Gossip-Based Self-organising Open Agent Societies

Sharmila Savarimuthu, Martin Purvis, Bastin Tony Roy Savarimuthu, and Maryam Purvis

Department of Information Science, University of Otago, Dunedin, New Zealand
{sharmilas,mpurvis,tonyr,tehrany}@infoscience.otago.ac.nz

Abstract. The objective of this work is to demonstrate how cooperative sharers and uncooperative free riders can be placed in different groups of an electronic society in a decentralized manner. We have simulated an agent-based open and decentralized P2P system which self-organises itself into different groups to avoid cooperative sharers being exploited by uncooperative free riders. This approach encourages sharers to move to better groups and restricts free riders into those groups of sharers without needing centralized control. Our approach is suitable for current P2P systems that are open and distributed. Gossip is used as a social mechanism for information sharing which facilitates the formation of groups. Using multi-agent based simulations we demonstrate how the adaptive behaviour of agents lead to self-organization.

Keywords: Self-organising systems, Gossip; Multi-agent Based Simulation, Cooperation, Sharing behavior, Peer-to-Peer, Artificial Societies.

1 Introduction

One of the most common problems in P2P networks is free riding [5, 17]. In our context, free riders are those agents that do not contribute to the collective goals of the networked society, but make use of the resources of the network [17]. These free riders decrease the overall performance of the society by degrading the common good [5].

Electronic societies suffer from these free riders who exploit the common resources (e.g. bandwidth in a file sharing system). Many existing approaches employ centralized social regulations to control free riders. Researchers have used monitoring agents or governor agents to control agent behaviour [7]. But these centralized mechanisms are computationally expensive for a system. Centralized mechanisms are known to cause performance bottlenecks and also suffer from scalability issues [17].

With the increase in processing power and storage capacity of low-cost, lightweight computing devices such as smart phones, the arena of computing is becoming much more distributed. The clients of file sharing systems are not only personal computers but also smaller devices such as smart phones. There is a need for decentralized solutions to deal with the free riders. Additionally, the openness of the Internet allows users to dynamically join and leave the system at any point of time. So, a solution to the free-riding problem should take into account the open, dynamic and distributed nature of modern software systems.

N. Desai, A. Liu, and M. Winikoff (Eds.): PRIMA 2010, LNAI 7057, pp. 105–120, 2012.

To that extent, this paper proposes a decentralized solution that makes use of social mechanisms such as gossip [10] and ostracism [11]. The inspiration to use social mechanisms for our work comes from the human societies, which have evolved over millennia to work effectively in groups. For human beings group mechanisms provide social machinery that supports cooperation and collaboration. Social control is a fundamental concept that has evolved in human societies. Social control can be employed through leadership mechanisms. For example, the leader can impose rules on his followers. The disadvantage of this approach is that it is centralized. On the other hand, social control can be achieved using a bottom up approach.

For example, a gossip-based mechanism can be used to achieve social control as it serves as a distributed referral mechanism where information about a person or a group is spread informally among the agents. This approach can be used to achieve control in agent groups. Another social mechanism that can be employed to deal with free riders is ostracism. Members that do not adhere to the values or expectations of the groups can be sanctioned by the other agents by their refusal to interact with those agents.

In this work we demonstrate how these social mechanisms can be employed in an open, dynamic and decentralized society where several groups are formed and are ranked based on their performance.

The remainder of the paper is organised as follows. The social concepts used in this work are introduced in Section 2. Our experimental setting and selected experimental results are described in Sections 3 and 4. In Section 5 we present the related work and the comparison with our previous work. Finally, Section 6 concludes the paper.

2 Modeling Social Dilemma between Sharing and Non-sharing

Our experimental model presents a social situation in which the agents have the option to share or not to share. Sharing would cost the donor who shares. But the receiver receives the benefit without incurring any cost. Non-sharing (defection) is the selfish option which benefits the individual but is not good for the society. Sharing benefits the society by improving the performance of the whole system, which leads to the overall betterment of the society. Since the donating agent spends some effort (e.g. bandwidth) in the process of donating, it incurs some cost in our model. That sharing agent could have decided to be selfish and thereby avoid incurring that cost. Thus free riding becomes a threat to the society, causing damage to the common good. This is the issue of the "Tragedy of the Commons" [5]. A brief overview of the social mechanisms used in our experiments to deal with free riding are described below.

2.1 Gossip

Gossip is a powerful social mechanism found in human societies for information sharing. Gossip is a public opinion which leads to the benefit of a social group [10].

According to research done by evolutionary biologists humans have shown more interest in gossip than the truth [16]. The research has shown that gossip is more powerful than the truth in human societies when the participants were presented with both types of information (the gossip information and the real information).

They note that "gossip has a strong influence even when participants have access to the original information as well as gossip about the same information" and also have noted that "gossip has a strong manipulative potential". There are other examples of agent based simulation and P2P systems [4, 6] which have used a gossip based protocol [3]. Gossip can be considered to be a distributed referral mechanism.

2.2 Ostracism

It has long been a feature of human and animal societies that the member of a group who do not abide by rules or norms can be punished by other members of the group (the followers of the rule/norm). One kind of punishment is ostracism [11]. Other members will stop interacting with the member who is being ostracized and don't consider that person as a part of their group by ignoring or refusing to interact. This social sanctioning mechanism works without a centralized control or authority.

3 Experimental Setup

In our experimental arrangement agents are engaged in the sharing of digital goods in a P2P environment of a simulated artificial agent society. The system is developed as a distributed system without central control.

3.1 3.1 Agent Attributes

For this experimental model we have used the agents which have fixed, randomly assigned attribute values which represent how they behave.

- **Cooperativeness value:** This attribute concerns how cooperative an agent is. Agents have a randomly assigned cooperation value between 0 and 10 that represents how much they cooperate (share), with 0 representing an agent that never cooperates and 10 representing an agent that cooperates every time. This value is known as the cooperativeness of the agent.
- **Tolerance value:** Agents have a tolerance value between 1 and 10, which characterizes how much non-cooperation the agent can tolerate before it decides to leave the group. A value of 1 identifies the least tolerant agent, and 10 identifies the most tolerant agent.
- **Rejection limit:** Rejection limit represents how many rejections the agent can face before it decides to leave for another group.
- **Gossip blackboard length:** Each agent has a gossip blackboard of certain length to store the gossip messages from other agents of its group. Each agent also has a memory of certain number of previous groups to which it belonged.

- **Life span:** Agents are set to have life spans, which determine how long the agents remain in the society (i.e. die). When an agent's life span is over it leaves the society.
- **Cost and benefit for sharing:** A sharing agent loses 0.1 as cost for sharing and the receiving agent receives 1 as benefit.

3.2 Experimental Parameters

In the initial setup agents are put into random groups. Each group can be imagined to be represented by a tag (badge). Agents within a group have the same tag. They interact within their group, and they can also move to other groups under certain conditions. In such cases they join the other, jumped-to group, and the tag changes accordingly. Agents can ask for gossip feedback about other agents' behavior. Groups are formed or dismantled based on their size. The procedure of the experiment is explained in the following sections. The experimental parameters are listed in Table 1.

Table 1. Experimental parameters

Experimental parameters	Values
Number of agents to start with	100
Number of groups to start with	5
Number of iterations	5000
Agent's cooperative value:	0-10 (random)
Agent's tolerance value:	1-10 (random)
Agent's rejection limit	10
Agent's gossip blackboard length	10
Agents group memory limit	4
Agent's lifespan	Varies
Number of gossip feedbacks	5
Group's size for dismantling	5
Group's size for splitting	40
Cost for sharing	-0.1
Benefit for receiving	1

The procedure of the experiment is explained below.

3.3 Publishing Gossip

In each iteration, a certain number of random players (agents) may ask for files from other players of their group. A player can gossip about the outcome of an interaction with another agent (random) in its group (report whether the other agent was cooperative or not). In this gossip mechanism we assume that there is no lying. Since this happens within the group (agents in a group have same tags), we have assumed that the agent has no motivation to lie. In this fashion, every transaction is reported

(gossiped about) to one of the other agents in the group. Thus the overall system has some partial information about the cooperativeness of each agent, maintained in a distributed way. For further illustration, the operation of how peers publish gossip is explained in the following example. Consider A, B and C as the three random agents in a group. A is the taking-player, B is the giving-player and C is the gossip holder. A asks for a file from B. If B shares then A gossips positively about him to C, otherwise A gossips negatively about him to C.

3.4 Using Gossip

Each peer has a limited amount of memory space for storing new gossip information. After reaching the storage limit, the memory register rolls over, based on a First-In-First-Out (FIFO) algorithm. When a player requests a file, the giving-player can check with a certain number of (e.g. five) other random agents (asking them what they know from the gossip information they have received) whether this taking-player is the worst cooperator of their group. The worst player is the one who has been uncooperative most times in its group (according to the available gossip information). If the taking-player is the worst player, the giving-player refuses to interact with the taking-player (ostracism). Otherwise this giving-player interacts (sharing a file or not based on its own cooperativeness). The operation of how peers use gossip is explained by the example given below.

Assume C and D are the players in the group where C is the taking-player, D is the giving-player. D checks with five other players in the group in order to see whether C is the worst player in their group. If so D refuses to play with (share file with) C. Thus C is ostracized. Otherwise D plays with C. When only a few agents (less than five) have gossip about the taking-player, then only the available information is taken into consideration. Sometimes it can be the case that none of the players have gossip about the taking-player. In such a case the taking-player is not considered to be the worst player, a privilege similar to what happens when a new player joins a group. By this process agents share file taking gossip into consideration which is about other agents' past behaviour.

3.5 Leaving a Group

An agent can leave a group for two reasons. A player can leave a group if its tolerance level is surpassed or its rejection level is surpassed. We call this leaving agent a "hopping peer". If its tolerance limit is reached, that means this agent is in a group where others do not cooperate at the rate that meets this agent's minimum level of expectation. Thus after a number of such non-sharing events from the group members (the agent's tolerance limit is surpassed) the agent will decide to leave that group and move to another group. If its rejection limit is reached, that means this agent is in a group where it is considered to be the worst cooperator by some other agents so it has been refused a play more often than others. If the rejection level is met then the agent will leave that group and move to another group.

3.6 Choosing a New Group to Join

When an agent decides to leave a group and join another, it looks for a group that may accept it. Agents can apply to enter into other groups they choose but they get entry into a group which matches its cooperativeness. A good agent would get into a group that is better than its current group while a bad agent should get into a group that is worse than its current group. This process is explained in detail in [9]. We have restated it in the following paragraphs.

The hopping peer collects information about other groups from their group members. Then it decides to which group to request admission from. Every agent has a memory record of its most recent groups (in our experiments the memory limit was set to 4). For example, assume agent E has been in 3 other groups before, as shown below in Table 2.

Table 2. Previous group history

Group No	Iteration No	Cooperativeness
1	560	4.5
3	700	6.0
2	1200	6.4

Table 3. Latest available information

Group No	Iteration No	Cooperativeness
5	1330	8.1
3	1170	7.5
2	1200	6.4
1	1199	3.8

The first row of Table 2 shows that E has left group 1 at the 560th iteration, and the cooperation value of that group was 4.5 at that time. E left group 3 at the 700th iteration and group 2 at 1200th iteration. Since the composition of groups change over time, the cooperativeness of the group also changes. So it is likely that the most recent information will be the most accurate and useful for an agent. Since all agents have a memory of their previous groups, the hopping peer can collect this information from all its group members and calculates the latest information about other groups. In particular, the agents who moved into this group recently from other groups have the most recent information. Taking into consideration this information, the agent decides where to move. For example assuming the current iteration is 1400, the latest information collected from the group members is given in Table 3.

Assume here that agent L intends leaving group 4, and Group 4's cooperativeness is 6.6 at that moment. From the latest information agent L knows about other groups and their cooperation value. For agent L, groups 5 and 3 are better, since the cooperation value in those groups appear to be higher than L's current group. Groups

2 and 1 are lower-ranked groups. So agent L chooses to move to the groups in the order of their ranking.

If L is intolerant of its current group (which means it is not happy about the cooperativeness of its current group), it will try to enter into the best group that it can find. This is the case of an agent being "too good" for its current group and wanting to move to a more cooperative group. But if the better groups on its list does not allow entry, then the intolerant agent L may determine that there is no group available that is better than its current group, and it will remain in its current group. In this case its tolerance limit is reset to 0.

On the other hand, an agent may not be good enough for its current group i.e. it is being shunned by the other members for being the worst member of its group. Because of refusals from other agents to play, its wealth will not increase, and it will want to leave and find some another group in which it can find players to play with. If the better groups do not allow entry, the agent will go to lower groups, since it is better off moving to any new group (even if it is a lower group) rather than staying in the current group where it is known as the worst player. How a player gets entry to another group is explained in the following section.

3.7 Joining Another Group

The hopping peer asks any randomly chosen agent in the group to which it seeks entry for its permission to enter. We call this permission-granting agent in the group to which entry is sought, the "checking peer". The checking peer will accept an agent whose cooperativeness value is greater than or equal to a value calculated by a formula (given below). This formula is the same one used in our previous work [9]. This hopping peer will gain permission to enter the group whenever its cooperativeness is greater or equal to the group's entry value calculated by the following formula:

$$EV = AC - (C1 / (SL - S)\, C2) + C3(S\text{-}SU) \quad (1)$$

The group Entry Value (EV) is calculated considering the given group's Average Cooperativeness (AC) and its group Size (S). AC is the average cooperativeness of the group calculated through the gossip mechanism, and S is the size of the group. C1, C2, C3 are constants whose values in our experiments are 25, 2, 10, respectively. These constants were adjusted to make the EV expression appropriate for two\``boundary values", the upper size limit of a group (SU) and the lower size limit of a group (SL). It is inappropriate or inefficient for groups of players or traders to become too big or too small. In our experiments, SU was set to be 25, and SL was set to be 10. That means if the size of the group is 10 or below the entry qualification value is set at a low value, making entry into the group very easy to obtain. If the size is 25 or above the entry qualification value is set to a high value and that would make it difficult for any but the most cooperative agents to join. Any values of the EV expression that fall below 0 are set to 0, and entry values above 10 are set to 10. Thus

a group's entry value is always between 0 and 10. A simple example illustrates the use of this formula.

Consider that a group's calculated cooperativeness (AC) is 6. When the group Size (S) is 14 the group Entry Value (EV) is 4.43. When the group Size (S) is 25 the Group Entry Value (EV) is 6.88. In our system, the checking peer needs to get an estimate of the cooperativeness of the hopping peer (the agent seeking entry). So the checking peer asks 5 randomly chosen players from the hopping peer's group about the hopping peer's cooperation. It is thus inquiring into gossip information from the hopping peer's group. Consider a case where E and F are in different groups. E is the checking peer, and F is the hopping peer that wants to enter E's group. F asks E for entry, and E asks 5 other randomly chosen players in F's group for gossip information about F's cooperativeness. The averaged value is calculated (out of 10) from this information considering the worst case scenario. This estimated cooperativeness would be a value between 0 and 10. If F's estimated cooperativeness calculated through this gossip information is greater than or equal to the entry value (EV) of its group, the checking peer allows entry for the hopping player; otherwise it denies entry. In that case the hopping peer will try to enter into other groups. The hopping peer will ultimately get into a group where its cooperativeness meets the eligibility criteria to enter. If no such group is available, the hopping peer stays in its current group.

The entire process is repeated for many iterations, and gradually, some groups will emerge as elite groups with many cooperators, and other groups will have less cooperative players. As a consequence, these mechanisms achieve a separation of groups based on performance.

3.8 Groups Splitting and Dismantling

Our aim has been to develop a self-organizing open and dynamic system, where new agents may come into the society and also agents may leave the society at any time. To start with, new peers are allowed to join the society by gaining entry into random groups in the society. They can build their way up to higher groups based on their cooperativeness. A truly open and dynamic system will allow the formation of new groups and dismantling of existing groups according to the population size. Our aim was to achieve the same in a decentralized manner without explicit control at the top level. Forming groups using tags is helpful, since it is scalable and robust [4].

The agents' lifespan determines how long the agents remain in the society and when they leave (i.e. "die"). At any time a new agent could join the society and an existing agent could leave when its lifespan is over.

Since the number of agents in the society at any time is dynamic the system adapts itself to form new groups if more agents join. It also dismantles groups if there are fewer agents in the society (less than the lower size limit of a group).

The motivation for splitting and dismantling comes from real life societies. For example, when the size of a group becomes too large, it becomes unmanageable. Larger hunter-gatherer groups split because of reasons such as seasonal change or inequality in resource sharing (e.g. when meat is not shared equally).

In our approach, a group splits into two if the size of group reaches a certain limit (40). Based on the local gossip information in the splitting group, the top cooperators (first half) form one group and the rest (second half) form the other group.

If the size of the group decreases and goes below a certain limit (5) then the group dismantles. The remaining agents in the group go to random groups where they could enter. This is similar to a society where it can be functional only if the society has a certain size. For example, in hunter-gatherer societies, in order to hunt larger preys a group has to have a minimum size. Otherwise, the prey cannot be hunted. The same holds in the context of playing a sport. For example, a team playing volleyball has to have six players. Otherwise, the team cannot exist.

It should be noted that the splitting and dismantling functionalities account for the scalability of the system and its robustness.

4 Results

Before we present the experiments we have conducted and the results obtained, we would like turn the attention of the reader to the work reported in [9] where the results of the closed society are presented. In this work, there were 5 groups. The total number of agents in the society was 100. The work shows how the agents self-organise themselves into these groups based on their cooperativeness values [12].

4.1 Experiment 1 – Self-organization in an Open Society

We have conducted experiments on an open system by varying the arrival and departure rate of the agents. For all the experiments presented in this paper we start with 100 agents in 5 groups initially. After that agents can join (new arrivals) or leave (if life span is over) the society.

Figure 1 shows two graphs which share the same x-axis. The x-axis shows the number of iterations. In the top graph y-axis shows the cooperativeness of groups. Each diamond shown in the graph represents the cooperativeness of a particular group. For a given iteration number in the x-axis, the y-axis shows the cooperativeness of all the groups that were present in that iteration. For example, in iteration 100, there were 6 groups (represented by diamonds), with different levels of cooperativeness. The graph given in the bottom of Figure 1 shows the total number of agents (alive agents) in the society for a given iteration. For example, in iteration 100 there were 130 agents in the society.

These two graphs together show the dynamic behavior of the system (the formation of new groups and dismantling of old groups). It can be observed that, at the start the groups had an average cooperativeness value of 5. As the number of agents increased, new groups were formed (iteration 100). As the number of agents decreased (iteration 200), the number of groups decreased. The separation between the good groups and the bad groups is distinct. When the total number of agents was about 40 in iteration 300, there were fewer groups. Note that the cooperativeness of these groups was about 5 at that point. As the number of agents in the societies then increased, there were more groups and the separation between the good and the bad groups is evident. We note this process can be appreciated better by viewing the video shown in link [13].

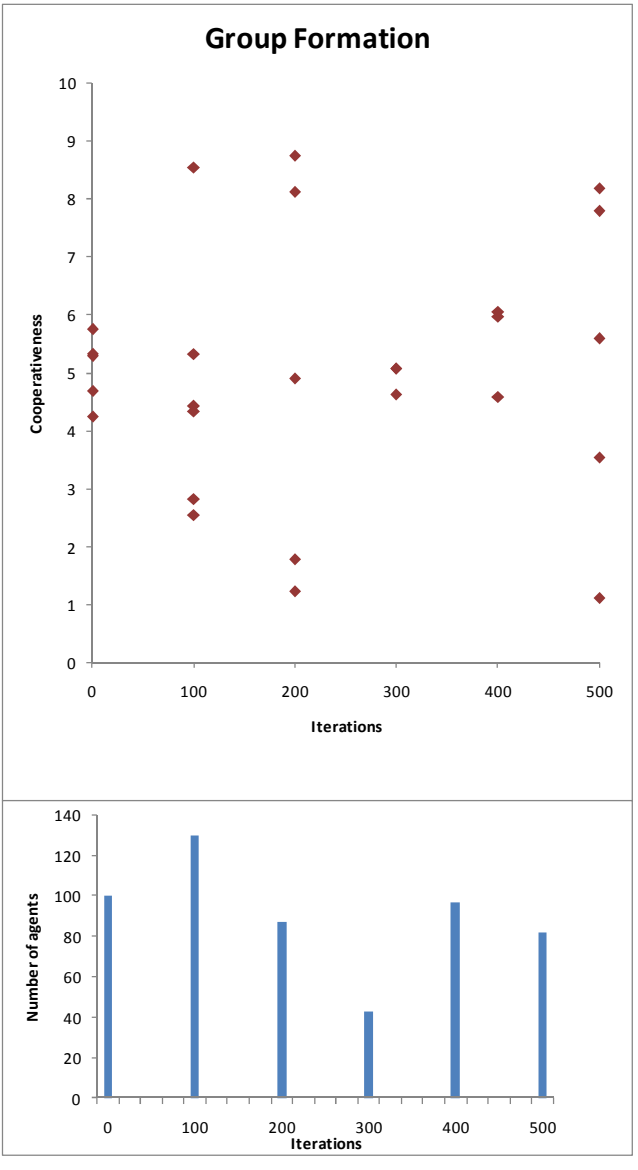

Fig. 1. Self-organisation of an open system when agents' arrival and departure rates are dynamic

There are two kinds of behavior we observe in the system. Firstly, the system dynamically enlarges or shrinks by creating more groups or dismantling existing groups based on the number of agents in the system. Secondly, it also forms groups based on cooperativeness. Cooperators move towards other cooperators and

non-cooperators end up with other non-cooperators. The agents self-organize into groups that have different ranges of cooperativeness. Thus this system restricts the non-cooperators taking advantage of cooperators by restricting their access to better groups.

4.2 Experiment 2 – Arrival Rate Greater Than Departure Rate

We conducted experiments by keeping the arrival rate greater than the departure rate. A run of this experiment is shown in Figure 2. It can be observed that when the number of agents increase, the system is able to dynamically create more groups and also these groups are separated based on the cooperativeness of the agents. This shows the scalability of the system.

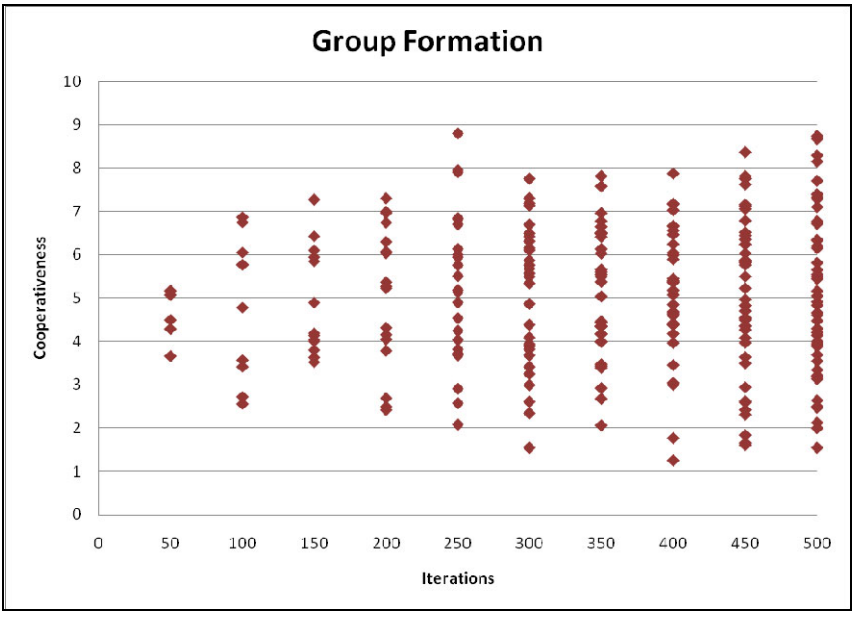

Fig. 2. Self-organization of an open system when agents' arrival rate is increased

4.3 Experiment 3 – Arrival Rate Equal to Departure Rate

When the number of new comers is roughly the same as the number of leaving agents in the system, the system will have same number of agents and the number of groups remain the same. But new agents who join the society have certain cooperativeness. Because of this the composition of groups and the cooperativeness of groups change over time. Figure 3 shows the cooperativeness of five different groups over 500 iterations. The cooperativeness of these groups varies depending upon the net effect of the cooperativeness of the agents that are present in the society. A new agent

whose cooperativeness value of nine joining a group whose average cooperativeness value is five will increase the group's average. In the same way, a bad agent leaving a good group will increase the group's cooperativeness average. Figure 3 shows how the 5 groups change over time based on the number of agents (composition of the group) and the cooperativeness of agents present in the system over time.

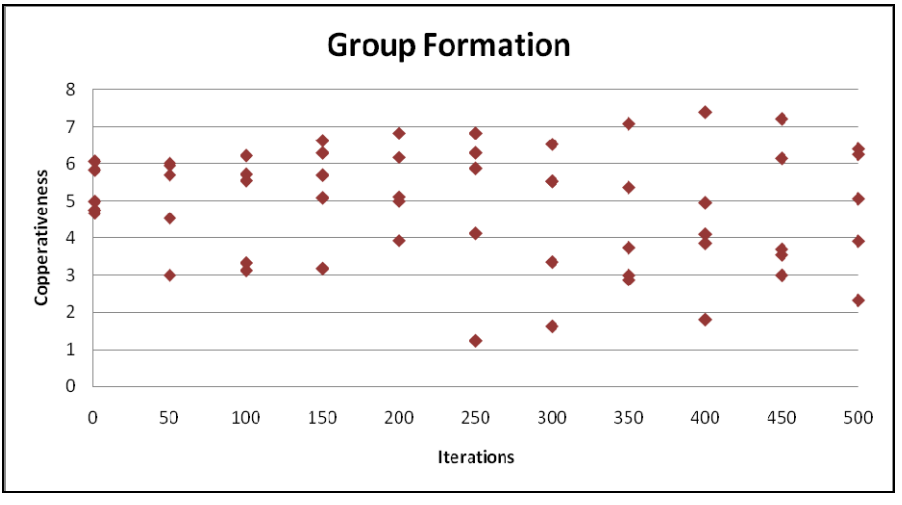

Fig. 3. Self-organization of an open system when the arrival rate is equal to the departure rate of the agents

4.4 Experiment 4 – Varying Life Spans of Agents

We varied the life span of the agents. We investigated the impact of the lifespan of agents on the system's behaviour. So we conducted two experiments by varying the lifespan. The lifespan of an agent is governed by the minimum time to live (TTL) parameter. The minimum TTL in one of the experiments was set to 300 and the other was set to 500. Figures 4 and 5 show the cooperativeness values of the groups for these two values of minimum TTL respectively.

Figures 4 and 5 show the result of groups' cooperativeness for 1000 iterations. From these results it can be observed that having longer life time (agents being in the society for longer period of time) helps to achieve better segregation of groups. This is because, when the agents live longer, they have a longer period to gather and use gossip. Additionally, when agents live for a shorter period of time, the system has a comparatively shorter period of time to segregate into groups than the system where the agents live longer. This can be observed by comparing the results for iterations 400 and 500. The separation of groups is better when minimum TTL=500. The same can also be observed in the circled regions of these two figures. The videos of these simulations can be seen in these links [14, 15].

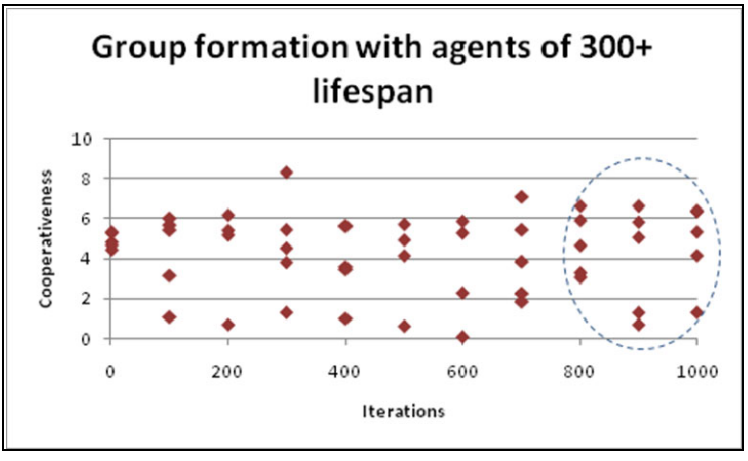

Fig. 4. Group formation with minimum TTL = 300

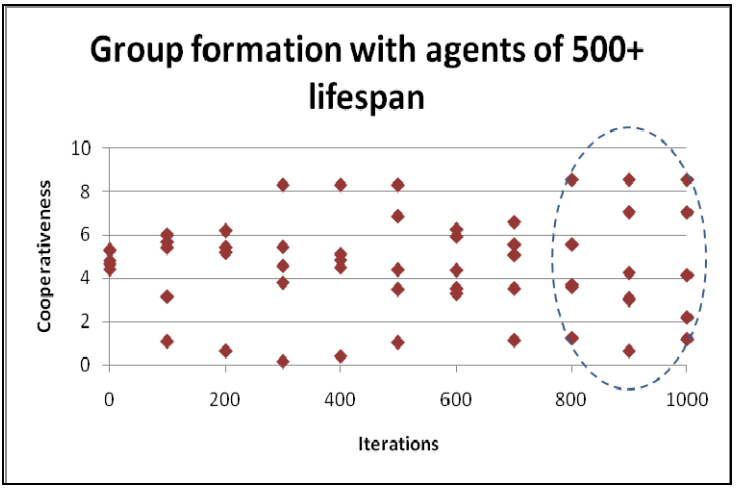

Fig. 5. Group formation with minimum TTL=500

5 Related Work and Comparison

In our previous work [7], the self-organization of peers in different groups was achieved by making use of tags and monitoring agents, where the population had a mixture of cooperators and non-cooperators. By employing a monitoring agent for each group, the system evolved into groups partitioned according to the performances of their group members. Each monitoring agent employed a voting mechanism within the group to determine which agents were the most and least cooperative members of the group. Then the most cooperative member was allowed to move to a new group, and the least cooperative member was expelled from the group. Those peers who left voluntarily or those who were expelled from their groups obtained membership in a

new group only if the local monitor agent of the other (new) group accepted them. Since the local monitor agents picked players for their group based on performance, the high performing player had a good chance to get entry into the best group, and the reverse conditions applied for the worst performing player. As a result, the players entered into groups based on their performances. Though this system produced good results, this approach is semi-centralized, because it required a local monitoring agent for each group. In addition the work considered a closed society. We believe this system can be applied in a regimented, closed society but cannot be applied to the modern systems which are open and distributed.

Hales's work [4], extends his previous work on tags to networks, considers a 'neighbor list of nodes' as a tag. The 'movement of node in a network' is modeled as a mutation. His results showed that tags work well for P2P systems in achieving cooperation, scalability and robustness.

In our present work, instead of the Prisoner's Dilemma game, we have adopted the more practical scenario of sharing digital goods in electronic societies. We investigate how a society can achieve the separation or self-organization of groups in a decentralized manner in an open society. Such a system would help to protect cooperators from being exploited by the non-cooperators. It would also restrict the non-cooperators from taking advantage of cooperators by restricting their entry to better groups where the access to resources is better. Hence, the quality of service (e.g. the quality of file sharing) and the performance (e.g. utility of agents) in the better groups will be higher. By doing so, the performance of the whole system can be improved; as resources can be distributed in greater proportion to the better performing groups [1]. Otherwise, it will be difficult to shield the cooperators from the defectors who rarely or never share their resources.

For easy understanding, we differentiate our current system from our previous work [7]. First we explain the results from the earlier system [7] for comparative purposes. In that work, all the 5 groups started with a similar number of cooperators in each group. Later the groups were separated into 2 groups having most of the cooperators, 2 groups having most of the non-cooperators and the middle group having a mixed population of both. But that earlier work employed localized group monitors and was therefore less scalable and semi-centralized.

The work presented in [8] is based on a closed society but cannot be applied to systems that are open and distributed. Even though the mechanism achieves self-organization, it is suitable for systems in which the performances of the other groups are directly revealed to the agents in the society.

The work presented in [9] shows the self-organization of groups using similar mechanisms and it has been improved upon in this current work. The differences between the work presented in [9] and current work are as follows. In earlier work [9] the game was played for certain iterations and the gossip information was stored. Later the agents use the stored gossip information when they play. In the current setup, the agents start using the gossip right from the start. If there is no information the agent is considered as a new player and allowed to play or enter into any group. As they play, the gossip is also stored and used. In the earlier work wealth has been taken into account. If the wealth of an agent has not increased in the last certain

number of iterations then the agent decides to move. In the current setup, instead of wealth if the rejection limit is met then the agent decides to move. We found that using a rejection limit works better for group separation than basing the decision on wealth, since it is likely that the wealth will increase for a certain number of iterations (because the agents play with bad agents if the gossip information was not available, hence the wealth of the bad players might increase).

In the earlier work [9] new players are introduced into the lowest group in the society and they are expected to build their way up to the higher groups based on their behaviour (cooperativeness). For that it was necessary to keep track of the lowest group of the system all the time, which is not a recommended practice if we want to achieve a decentralized environment. In the current setup new agents go to random groups in the society. As they are new they have no past behaviour to track and they are allowed in any group as they come in. Eventually they will end up in a group based on their behaviour by the mechanism we have in place. In the earlier work the remaining agents in a dismantling group go to the lowest performing group. In the current setup, they can apply to other groups and go to the group that accepts them. If they are not allowed then they keep trying to get entry into one of the groups.

In summary, our current work focuses on addressing the free-riding problem in an open, dynamic and distributed society. The work presented here provides an improved model when compared to the model presented in [9].

In future, we intend to include false gossip (lying) in the system and examine the mechanisms for handling the lying problem.

6 Conclusion

We have presented a gossip based decentralized mechanism to facilitate the self-organization of agent groups in open agent societies. Through agent based simulation we have demonstrated that our mechanism helps the sharing agents (cooperators) to move to better groups while the non-sharing agents are restricted from getting into the better groups. Thus, the mechanism achieves the separation of groups. The mechanism allows for dynamic group formation through the splitting and dismantling processes. We have also demonstrated that our system is scalable. Finally, we have compared our results with previous works.

Acknowledgments. Our sincere thanks to the New Zealand Federation of Graduate Women (NZFGW-Otago branch) for the NZFGW Travel Award.

References

1. Antoniadis, P., Grand, B.L.: Incentives for resource sharing in self-organized communities: From economics to social psychology. In: ICDIM, pp. 756–761. IEEE (2007)
2. de Pinninck, A.P., Sierra, C., Schorlemmer, M.: Distributed Norm Enforcement: Ostracism in Open Multi-Agent Systems. In: Casanovas, P., Sartor, G., Casellas, N., Rubino, R. (eds.) Computable Models of the Law. LNCS (LNAI), vol. 4884, pp. 275–290. Springer, Heidelberg (2008)

3. Eugster, P., Felber, P., Le Fessant, F.: The "art" of programming gossip-based systems. SIGOPS Oper. Syst. Rev. 41(5), 37–42 (2007)
4. Hales, D.: Self-Organising, Open and Cooperative P2P Societies – From Tags to Networks. In: Brueckner, S., Di Marzo Serugendo, G., Karageorgos, A., Nagpal, R. (eds.) ESOA 2005. LNCS (LNAI), vol. 3464, pp. 123–137. Springer, Heidelberg (2005)
5. Hardin, G.: The Tragedy of the Commons. Science 162, 1243–1248 (1968)
6. Jelasity, M., Montresor, A., Babaoglu, O.: Detection and removal of malicious peers in gossip-based protocols. In: FuDiCo II: S.O.S., Bertinoro, Italy (June 2004)
7. Purvis, M.K., Savarimuthu, S., De Oliveira, M., Purvis, M.: Mechanisms for Cooperative Behaviour in Agent Institution. In: Nishida, T., Klusch, M., Sycara, K., Yokoo, M., Liu, J., Wah, B., Cheung, W., Cheung, Y.-M. (eds.) Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2006), pp. 121–124. IEEE Press, Los Alamitos (2006) ISBN 0-7695-2748-5
8. Savarimuthu, S., Purvis, M.A., Purvis, M.K.: Self-Organization of Peers in Agent Societies. In: IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Milan, Italy, Los Alamitos, CA, USA, September 15-18, vol. 2, pp. 74–77 (2009) ISBN 978-0-7695-3801-3
9. Savarimuthu, S., Purvis, M., Purvis, M., Savarimuthu, B.T.R.: Mechanisms for the Self-Organization of Peer Groups in Agent Societies. In: Bosse, T., Geller, A., Jonker, C.M. (eds.) MABS 2010. LNCS, vol. 6532, pp. 93–107. Springer, Heidelberg (2011)
10. Rebecca, S.B.: Some Psychological Mechanisms Operative in Gossip. Social Forces 34(3), 262–267 (1956), Stable http://www.jstor.org/stable/2574050
11. Thomsen, R.: The Origins of Ostracism, A Synthesis. Gyldendal, Copenhagen (1972)
12. Savarimuthu, S.: Self-organising groups (gui for closed society). University of Otago (February 2010c), http://unitube.otago.ac.nz/view?m=9GT31pqTPSk
13. Savarimuthu, S.: Self-organising groups (gui for open society). University of Otago (February 2010d), http://unitube.otago.ac.nz/view?m=HbOw1pni7qS
14. Savarimuthu, S.: Self-organising groups (gui for lifespan=300+). University of Otago (February 2010a), http://unitube.otago.ac.nz/view?m=JHaY1poMt9P
15. Savarimuthu, S.: Self-organising groups (gui for lifespan=500+). University of Otago (February 2010b), http://unitube.otago.ac.nz/view?m=7SK81pp4WP0
16. Sommerfeld, R.D., Krambeck, H.J., Semmann, D., Milinski, M.: Gossip as an Alternative for Direct Observation in Games of Indirect Reciprocity. Proceedings of the National Academy of Sciences of the United States of America 104(44), 17435–17440 (2007), Stable http://www.jstor.org/stable/25450253
17. Saroiu, S., Gummadi, P., Gribbe, S.: A measurement study of peer-to-peer file-sharing systems, Technical report UW-CSE-01-06002, University of Washington (2002)