Table 17. Selected scout robot attributes

7.5.6 Influence of Kinship and Trust Parameter Values

The experiments that are presented in this section concentrate on the effects of changes in the ratio between kinship and trust. Three simulations were done with different parameter values for kinship and trust, while the rest of the simulation parameters were the same for all three simulations. A summary of the simulation parameters is given in table 18.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	700
Environment Type	Variable
Random Initial Positioning	Yes
Foraging Team Size	6

Table 18. Simulation parameters used for the investigation of probabilistic selection influence

The results of three sets of simulations are presented and discussed next.

7.5.6.1 Performance of The Model Using Only Kinship

For this simulation, only kinship was used to calculate the strength of social networks. Referring to equation 7.2, the value of parameter k is 1. The results are presented in figure 23. The result was somewhat surprising, as the social networks based approach still demonstrated the ability to learn by improving its performance over time. Trust was envisaged to be the main mechanism that stores historical performance, so how was it possible to improve performance over the time? The answer lies in the fact that the strength of kinship is calculated in relation to the scout. This means that as scout selection improves, the performance of the whole team improves. The rest of the team is now selected according to the kinship relationship to a scout that is better than previous scouts. A scout is better than another scout if its attributes are more suited to a variety of environments. By selecting the team members that share similar attributes (due to the strong kinship), the performance of team improves, however the maximum performance is not reached.

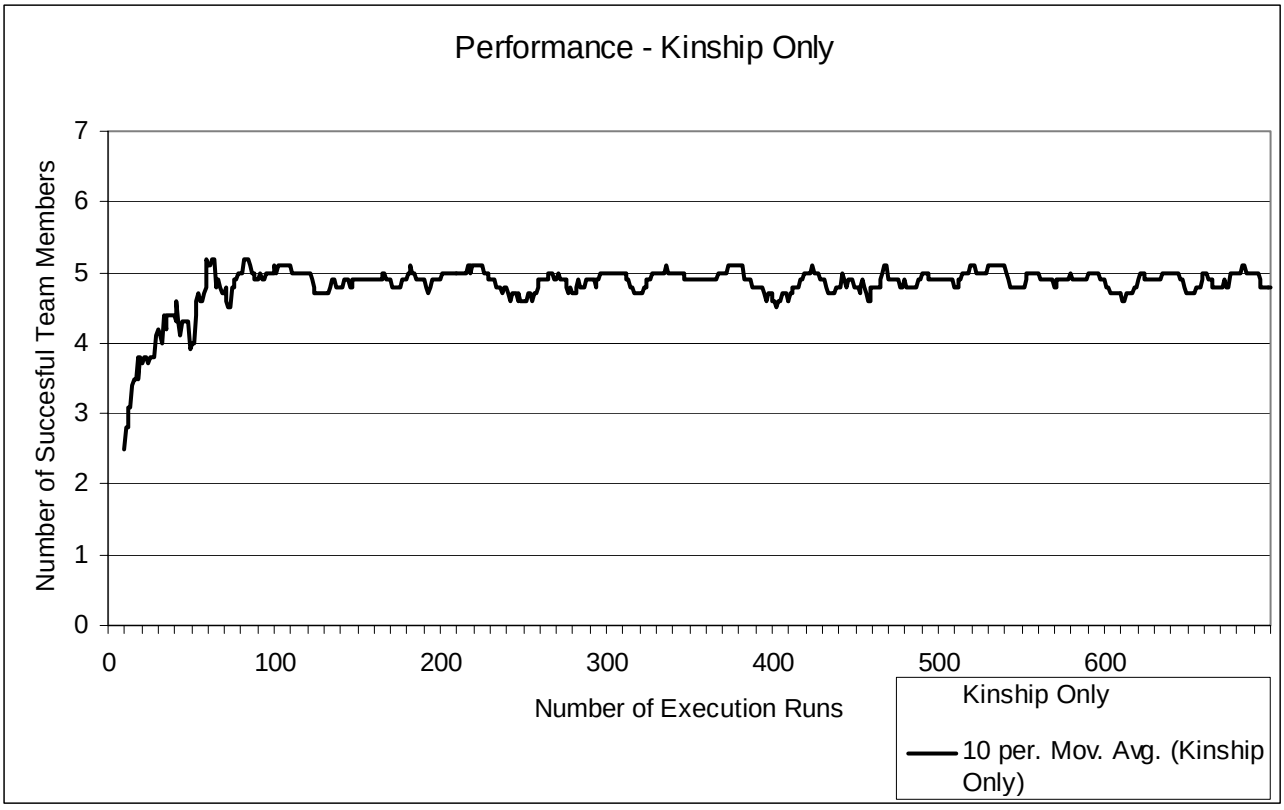

Figure 23. Performance of social networks based approach with only kinship relationship (700 execution cycles)

7.5.6.2 Performance of The Model Using Only Trust

For this experiment, only trust was used in the calculation of the strength of social networks. Referring to equation 7.2, the value of parameter k is 0.

The results are presented in figure 24. As for the previous experiment, the social networks based approach demonstrated the ability to learn by improving its performance over time. In the social networks based approach, the trust is the main mechanism that allows for storage of the historical data that reflects the past performance of team members. It is also important to note that maximum performance is also occasionally reached.

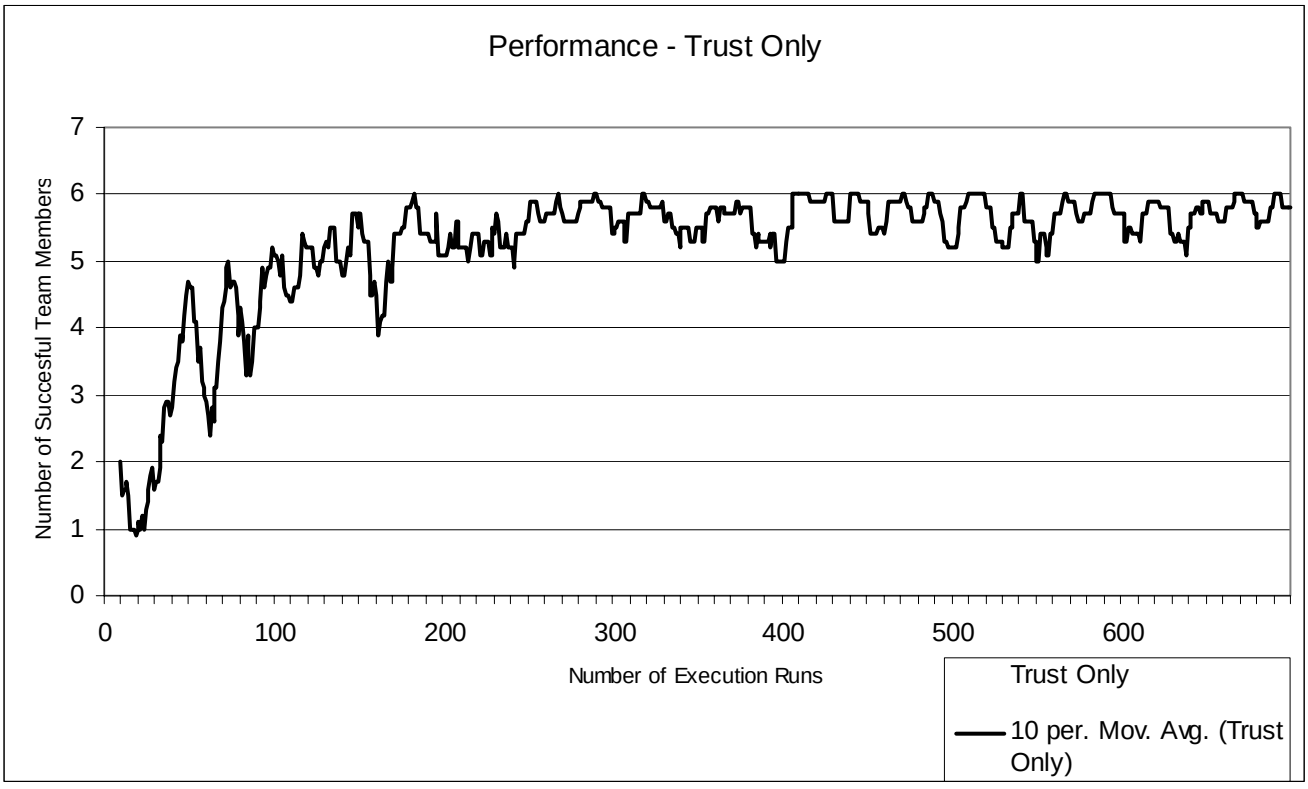

Figure 24. Performance of social networks based approach with one social relationship only – trust relationship (700 execution cycles)

7.5.6.3 Performance of The Model Using Trust and Kinship

In the last experiment that explores the effect of various parameter values to the performance of the social networks based approach the value of parameter k in equation 7.2 was set to 0.3. Various values ranging from 0.1 to 0.7, in increments of 0.1, have been tested and the best performance has been observed for the value of 0.3.

The results are presented in figure 25. The results were similar to the results from previous experiments. The ability of social networks to improve their performance over the time has been demonstrated in all experiments conducted for the purpose of this thesis.

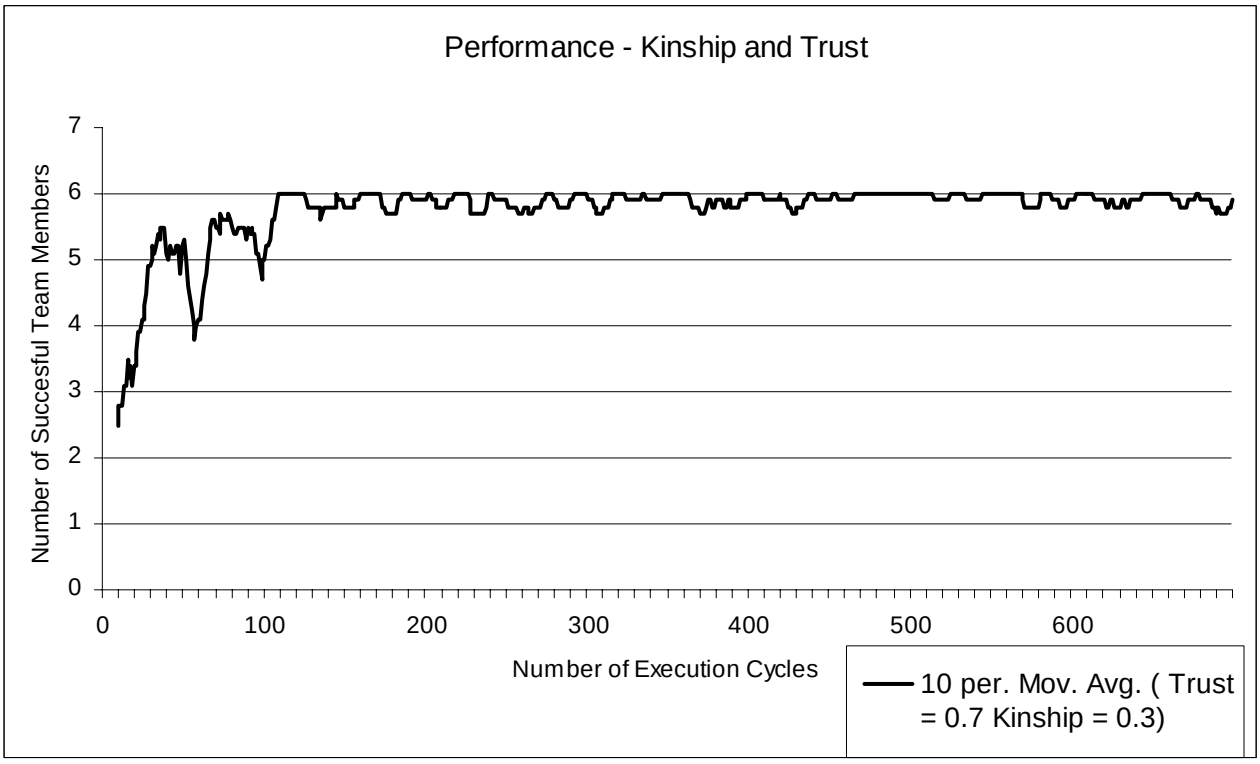

Figure 25. Performance of social networks based approach with one social relationship only – kinship relationship (700 execution cycles)

7.5.6.4 Discussion on Effects of Different Parameter Values

The social networks based approach to team allocation has demonstrated the ability to improve performance regardless of the value of the parameters that determine the

ratio between kinship and trust (refer to equation 7.2). The results presented in the previous sections do, however, show that the behaviour of the social networks based approach differs for different values of parameter k.

The results of experiments with varying value of k that were presented in sections 7.5.6.1 – 7.5.6.3 are combined in figure 26.

Initially, while there is no trust relationship (in other words, no historical performance data), the kinship relationship plays the predominant role in determining the strength of social networks. To illustrate this characteristic, the results of the first 100 execution cycles for all three values of k are given in table 19. A column represents number of successful executions per ten execution cycles. Maximum is 60, as there are six team members and ten execution cycles.

Execution Cycle	Successful Executions Kinship Only	Successful Executions Trust Only	Successful Executions Trust = 0.7 Kinship = 0.3
10	23	18	23
20	35	9	31
30	41	19	49
40	44	27	55
50	43	45	48
60	48	31	40
70	47	38	56
80	50	42	57
90	49	50	55
100	50	49	49

Table 19. Comparison of social networks over first 100 execution cycles

It is interesting to note that a social network that uses both trust and kinship relationship performs either as the best approach or the second best. In other words, the performance of the combined approach is never the worst. This characteristic is observed throughout the experiments, leading to conclusion that the combined trust and kinship approach is the safest, if not always the optimal, option.

As historical data based on previous execution cycles grows, the performance of all three social networks improve over time. This is illustrated in figure 26. It is interesting to note that it takes longer for a social network that uses only the trust relationship to achieve the same level of performance as a social network that uses

only the kinship relationship, and that which uses both. However, the performance increases over time to exceed that of the kinship only method. The combined approach, using kinship and trust out-performs the single relationships approaches, albeit only after more than 500 execution cycles. The performance summary for each of the approaches after 700 execution cycles is given in table 20.

Performance over 700 execution cycles	**Kinship Only**	**Trust Only**	**Kinship and Trust**
Total number of successful team members over 700 execution cycles (max 4200)	3370	3661	4011

Table 20. Comparison of social networks over 700 execution cycles

The results of the experiments presented in sections 7.5.6.1 – 7.5.6.3 are combined in figure 26.

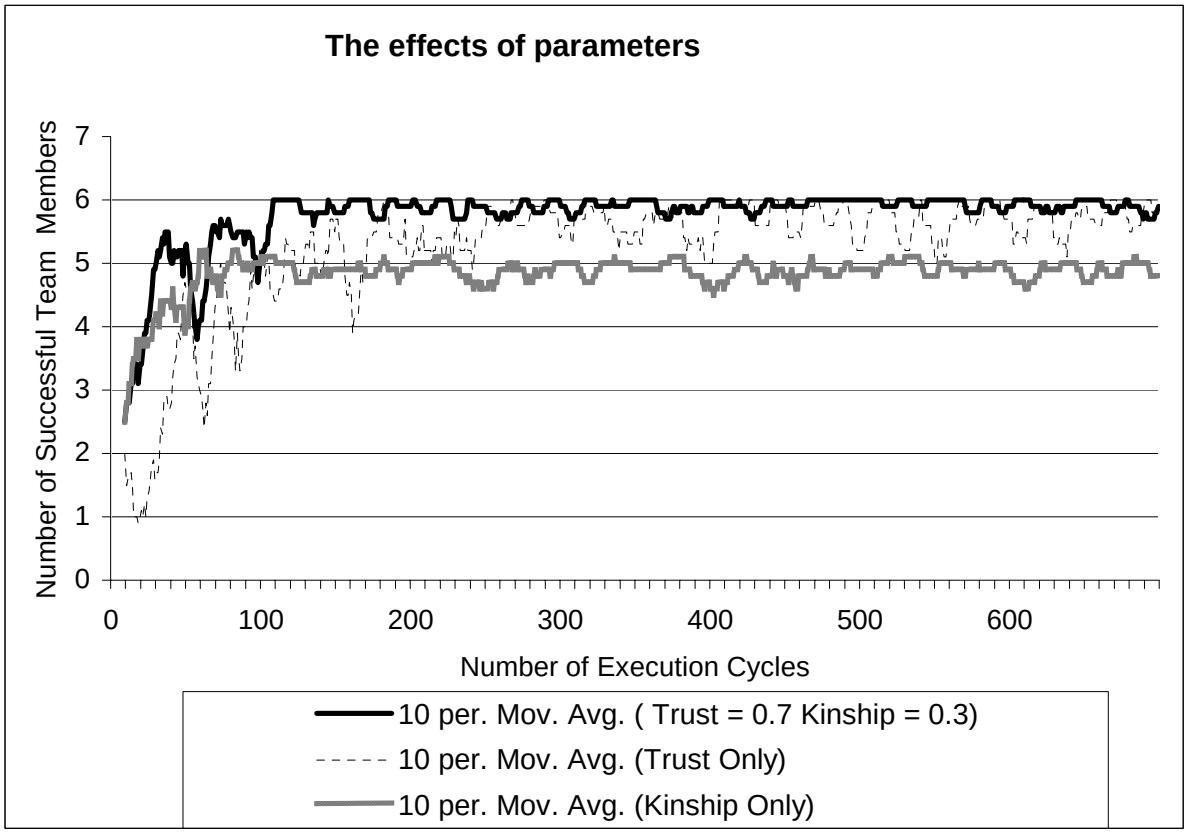

Figure 26. Performance of social networks based approach with one social relationship only – kinship relationship (700 execution cycles)

7.5.7 Evolution of Subgroups

The forming of subgroups within a social network has been observed in human societies. The forming of subgroups and their effects was studied by many researchers, mainly in the field of sociology. An overview of such research falls outside the scope of this thesis and the reader is referred to [195].

In relation to this thesis, the pertinent question is if the proposed social networks based approach indeed mimics real-life social networks as observed in higher mammalian societies. One of the characteristics of real-life social networks is the formation of social structures that are often referred to as subgroups. As noted by Collins [42], if a subgroup is tightly connected, it is referred to as a *clique*. Agents in a *clique* tend to exhibit homogenous beliefs and common characteristics.

In order to confirm that *cliques* do form between the agents in the social network based approach as presented in this thesis, this section investigates forming and evolution of such structures. In the context of an abstract simulated environment a *clique* is a subgroup of agents that are well-suited to a particular environment type. The investigation focuses on two main characteristics of a *clique* - the relative stability (once established, the members of a *clique* do not easily leave the *clique*) and the homogeneity between the members (in the context of the abstract simulator, the similarity between agents' attribute values).

The formation of *cliques* has been observed, regardless of the environment type. However, for the purpose of illustrating the process, the results presented in this section are related to one, randomly selected, environment type. The simulation used 100 execution cycles. The summary of simulation parameters is given in table 21.

Simulation Parameters	Default Values
Execution Cycle	Scouting, Foraging
Number of Execution Cycles	100
Environment Type	Single
Random Initial Positioning	Yes
Foraging Team Size	6

Table 21. Simulation parameters used for the investigation of evaluation of subgroups.

The first part of the execution cycle is the scouting task. Agent 33, defined by attribute vector (NORMAL, TRACK, FAST, LONG RANGE, TETHERED, NO), was tasked with scouting (refer to table 7 and section 7.5.5). Once the scouting task was executed, agent 33 has formed a foraging team, using the social networks based approach to team selection (refer to algorithm 10). The selected team consisted of agents 0, 12, 14, 18, 19 and 33 (refer to table 7).

During the execution of the foraging task, which is the second part of an execution cycle, four agents have completed the allocated foraging task. The graph presented in figure 27 shows the social network after the first execution cycle. Note that in order to keep the graph relatively uncluttered, the only social relationship illustrated is the trust relationship in relation to the team leader, agent 33. Nevertheless, the forming of a *clique* is clearly illustrated and the final state of the social network is given later in this section. The indices represented as a full line illustrate the current clique members, while indices represented as a dashed line indicate the agents that were considered and rejected as the clique members. The indices are weighted and their weight is given as a ratio between successful task executions and attempted task executions. The selected team leader is indicated by the capital letter T in parentheses (T).

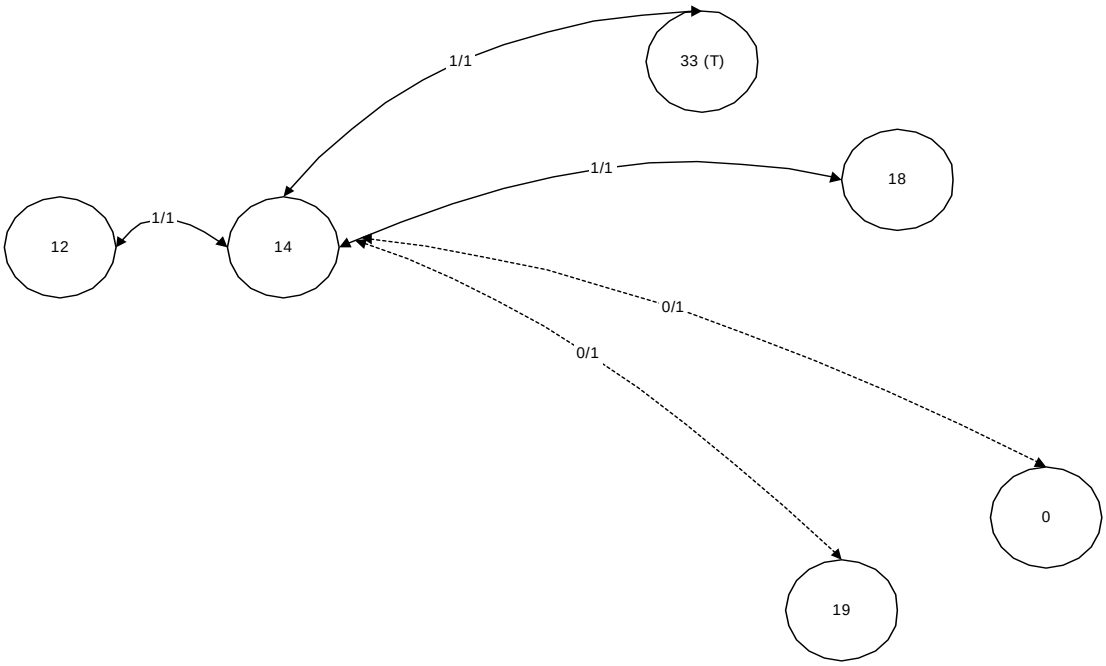

Figure 27. The sociogram after first execution cycle

Figure 28 represents the social network after the second execution cycles. During the second execution cycle, only two agents have completed the task (agents 14 and 12). The team leader, agent 33, has also failed to complete the task due to the range (agent 33 has TETHERED as attribute value of attribute POWER and according to the rules of interaction with its environment it has a limited range – refer to section 7.1). It is important to note that agents 0 and 19 have failed to complete the task due to their kinship to the scouting agent (TETHERED, refer to table 7 for agent attributes), since the scouting agent is selected as a team leader due to the initial lack of trust data. To illustrate, consider the initial state where there is no trust relationships established between the agents. The only agent with a historical performance record (agent's own trust in its suitability to perform a task in a given environment) is the scouting agent. For the kinship calculation purposes, it is considered the team leader and agents are ranked according to their kinship related to the team leader (see section 6.6.4.2).

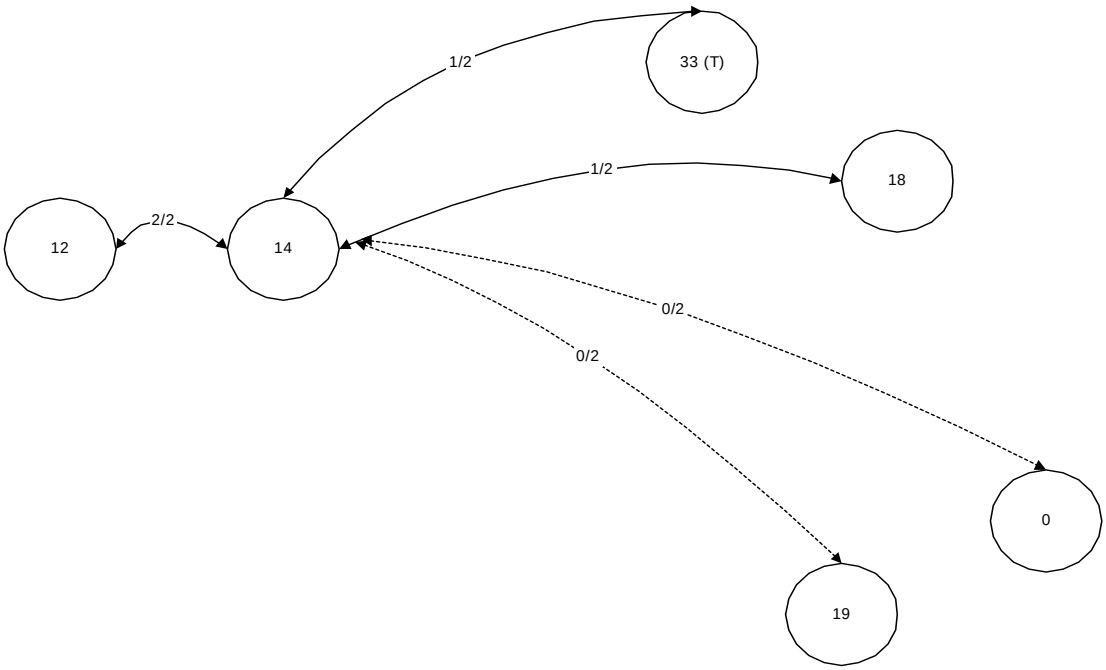

Figure 28. The sociogram after the second execution cycle

The agent's performance is considered unsatisfactory if the ratio between successful task executions and attempted task executions is less than 0.5 (a problem dependent parameter). After the second execution cycle, the performance of agent 33 that performed the roles of a scouting agent and team leader was deemed unsatisfactory. The performance of agents 0 and 19, was also deemed unsatisfactory. Agent 34 became the next scouting agent and team leader. Agents 0 and 19 were replaced by agents 49 and 23 which, were the next best ranked agents in relation to the new team leader. The foraging task was executed and agents 12, 14 and 23 have successfully completed the task. Agent 34, which was selected as the team leader, has failed to complete the task and it was replaced in the next execution cycle. Figure 29 illustrates the social network after the third execution cycle.

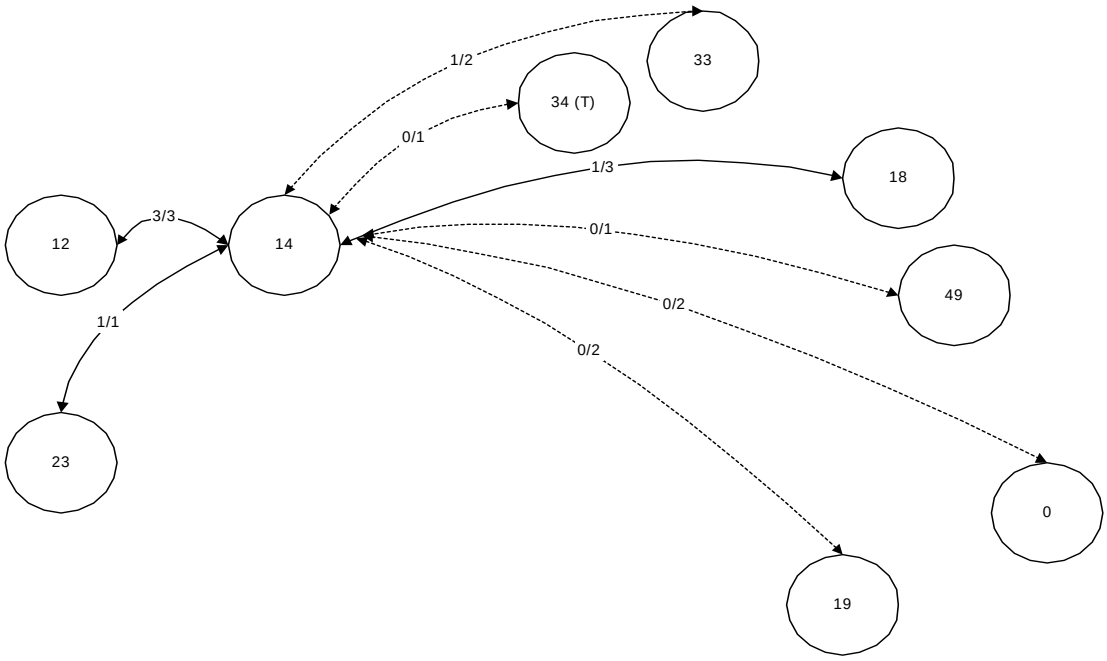

Figure 29. The sociogram after third execution cycle

It is important to note that after the third execution cycle there was sufficient historical information about previous performance related to the foraging task to enable agent 14 to become the next team leader.

As the performance of agents 18, 33 and 49 was not satisfactory, they were replaced. The next execution cycle illustrates the importance of kinship, as kinship is now calculated in relation to the new team leader. The new permanent members 9, 20, and 32, that have strong kinship relationships to the new team leader, were introduced as team members during the fourth execution cycle. The social network after the fourth execution cycle is illustrated in figure 30.

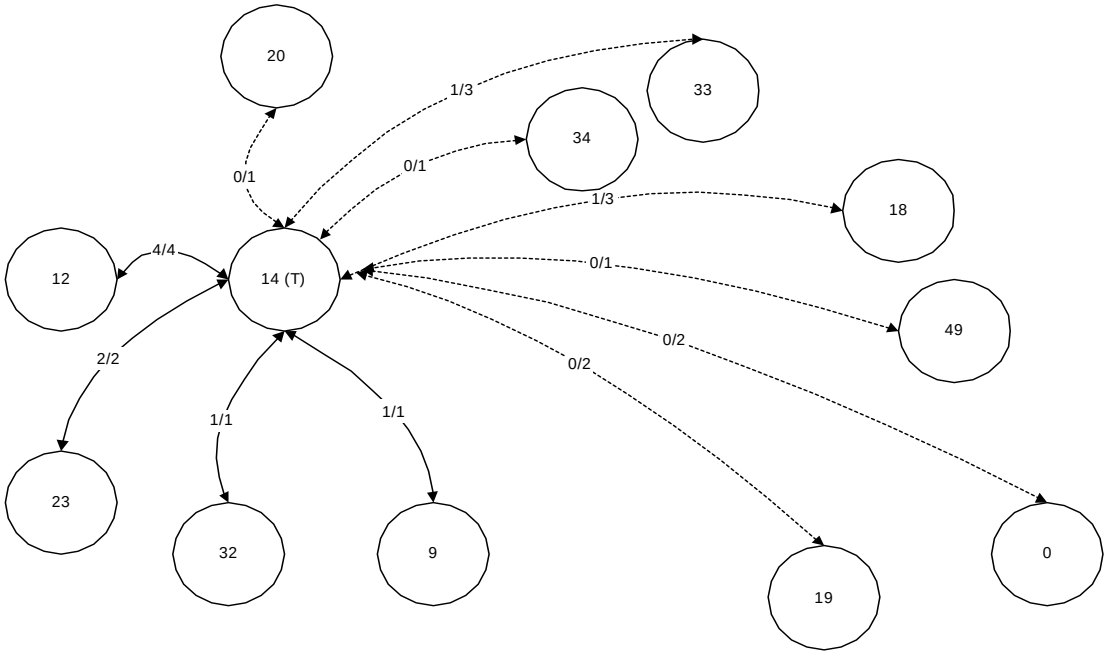

Figure 30. The sociogram after fourth execution cycle

During the fourth execution cycle all agents, with the exception of agent 20, have completed the task. Agents 12 and 14 were successful in all execution cycles so far and they can be viewed as the core of a clique. Agent 20 was replaced by a new team member, agent 43. The resulting sociogram is presented in figure 31.

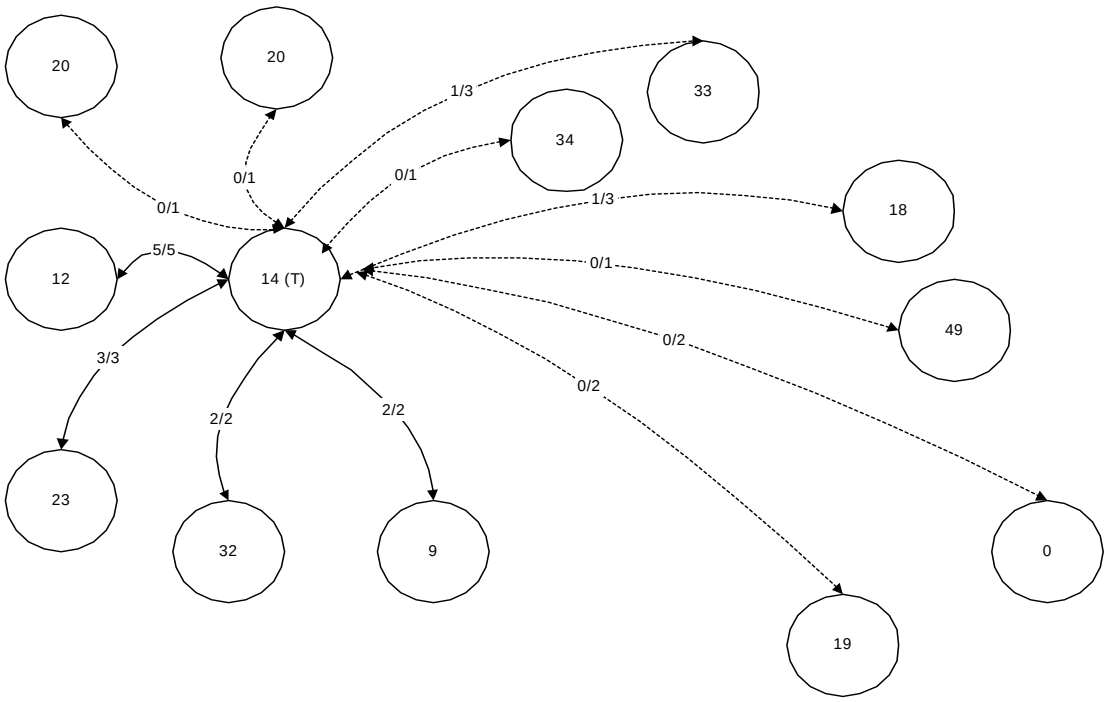

Figure 31. The sociogram after fifth execution cycle

It is important to note that after the fifth execution cycle, resilience (stability) of a clique has already been identified. Once part of the clique, members (agents 12, 23, 32 and 9) do not easily leave. The team member introduced in the fifth execution cycle, agent 43 was the only unsuccessful team member agent and agent 25 replaced it. The sixth execution cycle again further enforced the clique. The new team member, agent 25 was yet another unsuccessful team member, while all other agents have completed the task. The social network after sixth execution cycle is illustrated in figure 32.

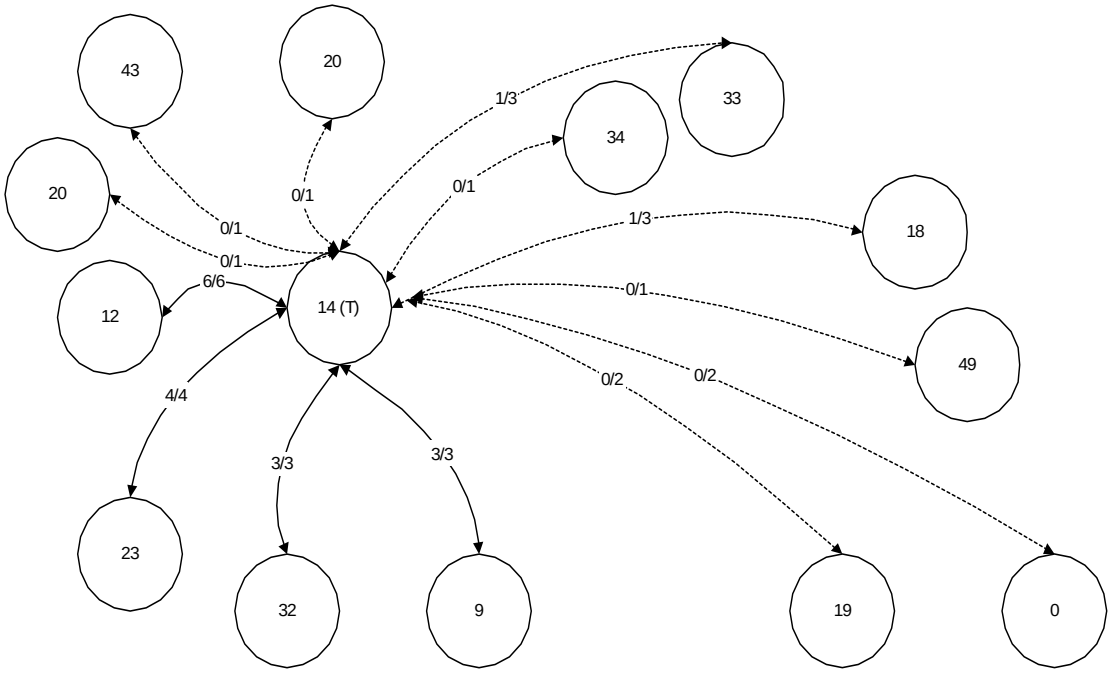

Figure 32. The sociogram after sixth execution cycle – established clique

Finally, the seventh execution cycle has introduced the sixth and final member, agent 15, of the clique around the team leader, namely agent 14. During the execution of the seventh execution cycle and subsequent execution cycles (the simulation executed 100 execution cycles) all team members were successful in task execution. The social network after the seventh execution cycle is presented in figure 33.

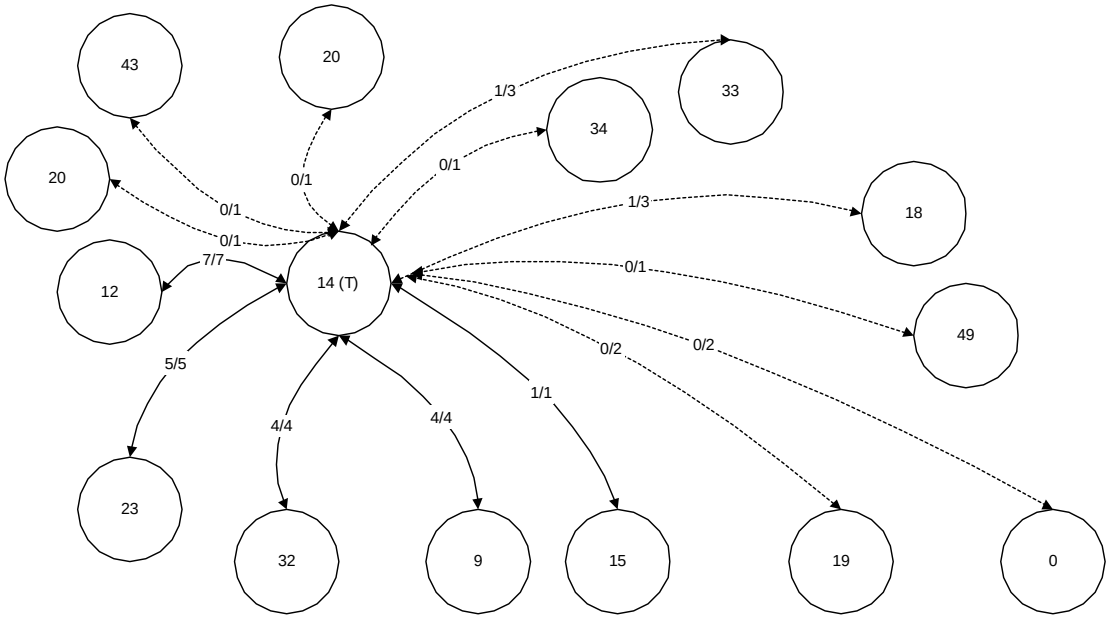

Figure 33. The sociogram after seventh execution cycle - stable clique

After the seventh execution cycle, the clique, consisting of agents 9, 12, 14, 15, 23 and 32, has reached stability and no further fluctuations were observed. Stability was achieved relatively early. Similar results were obtained in further experiments. The final state of the social network after 100 execution cycles is presented in figure 34.

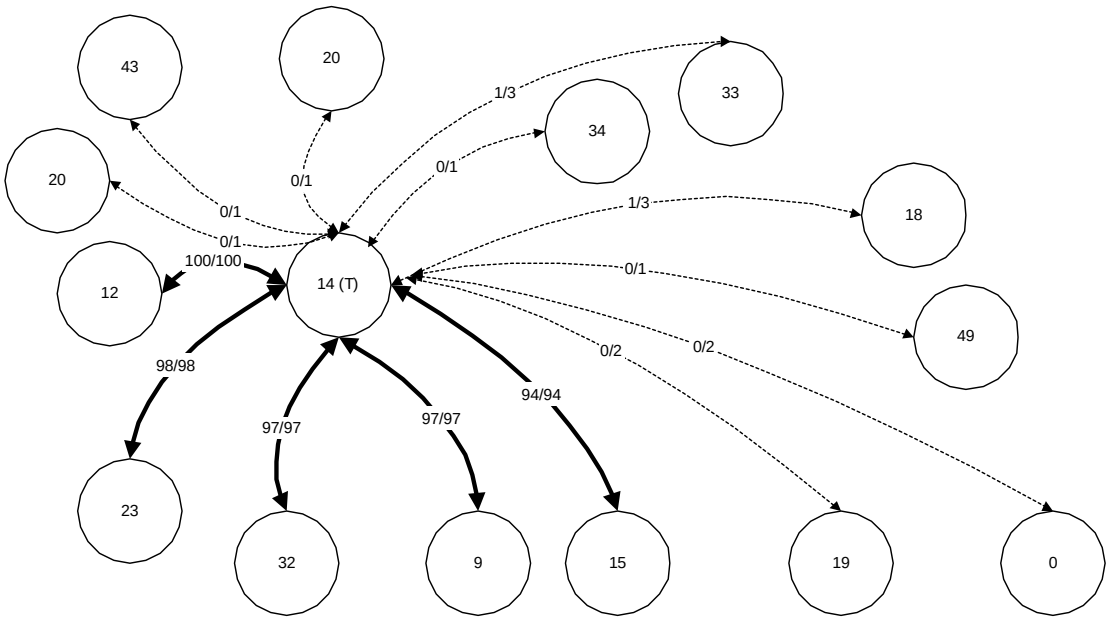

Figure 34. The final social network after 100 execution cycles

The investigation has confirmed that *cliques* do form in an artificial agent society, such as a society of agents in an abstract simulated environment. Firstly, the stability of a *clique* was investigated by observing the fluctuation of members over the period of execution. The *clique* has reached stability relatively early, after 6 execution cycles.

7.6 Summary

A partial implementation of the INDABA architecture, consisting of the upper two layers, was presented in this chapter. The agents in the abstract simulated environment were implemented using this reduced INDABA architecture. The social networks based approach was used as the main coordination mechanism and it was compared to an auctioning mechanism. A number of experiments were conducted that investigated all the parameters of the social networks based approach. Characteristics of the social network based approach were discussed, namely learning capability, specialisation of agents and the formation of *cliques*.

The next chapter presents a more comprehensive INDABA architecture implementation, again utilising the social networks based approach as a coordination mechanism.