Savarimuthu, S., Purvis, M., Purvis, M., & Savarimuthu, B. T. R. (2015). An agent-based simulation for restricting exploitation in electronic societies through social mechanisms. AI & Society, 30(3), 345-358. doi: 10.1007/s00146-013-0529-0

ORIGINAL ARTICLE

An agent-based simulation for restricting exploitation in electronic societies through social mechanisms

Sharmila Savarimuthu · Maryam Purvis · Martin Purvis · Bastin Tony Roy Savarimuthu

Abstract One of the problems in artificial agent societies is the problem of non-cooperation, where individuals have motivations for not cooperating with others. An example of non-cooperation is the issue of freeriding, where some agents do not contribute to the welfare of the society but do consume valuable resources. New mechanisms for group self-organisation and management in multi-agent societies are presented and examined in a multi-agent societies where nodes of a P2P system are modelled as interacting agents belonging to different groups. The context of interaction between agents is the sharing of digital goods in electronic societies. We have simulated a decentralised P2P system which self-organises itself to avoid cooperative sharers being exploited by uncooperative free riders. Specifically, we illustrate how cooperative sharers and uncooperative free riders can be placed in different groups of an electronic society in a decentralised manner. Inspired by human society, we use social mechanisms such as tags, gossip and ostracism. Our aim here is to restrict exploitation or in other words restrict uncooperative behaviour by separating groups based on performance since it reduces the likelihood of bad agents exploiting the good agents in the better groups. The developed system shows promising results by encouraging sharers to move to better groups and also by restricting free riders without any centralised control, which makes these mechanisms appropriate for distributed policy governance. Our work offers new insights into policy mechanisms for regulation of distributed societies.

Keywords Policy mechanisms · Self-organising systems · Freeriding problem · Multi-agent systems

S. Savarimuthu (✉) · M. Purvis · M. Purvis · B. T. R. Savarimuthu
Department of Information Science, University of Otago, Dunedin, New Zealand
e-mail: sharmilas@infoscience.otago.ac.nz

M. Purvis
e-mail: tehrany@infoscience.otago.ac.nz

M. Purvis
e-mail: mpurvis@infoscience.otago.ac.nz

B. T. R. Savarimuthu
e-mail: tonyr@infoscience.otago.ac.nz

1 Introduction

Human societies have long developed and evolved social mechanisms for facilitating cooperation among individual members and subgroups. The advent of dynamic societies of the technological world, in particular the large and rapidly changing electronic societies, call for the use of new mechanisms to facilitate cooperation among artificial agents that can be modelled after those employed in human societies.

One of the most persistent problems in peer-to-peer (P2P) networks is freeriding (Ramaswamy and Liu 2003; Feldman and Chuang 2005; Krishnan et al. 2004). There are published examples of centralised approaches in facilitating cooperation that employ centralised regulations to control freeriders (Esteva et al. 2004; Purvis et al. 2006). These researchers have used monitoring agents or governor agents to control agent behaviour. Even though centralised systems have several advantages, such as direct control and access, they have several limitations as well. They suffer from bottlenecks when the number of agents increases in the system. They are computationally expensive, because of the cost associated with avoiding performance bottlenecks, and they are prone to single-points-of-failure.

Electronic marketplaces are increasingly being used. Some of the well-known examples include eBay (Omidyar 1995), TradeMe (Morgan 1999) and Amazon.com (Bezos 1995). These market places facilitate trading (buying and selling) goods online, and they provide mechanisms that ensure a fair trade where the expectations of buyers and sellers are met. These systems also provide mechanisms for specifying and monitoring obligations (e.g. a buyer should pay within 3 days after winning an auction). These mechanisms ensure secure and safe trading in these electronic market places.

Electronic societies are also plagued by scenarios that can be likened to the tragedy of the commons. Freeriding is a well-known example in P2P systems, where an agent does not contribute to the society by sharing its files but downloads the files from other users. P2P communities heavily rely on altruistic sharing of digital goods such as video files, audio files and e-books. In bittorrent, tracking servers are used to monitor its users. If they share, their downloading speed will be increased. If they leech, it will be reduced.

To ensure security and safe trading, online markets have a mechanism of users rating each other based on their interactions. Consider eBay (Omidyar 1995) as an example, which is currently the world's largest electronic market place. In eBay, people leave feedback about others whom they have had transactions with. Users could give positive, neutral or negative feedback. The feedback about users is associated with their profile and is visible to everyone. The reputation of a user (both as a seller and a buyer) is calculated by the overall ratings. In this approach, the users are expected to report honestly and, this approach works well for the current system. The user has no control over his ratings being visible to everyone using the system. Ebay displays the reputation records about all its members. It suspends or removes users violating the rules. People generally tend to behave honestly and show their good side, since their reputation score is visible publicly. The advantage of this kind of social control is that it enables smoother functioning in electronic societies. The disadvantage is that the user loses control to the central authority which stores the user's information. The user has no control over his/her information being visible to everyone. We believe decentralised mechanisms where information is distributed across different entities can be adopted in order to achieve the same purpose. For instance, partial information about the reputation of users can be held by distributed peers.

Once the information about agent behaviour has been collected (either through centralised or distributed mechanisms), this information can be used for two purposes in the context of identifying good behaviour and discouraging bad behaviour in the system. Firstly, this information can be used for producing recommendations. For example, an agent can vouch for another agent based on that agent's interactions (i.e. through a referral process). An agent can also identify the best agent or a set of good agents that have cooperated in the past (e.g. in the context of voting in a society for the best agent, agents can nominate the best one based on their past interactions). Secondly, this information can be used for controlling bad agents. For example, the reputation score can be used to restrict resources for bad agents and can even be used to sanction bad agents in a society. Reputation-based mechanisms are shown to solve social dilemmas (Milinski et al. 2002).

Monitoring agent interactions can be achieved through two approaches. The first approach is to use a centralised reputation system which collects all the information (e.g. eBay, TradeMe, Amazon.com) as explained before. Since there are several disadvantages of this approach, several multi-agent researchers have investigated reputation-based systems for monitoring agent interactions (Gursel et al. 2009; Yolum and Singh 2005) which use distributed approach. They make use of a decentralised, partial reputation system such as the use of referral or gossip in agent systems. In these systems, each agent has only partial information. These types of systems are more scalable and decentralised. We have adopted a similar approach of using several social mechanisms to establish social control in this work.

From a larger perspective, our work offers new insights into policy mechanisms for regulation of distributed societies. The mechanisms we describe take advantage of community-based context to discriminate between cooperative and antisocial members so that scarce resources can be appropriately apportioned in a manner that scales effectively with increasing group size. This can be more effective than having group policies monitored and governed by central authorities, which places heavy dependence on the reliability of a limited number of managing agents.

There is a need for decentralised solutions to deal with the freeriders in these distributed P2P societies. In response to this need, our work proposes a decentralised solution that makes use of social mechanisms such as tags, gossip and ostracism. The inspiration to use social mechanisms for our work comes from the human societies, which have evolved over time to work effectively in groups. For humans, group mechanisms provide efficient and decentralised social machinery that supports cooperation and collaboration. Of course, social control can always be employed through leadership mechanisms. For example, the leader can impose rules on his followers. The disadvantage of such an approach is that it is centralised. On the other hand, it is known that social control can also be achieved by means of decentralised approaches.

2 Related work

In the work of Purvis et al. (2006), the cooperative self-organisation of peers in different groups was achieved by playing the Prisoner's Dilemma (PD) game and 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 or were expelled from their groups obtained membership in a new group only if the local monitor agent of the new group to which entry was sort accepted them. Since the local monitor agents selected players for entry to their group based on performance, a high-performing player had a good chance to get entry into a top group, and the reverse conditions applied for a poor performing player. As a result, the players entered into groups based on their performances. But this approach was still semi-centralised, because it required a local monitoring agent for each group. Hales's work (2004) showed that tags work well for P2P systems in achieving cooperation, scalability and robustness.

In our present work, instead of the PD game, we have adopted the more practical scenario of sharing digital goods in electronic societies. We investigate how we can achieve into the separation (i.e. self-organisation) of groups, based on their behaviour in a decentralised manner and in an agent society. Such a system can help to protect cooperators from being exploited by the non-cooperators. It would also restrict the non-cooperators from taking advantage of cooperators and restrict their entree to better groups where the access to resources is set to a higher level, and hence, the quality of service/performance is higher. By doing so, the performance of the whole system can be improved, because resources can be distributed in greater proportion to the better performing groups (i.e. with increased bandwidth for downloads in file-sharing systems). Otherwise, it will be difficult to shield the cooperators from the defectors who rarely or never share their resources.

Skyrms and Pemantle (2000) and Skyrms (2009) investigate the dynamic evolution of social networks to maintain cooperation. Agents in their model do not make use of gossip information, but they make use of their own experience with other players (i.e. depending upon whether they were treated nicely in the visit they made to other agent's place). The agents cut-off their link to other players if the interaction was not pleasant. In our work, an agent ostracises another agent based on the gossip information.

De Pinninck et al. (2008) have adopted gossip and ostracism mechanisms to achieve a similar goal as ours. In their work, gossip has been used as an identification mechanism to spread and find information about the norm violators. In our work, we use gossip for a similar purpose, which is to identify freeriders. In their work, gossip is shared locally with nearby agents called "mediators", and it does not need complex computations. The mediator agent contacts the enforcer agent to punish the violator. Gossip has been used in relation to the norm enforcement technique in their work. It has also used an ostracism-based mechanism to punish norm violators. Their work makes use of the concept of a normative reputation for each of the agents in the society. Depending upon whether an agent abides by the norm, its reputation is spread through gossip. Agents with a bad normative reputation are ostracised by the members of the society (i.e. agents stop interacting with a norm violator). By ostracising the norm violators, agents achieve better payoffs by interacting only with the normative agents. Thus, the norm is enforced, and the norm violators are punished in an open-agent society. Similarly, our work also uses ostracism along with gossip to identify freeriders and restrict exploitation in a simulated P2P environment. In their work, gossip has been used as an identification mechanism to spread and find information about the norm violators. In our work, we use gossip to identify freeriders.

The social mechanisms deployed in this work are tags, gossip and ostracism. Tags are used to form groups. A gossip-based mechanism is used to achieve social control, as it serves as a distributed referral mechanism where information about a person is spread informally among the agents. Another social mechanism to restrict freeriders is ostracism. Members that do not adhere to the values, expectations or norms of the groups can be sanctioned by other agents by their refusal to interact with those agents. These mechanisms can be used to design policies to restrict exploitation in artificial societies.

The remainder of the paper is organised as follows. The social concepts used in this work are introduced in Sect. 3. The developed mechanisms for self-organisation of groups are listed in Sect. 4. Our experimental setting and selected experimental results are described in Sects. 5–8. In Sect. 9, we discuss the future work. Finally, Sect. 10 concludes the paper.

3 Social mechanisms

The social mechanisms used in our experiments to deal with free riding are described below.

3.1 Tags

Tags are used as group-identifying mechanism to form teams and to improve cooperative behaviour within the group. In our work, tags are used as bootstrapping mechanisms to form initial groups, and later the groups evolve based on agents' behaviour. In the initial set-up, agents are put into random groups. Each group is 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.

3.2 Gossip

Gossip is a powerful mechanism in human society for information sharing. Research done by evolutionary biologists suggests that humans have shown more interest in gossip more than in the original information (Sommerfeld et al. 2007), when participants were presented with both types of information (the gossip information and the "real", original information). Based on their research, they have noted "gossip has a strong influence… even when participants have access to the original information as well as gossip about the same information" and also that "gossip has a strong manipulative potential".

In a way, a gossip mechanism can be considered to be a "distributed referral" mechanism (Eugster 2007). It is similar to having a reputation system, except that the gossip information is distributed. Paine (1967) has explained that people who gossip within their gossip circle feel the fellowship or belongingness to the community. In fact gossip is a property of a group (Paine 1967) that can be used to provide local social control; it aids in maintaining the social structure. Gossip can also be considered as an indirect attack on a person where no other way of sanctioning is easily possible. Thus, gossip is a mechanism for transmitting public opinion which can lead to some social benefit (Stirling 1956). Gossiping is also a way of doing social comparison (Suls 1977; Paolucci and Marsero 2000).

The grooming behaviour in animals is analogous to the gossip behaviour in humans (Dunbar 1996). Dunbar's work (2004) also acknowledges that gossip is a way of social bonding, and it is a part of social life. An additional benefit of gossip is that it helps to control freeriders. Gossip information about freeriders is important, because people do not want to be exploited by a freerider, and freeriders degrade the societal welfare. The most common way of controlling freeriding is based on memory of past behaviour (Dunbar 2004). Gossip helps this by being a medium for 'information storage and retrieval' (Roberts 1964).

3.3 Ostracism

It has long been the case in human and animal societies that the member of a group who does not abide by rules or norms can be punished by other members of the groups (the followers of the rule/norm). One kind of punishment is ostracism (Thomsen 1972), which results in the social exclusion of the punished member. All the other members of the society would stop interacting with the member who is being ostracised and would no longer consider that person to be a part of their group (by ignoring him/her). This kind of behaviour is used as a social punishment mechanism where there is no higher authority or institutional monitor to check deviations and establish punishment. Thus, it is a decentralised mechanism as opposed to having a central controlling authority to establish sanctions.

4 Mechanisms developed for self-organisation of groups

The four mechanisms are:

- Dynamic grouping mechanism (Sect. 5)
- Random hopping mechanism (Sect. 6)
- Individual group history mechanism (Sect. 7)
- Sharing group history mechanism (Sect. 8)

The developed mechanisms and their results are explained in detail in their separate sections.

In this work, we demonstrate how social mechanisms can be developed and employed for agents in a closed and decentralised society which has several groups. In open societies, the number of agents changes over time (Savarimuthu et al. 2013). In a closed society, the number of agents is fixed. Our aim here is to restrict exploitation, specifically restrict uncooperative behaviour by separating groups based on performance in closed societies.

5 Dynamic grouping mechanism

In our simulation model, agents are engaged in the sharing of digital goods in a simulated P2P environment of an artificial agent society. The interaction between the agents is in the context of sharing files.

5.1 Experimental set-up

In the initial set-up, 100 *(Number of agents)* agents are randomly divided into 5 *(Number of groups)* groups, 20 each. Agents are initialised with random attributes. Each agent has a blackboard of certain size *(Gossip blackboard size)*, which is used for storing gossip information. After reaching the limit, it rolls over based on First-In-First-Out (FIFO) algorithm. This set-up updates old gossip information as new

gossip information comes in. These blackboards are individual blackboards for agents. We do not model them as common blackboards, because if a common blackboard fails, then that would affect a group of agents.

5.2 Agent attributes

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

- **Cooperativeness**
  This attribute specifies how cooperative an agent is. An agent has a randomly assigned cooperation value between 0 and 10 that represents how much it cooperates (shares), 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. For example, if the cooperativeness of an agent is 3, then it will cooperate 3 out of 10 times.
- **Tolerance level**
  The tolerance level is a value between 1 and 10, which characterises 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. An agent with a tolerance value of 1 leaves the group after experiencing defection once and an agent with a tolerance value of 10 leaves the group only after 10 defections. Each time the agent experiences a defection, it increases its tolerance count. The agent decides to leave when its tolerance count reaches its tolerance level.
- **Rejection limit**
  The rejection limit is an attribute of an agent which represents how many *rejections* (rejections are described in 5.5) the agent can face before it decides to leave for another group. Every time it is rejected from the play, it increases its rejection count. The agent decides to leave when its rejection count reaches its rejection limit.
- **Gossip blackboard length**
  Each agent has a gossip blackboard of certain length to store the gossip messages from other agents of its group. This blackboard will be used by an agent to post the gossip information provided by other agents. For example, agent A posts the gossip it heard from agents B and C on their interactions with agents D and E, respectively. These are individual blackboards for agents, but can be referred to, if any requests are received.
- **Cost and benefit for sharing**
  Agents share files. Whenever a file is shared, the receiving agent receives a value of 1.0 as its benefit, while the sharing agent loses 0.1 as a cost (cost is associated with sharing since the sharing agent loses time and bandwidth for sharing).

Table 1 Experimental parameters for dynamic grouping mechanism

Parameters	Values
Number of agents	100
Number of groups	5
Number of iterations	5,000
Cost for donation	−0.1
Benefit for receiving	1
Number of gossip requests	5
Cooperativeness	0–10 (random)
Tolerance	1–10 (random)
Rejection limit	10
Gossip blackboard size	10

- **Tag groups**
  In the initial set-up, agents are put into random groups. Each group is 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.

5.3 Experimental parameters

The experimental parameters are listed in Table 1.

5.4 Gossip interaction

An agent can make a request for a file to its fellow group agent. Whether the agent gets the requested file or not, it can gossip about the outcome to another agent in its group. In the gossip mechanism, there is no lying. It is assumed that the agents report honestly, since this happens within the group and there is no incentive to lie. In this fashion, every transaction is reported (gossiped about) to one of the other agents in the group. Thus, the total system has some partial information about every agent, maintained in a distributed way. The first 1/10 iterations (500 out of 5,000 in this experiment) are played in this manner to build up a distributed gossip repository among the players. For further illustration, the operation of how peers publish gossip is outlined schematically in Algorithm 1. Consider agents A, B and C. A gossips about B to C

Algorithm 1: Pseudocode for gossip.

```
1 begin
2 |  A requests for file to B ;
3 |  if B shares then
4 |  |  A gossips positively about B to C;
5 |  else
6 |  |  A gossips negatively about B to C;
7 |  end
8 end
```

After 500 iterations, agents use this gossip information to avoid "playing" (i.e. engaging in trades) with the worst player of their group. When a player requests a file, the giving player can request gossip from five *(Number of gossip requests)* other random agents (asking them what they know from the gossip information they have received) whether this asking player (taking player) is the worst cooperator of their group. The worst player is the one who has been uncooperative the most often in its group (according to the available gossip information). If the taking player is the worst player, the giving player refuses to interact with (i.e. rejects) the taking player. Otherwise, this giving player interacts (sharing a file or not based on its own cooperativeness). The operation of how peers use gossip is outlined in Algorithm 2, where D and E are the players in the example group. Assume here that E is the taking player, D is the giving player, and D checks with any 5 players in the group in order to see whether E is the worst player in their group.

Algorithm 2: Pseudocode to avoid the worst player.

```
1 begin
2 |  D makes a request to 5 other players for gossip about E ;
3 |  D receives gossip;
4 |  if E is the worst player then
5 |  |  D refuses to play with E;
6 |  else
7 |  |  D plays with E ;
8 |  end
9 end
```

When only a few agents (<5) have gossip about a taking player, then only the available information is taken into consideration. Sometimes, it can be the case that none of the players has gossip information about the taking player. In such a case, the taking player is considered to be "average" and not to be the worst player, a privilege similar to what often happens when a new player joins a group.

5.5 Leaving a group

A player can leave a group for two reasons. One reason is when if an agent's tolerance level is surpassed or exceeded, which means the agent is in a group where others do not cooperate as much as the agent's expectation. After a number of defections from the group members, the agent may decide to leave that group. The agent asks other groups about their cooperativeness and tries to enter into the group whose cooperativeness is better than its current group. If the better groups do not allow the agent in, then since it does not want to move to lower groups, it stays in its current group and its tolerance count is reset to 0.

Second reason for leaving a group is when an agent moves to another group based on its rejection limit. Other agents can refuse to interact (i.e. share resources) with an agent if that agent is identified as the "worst", (i.e. the least cooperative agent) in the group. Instead of picking an agent randomly and checking it for cooperativeness (as mentioned in the previous work by Savarimuthu et al. (2009), the agents in this set-up stop playing with a worst player (as shown in Algorithm 2) hence after being rejected for a certain number of times (the *rejection limit*), the worst agent chooses to leave that group and moves to another group in the hope of entering into a more favourable trading environment.

5.6 Choosing a group to join

The leaving agent asks other groups about their group's calculated cooperativeness and tries starting from best to next best and so on, seeking to find a group which allows it in. If none of the groups allow it, then it stays in its current group.

The criteria for a group accepting or rejecting an agent depends on the situation of that group. In this set-up, a group's entry value is calculated based on the group's standards which are the group's current size and cooperativeness. Based on these two factors, the entry value is determined, and the agents seeking entry are assessed [In the previous work (Savarimuthu et al. 2009), the entry level was set to a particular value based on group rank using predefined values at design time].

Determining a group's entry value is described in the next section (Sect. 5.7).

5.7 Entry criteria

We call the agent that tries to hop to a new group a "hopping peer". The hopping peer asks any randomly chosen agent in the group to which it seeks permission to enter. We call this permission-granting agent in the group to which entry is sought, the "checking peer". The 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:

$$\mathsf{EV} = \mathsf{C} - (\mathsf{C1}/(\mathsf{SL} - \mathsf{S})^{\mathsf{C2}}) + \mathsf{C3}^{(\mathsf{S}-\mathsf{SU})} \quad (1)$$

The group Entry Value (*EV*) is calculated considering the given group's calculated cooperativeness (*C*) and its group Size (*S*). *C1*, *C2* and *C3* are constants whose values in our experiments were 25, 2 and 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. If a group becomes too big, then it becomes unmanageable. If a group becomes too small, below a certain value, then the group is not considered to be an active group. There should be at least a certain number (minimum value) of players in it to be considered as a team (e.g. a sports team may have a certain number of players as a minimum for the existence of a team/group. Thus, a volleyball team must have at least 6 players. If there are <6 players, then it is not considered to be a team/group). In our experiments, *SU* was set to be 25, and *SL* was set to be 10. That means that if the size of the group is 10 or below, the entry qualification value will be set to 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.

Figure 1 shows the range of entry values for groups with different group sizes and cooperativeness. A simple example illustrates the use of this formula. Consider that a group's calculated cooperativeness (*C*) 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. This can be identified in Fig. 1 by examining the line Avg6 for size 14 and for size 25. The avg line represents the group's calculated cooperativeness. It can be inferred that if the group size becomes large (closer to *SU*), the group is strict about who it lets in, and if the group size is small (closer to *SL*), it lets almost anyone into the group. When the group size is 10, the entry value is 0.

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 asking for gossip information from the hopping peer's group in order to assess the request of the hopping peer to join its group.

Consider a case where G and F are in different groups. G is the checking peer and F is the hopping peer that wants to enter G's group. F asks G for entry, then G asks 5 other randomly chosen players in F's group for gossip information about F's cooperativeness. If F's estimated cooperativeness calculated through this gossip information is greater than or equal to the entry value (EV) of G's group, G allows entry for F; otherwise it denies. If it is denied entry to a group, F will try to enter into other groups. This process is outlined in Algorithm 3. The hopping peer will ultimately get into a group where its cooperativeness makes it eligible to enter. If no such group is available, the hopping peer stays in its current group.

Algorithm 3 depicts the process of Agent F seeking entry into the group of agent G.

Algorithm 3: Pseudocode for using gossip information to hop between groups.

```
1  begin
2  |  G requests 5 other players for gossip about F ;
3  |  G receives gossip about F and calculates F's estimated
   |  cooperativeness;
4  |  G calculates entry value based on Formula 1;
5  |  if F's estimated cooperativeness ≥ entry value then
6  |  |  F gets entry;
7  |  else
8  |  |  F does not get entry ;
9  |  end
10 end
```

5.8 Results of experiments

The entire process explained so far is repeated for many iterations, and over time, the agents are found to collect into groups where there are agents with similar cooperativeness to them. Because of this process, some groups will emerge as elite groups with many cooperators, and other groups will have less cooperative players. As a consequence, this mechanism achieves a separation of groups based on performance. Figure 2 shows the self-organisation of groups based on their cooperativeness.

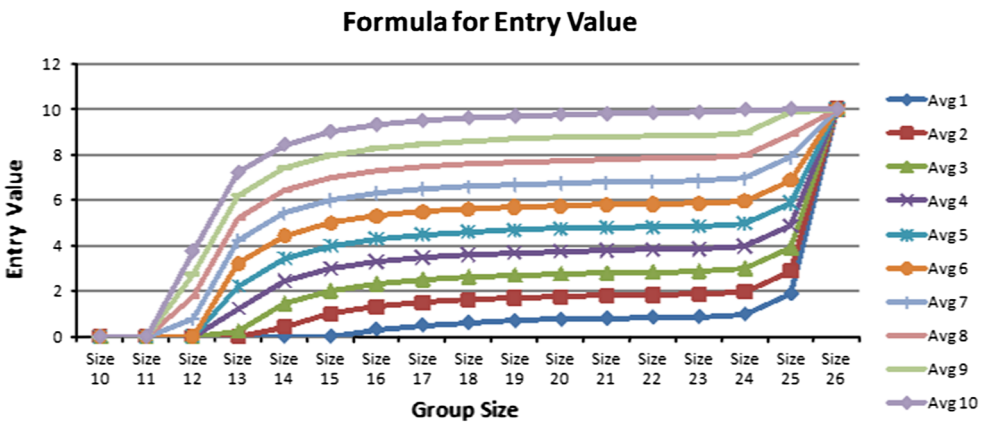

Fig. 1 Formula for entry value

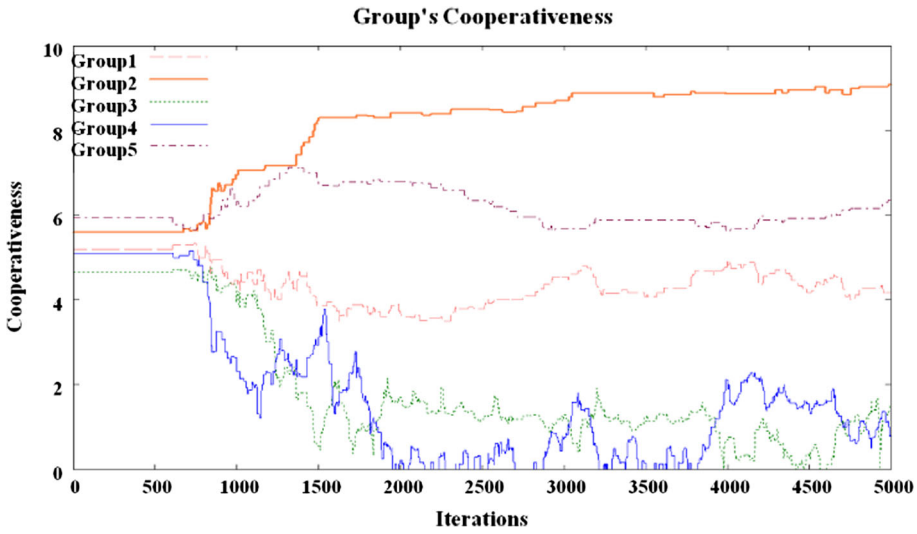

Fig. 2 Self-organisation of groups using dynamic grouping mechanism

The main differences between the previous work (Savarimuthu et al. 2009) and the current mechanism are that (a) this mechanism uses gossip information to avoid entering into interaction with the worst agent, (b) an agent decides to leave the group based on its tolerance level and rejection limit and (c) in this mechanism, the entry value is determined by group size and cooperativeness of a group at run-time (whereas it was determined at design time in the previous mechanism). According to the number of groups, the specific values for group entry needed to be specified at design time in the previous work (Savarimuthu et al. 2010), but for the current mechanism, no such requirements are necessary.

6 Random hopping mechanism

We conducted an experiment to investigate whether random hopping of agents would lead to some kind of behaviour pattern in the system. This experiment has the same experimental set-up as the previous experiment, except for a few changes. The agents do not have information about the other groups' performance (group's calculated cooperativeness) and so they decide to hop to another random group. After playing a certain number of games in a group (10), agents try to hop to another random group. The entry criterion is the same as the previous mechanism as explained in Sect. 5.7. This mechanism did not result in the separation of groups, since the agents are hopping randomly and they never settle down in a group. Due to this, there is no separation of groups based on cooperativeness like the previous mechanisms.

7 Individual group history mechanism

In order to overcome the random hopping issue in the previous experiment, we developed a mechanism in which agents can keep the memory of their previous groups. Each agent has a memory of a certain number of previous groups to which it belonged. In both the previous mechanisms we have described, an agent has no memory of the previous groups it has belonged to. In the new mechanism, however, agents keep the memory about the previous group's cooperativeness.

This information about other groups might be helpful for agents in making their decision. By keeping the memory, agents can avoid making random choices. Agents can make their decision relying only on their own experience about the other groups, which could potentially lead to self-organisation.

We have conducted two experiments, one with memory of just the previous group only and another with memory of all previous groups an agent has been a member of.

7.1 Memory of last group

In this experiment, agents have just one memory slot to store the information about its previous group. After a certain number of "games" in a group, the agent compares its current group's cooperativeness and the previous group's cooperativeness. If the previous one was better, it tries to hop back to the previous group. Otherwise, it stays in the current group. This experiment did not show significant self-organisation in terms of separation into groups of different cooperation values (see Fig. 3), since the agents keep on hopping back and forth between previous and current groups.

7.2 Memory of all previous groups

Instead of keeping just one memory, the agents in this set-up have memory of all their previous groups. The agent compares its current group's cooperativeness with the other

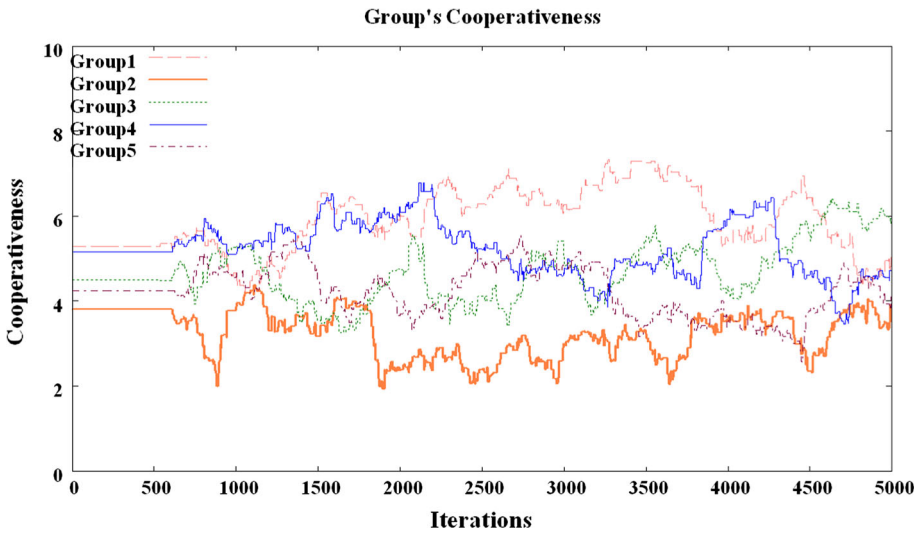

Fig. 3 Group history mechanism with memory of last group

groups in its memory. If its current group is not the best, then it tries to move to the best group in its memory. If not allowed, then it tries the next best group and so on. If it is not allowed in any of the better groups, then it stays in the current group.

For example, the group with best cooperativeness at iteration 1,000 may later (at 3,000th iteration) be the worst group and the agent does not know that. The agent left that group at iteration 1,000, but later the group's cooperativeness has been changed because of arrival and departure of agents over time.

Figure 4 shows there was no evident self-organisation of groups based on cooperativeness. Thus, this mechanism also did not achieve significant self-organisation of groups based on cooperativeness since the agents keep on moving to different groups. The main reason for this is the agents not having the latest information about other groups. This is because the group's size and cooperativeness change from time to time. But agents have the memory from their previous experience which may not be recent. So this mechanism did not achieve self-organisation.

8 Sharing group history mechanism

As the previous mechanism (Sect. 7) did not lead to self-organisation because of agents hopping to other groups based on their own memory of cooperativeness, we investigated a mechanism where agents share these information and follow the recent information available. We call this mechanism the "sharing group history" mechanism. This experiment differs from the previous experiment in several respects, which are described in the appropriate sections below.

8.1 Gossip interaction

The gossip interaction takes place in the same manner as described in connection with Algorithm 1. If the requesting agent is the worst cooperator in the group, the player refuses to play with him (in the same manner as described in connection with Algorithm 2).

8.2 Leaving a group

A player can leave a group if its tolerance level is surpassed or when its wealth (score) has not increased (remember every time the agent receives a file, it adds up 1 *(benefit for receiving)* to the score). If the agent's tolerance limit is reached, the agent will decide to leave that group and move to another group.

In addition, when other agents reject to play with the worst agent, and it is regularly rejected from play, then, of course, that agent's score will not increase. If, over a given period of play opportunities (e.g. 15 iterations), an agent's wealth (score) has not increased, then it will choose to leave that group and move to another group. Since the other players in its current group are not playing with it, it will be better off (i.e. able to increase its wealth) by moving to another group.

8.3 Choosing a group to join

The hopping peer then collects information about other groups from their group members. Then, it decides from which group to request admission. Every agent has a memory record of its most recent groups (in our experiments, the memory capacity was set to 4 past experiences).

For example, assume agent E has been in 3 other groups before, as shown in Table 2. The first row of the

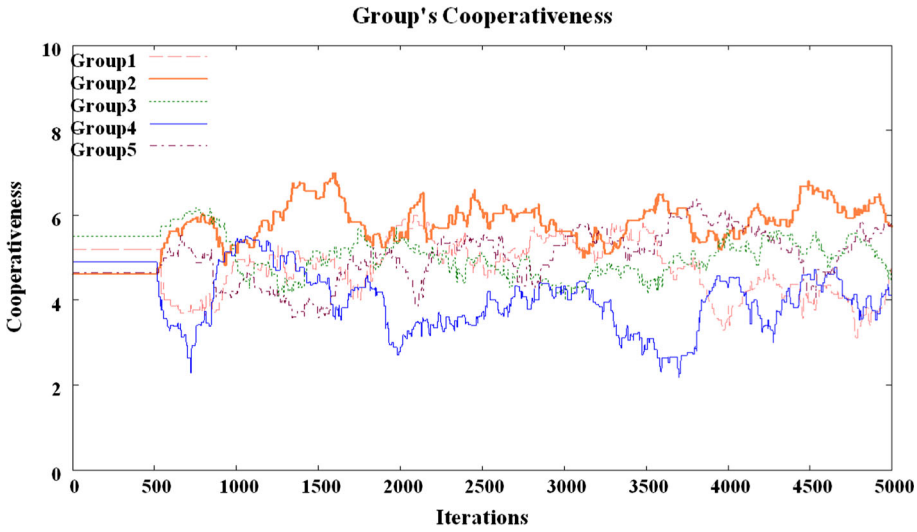

Fig. 4 Group history mechanism with memory of all previous groups

Table 2 Previous group history

Group No	Iteration No	Cooperativeness
1	560	4.5
3	700	6.0
2	1,200	6.4

Table 3 Latest available information

Group No	Iteration No	Cooperativeness
5	1,330	8.1
3	1,170	7.5
2	1,200	6.4
1	1,199	3.8

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 when that group's cooperativeness was 6 and group 2 at 1,200th iteration when that group's cooperativeness was 6.4. Since the composition of groups invariably change over time, the cooperativeness of any group will change as time progresses. So it is likely that the recent information will be more accurate reflection of the cooperativeness of those groups. 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 get to see which agent has moved into this group recently from other groups. Taking into consideration the most recent information available, the agent decides where to move. For example, assuming the current iteration is 1,400, 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 (group's calculated cooperativeness) at that moment. From the latest information, agent L knows about other groups and their cooperation values. 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 do not allow entry into their groups, 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 count is reset to 0.

On the other hand, an agent may not be good enough for its current group—it is being shunned by the other members for being the worst member of its group. Because of play rejections, its wealth will not improve, and it will want to leave and find some other group in which it can find players to play with. If the better groups do not allow entry, the agent will go to lower and lower groups, since it is better off moving to any new group rather than staying in the current group where it is known as the worst player and therefore shunned.

8.4 Entry criteria

How a player gets entry to another group (entry criteria) is as explained in Sect. 5.7.

The entire process is repeated for many iterations in this configuration and the separation of groups is observed. The overall process of this experiment is outlined in Algorithm 4.

Algorithm 4: Pseudocode for sharing group history mechanism.

```
1  begin
2      initialisation;
3      bootstrap agents in groups;
4      foreach iteration do
5          select random number of agents;
6          if iteration is less than 500 then
7              foreach selected agent do
8                  play with another agent in the group;
9                  collect payoff;
10                 gossip;                                          // Algorithm 1
11             end
12         else
13             foreach selected agent do
14                 play with another agent based on gossip;         // Algorithm 2
15                 collect payoff;
16                 gossip;                                          // Algorithm 1
17                 if selected agent's tolerance level is exceeded then
18                     collects recent information about other groups from group
                       members;
19                     selected agent decides to leave the group;
20                     attempts to join another group under certain condition;
                       // Algorithm 3
21                 end
22                 if selected agent's wealth has not increased then
23                     collects recent information about other groups from group
                       members;
24                     selected agent decides to leave the group;
25                     attempts to join another group under certain condition;
                       // Algorithm 3
26                 end
27             end
28         end
29     end
30 end
```

8.5 Results of experiments

We present our results in Fig. 5. Initially, all the five groups were randomly seeded and started with roughly similar average cooperativeness values among their members. They ended up showing a separation among the groups based on their cooperativeness. Group 4 contained mostly the best cooperators, groups 2 and 3 had mostly non-cooperators, and groups 1 and 5 had moderate ones.

This partitioning was achieved using social mechanisms without central control. A demo video can be seen on UniTube [see (Savarimuthu 2010) for the link]. In the video, the larger green circles represent groups. Agents are colour coded based on their cooperativeness which are represented by smaller circles inside the larger green circle. The range of cooperativeness values and their colour codes are 0–2 is Red, 3–4 is Orange, 5–6 is Yellow, 7–8 is Green and 9–10 is Blue. In the beginning, there were mixed colours in all five groups. At the end, the agents are observed to have self-organised themselves into different groups based on their cooperativeness (showing predominantly different colours).

A paired-samples *t*-test was conducted to compare the separation of groups based on cooperativeness (standard deviation of cooperativeness of groups) at the start and end of the runs. The paired *t*-test was performed with null hypothesis (for 30 sample runs) that there is no significant difference between the standard deviations at the start and the end of the experiments. The standard deviation of the

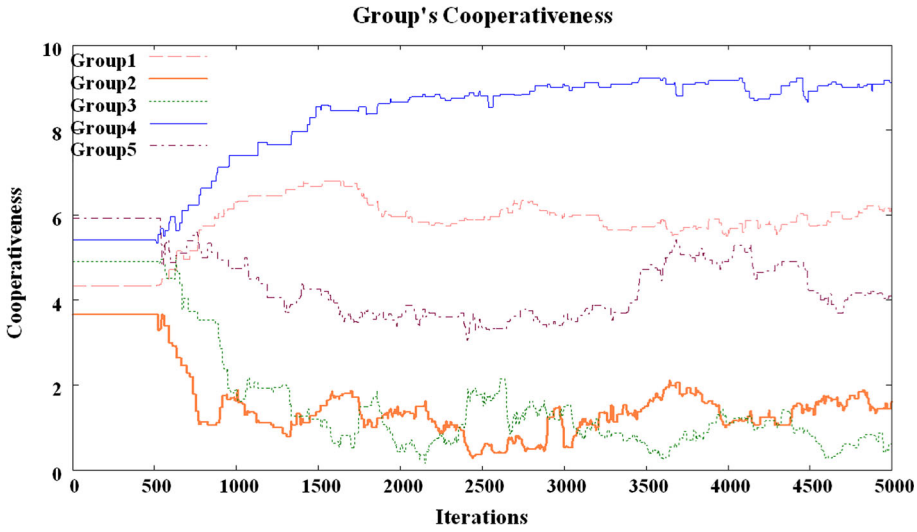

Fig. 5 Self-organisation of groups using sharing group history mechanism

groups' cooperativeness at the start and end of the run was measured. There was a significant difference in the values at the start ($M = 0.71$, SD = 0.25) and at the end ($M = 3.27$, SD = 0.31) conditions. The average difference between the mean values ($M = 2.55$, SD = 0.07, $N = 30$) was significantly greater than zero, $t = 33.77$, one-tail $p = (3.89 \times 10^{-25})$, providing evidence that our mechanism is effective in producing the separation of groups based on cooperativeness with a 95 % confidence interval.

9 Future work

There are many aspects that were outside the scope of the present work. We consider the following for our future work.

- **Consideration of agents changing behaviour:**
  In this work, we have not considered agents changing their behaviour in their life time. In real life, bad agents may redeem themselves or may be forced to cooperate through institutional punishment mechanisms. In our future work, we intend to examine more advanced situations in which agents can dynamically alter their cooperation strategies. That would mean that a peer could start with a certain cooperative value, but later based on the circumstances (e.g. based on learning), decide to change its behaviour. For example, an agent could try to enhance its performance by becoming a "bad guy" temporarily and then returning to being a "good guy", since it may estimate that its potential rewards could be even higher because of occasional cheating in a good group of agents. Such behaviour changing mechanisms can be investigated in the future.
- **Lying problem:**
  The systems investigated in this work make use of recommendations from other agents to decide whether to interact with another agent or not and also to know the performances of other groups, relying on the fact that the other group agents are honest in revealing the information about their group (i.e. these agents do not lie). However, this may not be the case in general. Agents being autonomous (and intelligently self-interested) may not want to share their group information (e.g. cooperativeness of the group) with outsiders and, worse still, may lie when such information is requested. This behaviour may lead to an undesired state of affairs in a society. Additionally, bad agents can spread false gossip which may also have deleterious effect in segregation of groups. Our intention is to examine these issues in our future investigations.
- **Varying gossip type:**
  In the future, we intend to examine the types of gossip in the system to determine conditions under which the gossip mechanism leads to separation of groups and conditions under which it does not lead to the separation (i.e. by experimenting with different types of gossip (e.g. about other groups, about other agents' cooperativeness, about other agents' trustworthiness in providing gossip information))

10 Summary

In this work, we have explored mechanisms for shared-resource policy management which are suitable for closed societies. We developed mechanisms for self-organisation

or separation of groups based on degrees of cooperativeness.

- The dynamic grouping mechanism achieves self-organisation and it has been enhanced for systems to set entry values dynamically considering the current system state.
- The random hopping mechanism does not lead to separation of groups because of random hopping and agents not settling.
- The group history mechanism also does not lead to separation because of lack of latest information about other groups.
- The sharing group history mechanism leads to self-organisation or separation of groups based on cooperativeness. Agents share their information about other groups; hence, the latest information is available.

The final system presented here is more suitable for sharing digital goods in P2P systems, where the aim is to restrict freeriding. It has produced self-organisation, or the so-called self-balancing of P2P systems, in a distributed and dynamic manner in closed societies. This system set-up takes advantage of social mechanisms, such as tagging to form groups, gossip to pass information and ostracism to shun uncooperative behaviour. As a result, it shows the self-organisation of groups based on behaviour (cooperativeness) in closed societies, without any control at the top level.

Overall, this work offers new socially inspired mechanisms that offer solutions towards restricting exploitation of freeriders in artificial societies through the segregation of groups. These mechanisms result in the improvement of the overall societal performance in a society that has both cooperative and uncooperative agents. We believe some of the mechanisms developed here in this work can be applied effectively to future ICT systems.

References

Bezos J (1995) Amazon homepage. http://amazon.com

de Pinninck AP, Sierra C, Schorlemmer M (2008) Distributed norm enforcement: ostracism in open multi-agent systems. In: Casanovas P, Sartor G, Casellas N, Rubino R (eds) Computable models of the law, languages, dialogues, games, ontologies, volume 4884 of lecture notes in computer science. Springer, pp 275–290

Dunbar R (1996) Grooming, gossip and the evolution of language. Faber and Faber, London

Dunbar R (2004) Gossip in evolutionary perspective. Rev Gen Psychol 8(2):100–110

Esteva M, Rosell B, Rodrguez-Aguilar JA, Arcos JL (2004) AMELI: An agent-based middleware for electronic institutions. In Proceedings of the third international joint conference on autonomous agents and multiagent systems, vol 1,236–243, AAMAS,IEEE Computer Society,Washington, DC, USA

Eugster P, Felber P, Le Fessant F (2007) The "art" of programming gossip-based systems. SIGOPS Oper Syst Rev 41(5):37–42

Feldman M, Chuang J (2005) Overcoming free-riding behavior in peer-to-peer systems. ACM Sigecom Exch 5:41–50

Gursel A, Sen S, Candale T (2009) Stability in referral systems. Multiagent Grid Syst 5(1):19–36

Hales D (2004) Self-organising, open and cooperative P2P societies: from tags to networks. In: Brueckner S, Serugendo GDM, Karageorgos A, Nagpal R (eds) Engineering self-organising systems, volume 3464 of lecture notes in computer science. Springer, pp 123–137

Krishnan R, Smith MD, Tang Z, Telang R (2004) The impact of free-riding on peer-to-peer networks. In Proceedings of the 37th annual Hawaii international conference on system sciences (HICSS'04)–Track 7–vol 7, HICSS '04, pp 199–208. IEEE Computer Society, Washington, DC

Milinski M, Semmann D, Krambeck H-J (2002) Reputation helps solve the 'tragedy of the commons'. Nature 415(6870):424–426

Morgan S (1999) Trademe homepage. http://trademe.co.nz

Omidyar P (1995) EBay homepage. http://ebay.com.

Paine R (1967) What is gossip about? an alternative hypothesis. Man 2(2):278–285 URL:http://www.jstor.org/stable/2799493

Paolucci M, Marsero M (2000) Rosaria conte what is the use of gossip? a sensitivity analysis of the spreading of respectful reputation. In: Tools and techniques for social science simulation. Physica, pp 302–314

Purvis M, Savarimuthu S, De Oliveira M, Purvis M (2006) Mechanisms for cooperative behaviour in agent institutions. In: Proceedings of the IEEE/WIC/ACM international conference on intelligent agent technology. EEE Computer Society, Washington, DC, pp 121–124

Ramaswamy L, Liu L (2003) Free riding: a new challenge to peer-to-peer file sharing systems. In Proceedings of the 36th annual hawaii international conference on system sciences, HICSS 2003, Volume 7. IEEE Computer Society, Los Alamitos, CA, pp 220–229

Roberts JM (1964) The self-management of cultures. In: Goodenough WH (ed) Explorations in cultural anthropology:essays in honor of george peter murdock. McGraw-Hill, NewYork, pp 433–454

Savarimuthu S (2010) Self-organising groups (GUI for closed society). University of Otago, http://unitube.otago.ac.nz/view?m=9GT31pqTPSk

Savarimuthu S, Purvis M, Purvis MK (2009) Self-organization of peers in agent societies. In Proceedings of the IEEE/WIC/ACM international joint conference on web intelligence and intelligent agent technology, vol 02, pp 74–77. IEEE Computer Society, Washington, DC

Savarimuthu S, Purvis M, Purvis M, Savarimuthu BTR (2010) Mechanisms for the self-organization of peer groups in agent societies. Multi-agent-based simulation xi: international workshop–volume 6532 of Lecture notes in artificial intelligence. Springer, Toronto, pp 93–107

Savarimuthu S, Purvis M, Purvis M (2013) Bastin tony roy savarimuthu. Gossip-based self-organising agent societies and the impact of false gossip. Minds Mach, Philosophy and Cognitive Science. ISSN 0924–6495. doi:10.1007/s11023-013-9304-8

Skyrms B (2009) Groups and networks: their role in the evolution of cooperation. In: Levin SA (ed) Games, groups, and the global good, springer series in game theory. Springer, Berlin, pp 105–114

Skyrms B, Pemantle R (2000) A dynamic model of social network formation. Proc Natl Acad Sci 97(16):9340–9346

Sommerfeld RD, Krambeck H-J, Semmann D, Milinski M (2007) Gossip as an alternative for direct observation in games of indirect reciprocity. Proc Natl Acad Sci USA, vol 104, No. 44 (Oct. 30,

2007), pp. 17435–17440, Published by: National Academy of Sciences, Stable URL:http://www.jstor.org/stable/25450253

Stirling RB (1956) Some psychological mechanisms operative in gossip, social forces, vol 34, No. 3 (Mar., 1956), pp 262–267, Published by: University of North Carolina Press, URL:http://www.jstor.org/stable/2574050

Suls JM (1977) Gossip as social comparison. J Commun 27(1):164–168. doi:10.1111/j.1460-2466.1977.tb01812.x

Thomsen R (1972) The origins of ostracism, a synthesis. Gyldendal, Copenhagen

Yolum P, Singh MP (2005) Engineering self-organizing referral networks for trustworthy service selection. IEEE Trans Syst Man Cybern A 35(3):396–407