

Beating Cheating: Dealing with Collusion in IPD

N. Höning^a T. Kozelek^b M.C. Schut^a

^a *VU University, Amsterdam*, {nhg400,mc.schut}@few.vu.nl

^b *Charles University, Prague*, tomas.kozelek@gmail.com

Abstract

The Iterated Prisoner's Dilemma (IPD) is a well-known challenging problem for researching multi-agent interactions in competitive and cooperative situations. In this paper, we present the Ask-First (*AF*) strategy for playing multi-agent IPD that is based on evolving trust chains between agents. Each agent maintains a (relatively small) table containing trust values of other agents. When agents are to play each other, they ask their neighbours what trust they put in the opponent. Chains are then followed until an agent is found that knows the opponent and the trust value is propagated back through the chain. The played move is then decided based upon this trust value. When two agents have played each other, they update their trust tables on the basis of the outcome of the game. The strategy is first evaluated in a benchmark scenario where it is shown that it outperforms a number of benchmark strategies. Secondly, we evaluate the strategy in a scenario where a group of colluding agents is organised like a cartel. The experiments show that the *AF* strategy is successful here as well. We conclude that the *AF* strategy is a highly flexible, scalable and distributed way (the chain topology adapts to the way that agents are picked to play each other) to deal with a difficult multi-agent IPD problem (i.e., robust against collusions).

\$Id: paper.tex 95 2008-06-25 13:38:02Z schut \$; last local update @ 2008-06-25 16:51

1 Introduction

In open e-commerce systems (e.g., an electronic market), *collusion* is a serious threat to the honest operation of the system. Collusion is the agreement between two or more persons to deceive others to obtain a secretive (or: illegal) objective. On eBay, this happens by so-called *shill bidding*, where friends of the seller bid on an item solely with the intention to raise the price. These friends can even be other eBay accounts of the seller himself and not necessarily real persons. Another example of collusion happens when in recommender systems, items are evaluated positively by the seller's friends (which may, again, be fake accounts of the seller). In such situations, it is very important for a potential buyer (who is truthful) to judge the reliability of the bids or the recommendations. Ideally, one wants to achieve this without having to rely on a trusted third-party or authority.

In this paper, we suggest a way to do exactly this based on the idea of *trust networks*. In such networks, agents maintain the trust that they have in each other and communicate this with each other when necessary. The intuition behind this method is that in real life, we ask our friends how they feel about a certain item

or some particular shop. This intuition has led us to develop the very simple Ask-First (*AF*) strategy that faithful agents can employ to protect them from malicious or unreliable agents.

Within the context of Iterated Prisoner's Dilemma (IPD), the strategy works as follows (we provide more detailed information later in the paper). Firstly, an *AF* strategy agent maintains a (relatively small) table containing trust values of other agents. Then, when agents are to play each other, they ask their neighbours what trust they put in the opponent. Next, chains are then followed until an agent is found that knows the opponent and the trust value is propagated back through the chain. Then, the played move is then decided based upon this trust value. Finally, when two agents have played each other, they update their trust tables on the basis of the outcome of the game.

The research objective of this paper is two-fold: 1) to show that the *AF* agents outperform the benchmark agents (all-cooperators and all-defectors); and 2) to show that the *AF* agents outperform collusion agents. For objective 2), we also want to show that the *AF* agents use the created network chains effectively and that colluding agents cannot invade these chains.

This paper has the following structure. In Section 2, we present the background for our work: computational trust and reputation, referral networks and social networks. We also explain the non-Iterated Prisoner's Dilemma. Section 3 lays out the details of our framework and the Ask-First (*AF*) strategy. In Section 4 we show by a number of experiments the effectiveness of the *AF* strategy. Finally, in Section 5, we draw conclusions about the performed study and provide some pointers for future work.

2 Background

The work presented in this paper, builds further on research on computational trust and reputation, referral networks, and social networks. In this paper, we place all within the context of the Iterated Prisoner's Dilemma (IPD). In this Section, we briefly touch upon each of the topics.

2.1 Trust, Reputation and Referral

The concept of *trust* is essential in societies and open systems in order to maintain 'good' social interactions and commitments between individuals. As mentioned above, on eBay, you want to trust the person from whom you are buying your items. In real life, we have many (albeit implicit, unconscious) mechanisms operating for managing trust; in virtual life (internet), we have not yet established such mechanisms [1]. Much research in the last decade has been dedicated to the question how to use trust as a mechanism for regulating social interaction. In the early 1990s, Marsh [10, 9] presented a formalisation of trust that pins down a number of defining properties of trust in order to facilitate a precise discussion about trust. The formalism was implemented for a multi-agent system in a PD scenario, demonstrating a recognisable behaviour of trust among the agents.

An often-used mechanism for managing trust is by means of *reputation*: a societal indication of how much you can trust someone, reflecting its past actions. Like the concept of 'trust', 'reputation' is a convoluted term and needs to be clarified unambiguously for precise discussion and application; Mui *et. al* [12] do this by giving a concise overview of the notion within the context of multi-agent systems. In terms of applications, Chadwick [4] provides a detailed classification system for reputation systems. Current major websites (eBay, Amazon) have *centralised* reputation mechanisms in place, where people's ratings about each other are collected, processed and communicated back. However, such centralisation of reputation is not always possible, e.g., for peer-to-peer service provision [6, 19]. It can be expected that future systems will become increasingly more open and require such decentralised mechanisms for maintaining trust and reputation.

A recent development in the search for “decentralised reputation management” concerns so-called *referral networks*. In such networks, agents communicate information about trust and reputations are built based on this information. The networks are used for query-based searching for information and expertise in a person’s social network [7]. In a referral network, nodes have neighbours (whom they query directly) and acquaintances (whom they query only when referred to); both sets are dynamic (neighbours can become acquaintances and vice versa) and usually limited. Studies of referral networks by Singh *et. al* [20, 18] have looked at how network structure evolved under various circumstances; how agent learning models (enabling agents to learn about each other in terms of expertise – producing correct answers, and sociability – providing good referrals) affects the quality of the work; and how to design self-organising referral networks.

2.2 Social Networks

The basis of the referral networks described above is the *social network* that connects individuals within a collective system. In such a network of individuals, there can be links between these individuals that represent, for example, friendship, kinship, values. Social network analysis views these networks as graphs, in which the nodes are individuals and the edges are the links. Within the context of this paper, research on social networks includes, for example, the study of decentralised search algorithms [8] and investigation of the relationship between social network topologies and emergent behaviour [2].

A particular stream of research worth mentioning is the investigation of so-called *small-world networks*. Since the formalisation of the small world problem (originally coined by Milgram [11, 16]) by Watts and Strogatz [17] in the nineties, much research on social networks looks at such networks being small-world networks: it takes relatively few steps to go from one random node to another. This type of networks has two important properties: a short average path length (hence, the relatively few steps to reach other nodes), and a high clustering coefficient (number of a node’s neighbours that also know each other). The experiments presented later in this paper also consider small-world networks.

2.3 Non-Iterated Prisoner’s Dilemma

The well-known Iterated Prisoner’s Dilemma (IPD) is a generic abstract representation of complex social dilemmas [3]. It is also used for the game-theoretic modeling of multi-agent systems (e.g., [15]). It is a (non-zero-sum) game where two players make decisions simultaneously whether to defect or cooperate and receive rewards based on the combination of their two actions. The ordinance of the payoffs makes this game a dilemma: $[dc] > [cc] > [dd] > [cd]$, where $[xy]$ denotes the reward for player 1 where player 1 plays x , player 2 plays y and $x, y \in \{c(\text{cooperate}), d(\text{effect})\}$. The dominant strategy for the one-shot PD (where the game is played once) is for both players to defect. The iterated version of the game (where it is played for a number of times) does not have a single dominant strategy. In a competition-based evaluation of the IPD, it was found that an extremely simple strategy beat the other strategy: *tit-for-tat*, where the opponent’s last action was simply copied.

In this paper, we have a population of agents that repeatedly plays pairwise PD games. Note, however, that it is not IPD, because the agents cannot remember whom they played. Like in [5, 14], we call this version the *non-Iterated PD* (nIPD).

While in the one-shot and iterated PD it is almost certain that the players will ultimately defect each other, in the nIPD this is not the case at all. Much research has been done on the *evolution of cooperation* in nIPD, addressing the important question how cooperation can arise between selfish agents. Nowak and May [13] showed that in a two-dimensional cellular automata (where a cell could be in a ‘cooperate’ or ‘defect’ state) if the PD was used as the update rule, patterns of cooperation and defection emerged. In follow-up

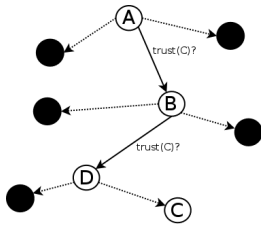


Figure 1: Agent A (an AF agent) is about to play against agent C. An information chain is built over agents B and then D, who finally knows C. The trust D has in C will be reported back to A, multiplied by the trust values of B in D and A in B. If the value that is reported to A exceeds the model’s *chain trust threshold*, he will use it for his decision whether to cooperate with C (if it doesn’t exceed the model’s *trust threshold*, he will defect). Otherwise, he will use his default behaviour.

work by Olifant [14], it is shown that in a society of agents, *spatiality* plays an important role: agents that are close by each other are more likely to play than those that are further away. This work gives rise to the issue of how agents are connected to each other: can the structure of the connection network affect the evolution of cooperation?

In more recent work, Ellis and Yao [5] address this issue by looking at a *social network* inspired approach for the evolution of cooperation. In their (non-spatial) nIPD, links are formed between cooperating agents. These links are reinforced by repeated cooperation, while defection breaks a link. All links taken together represent the social network of the agents. Ellis and Yao present a strategy that can exploit this network: the *discriminator* agent bases its move (cooperate or defect) on the *centrality* of its opponent. (This centrality represents the agent’s reputation among its neighbours.) The experiments for this strategy showed that the discriminator strategy is more successful than all-cooperators or all-defectors. In another series of experiments, agents are able to evolve strategies while playing – ranging from all-cooperate to all-defect. The evolving parameters included 1) the probability that an agent interacts with another agent that has a lower reputation than some threshold value, and 2) this threshold. The experiments showed that when agents are able to observe the centrality, then the population evolves to all-cooperators.

3 Model

In our model, agents play games of the Prisoner’s Dilemma with opponents in a Small World Network¹ and accumulate payoff. In principle, agents can be heterogeneous and act autonomously. Every ϵ th round, we perform an evolutionary replacement on the population to translate payoff success into numerical representation in the population.

3.1 Communication

Our model is distributed - agents do not need to contact a central authority. Instead, agents can use a communication protocol and try to ask agents they know (have in their trust table) if they know the opponent. If they don’t, they can in turn ask someone they know. We call the resulting communication path an *information*

¹Opponent selection is partly randomised, and subject to the Small World Network building process. See section 3.5 for more details.

chain.

If agent A knows agent B, then agent B can be asked by A about the trustability of a particular opponent C. If agent B doesn't have information on agent C in his trust table, then he will in turn ask another agent D that he knows. See figure 1 for an illustration. Unlike in the work of Singh *et. al* [19?] this is a recursive process, not an iterative one. In order to keep network communication low, the maximal number of hops along such a chain is bound. If it is not possible to find any agent that knows the opponent in 6 steps (in relation to the famous "six degrees of separation" [?]), we consider the chain as not buildable. Furthermore, each agent is allowed to ask only one other agent. Both of these restrictions are done purely for practical reasons. In this work, agents decide who they should ask for information by evaluating how much they trust their friends and how far away they are from the opponent. For example, if $d(D, C)$ denotes the distance from D to C and $trust(B, D)$ denotes the trust that B has in D , then agent B evaluates D by maximising the formula

$$distance_bias * d(D, C) + trust_bias * trust(B, D)$$

This formula balances a tradeoff between network effect and fraud safety. The higher the bias on distance, the more probable it is that the opponent will be reached soon. On the other hand, high bias on trust might produce more reliable chains but it is also more likely to fail in finding the opponent at all.

3.2 Distance

The agents are ordered along a circular likeness metric, so a distance measure $d(A, B)$ is an abstract measure of how close agent A is to agent B, i.e. how similar they are. Each agent X is indexed by I_x , where $I_x \geq 1$ and $I_x \leq N$. Then, the distance of two agents A and B is given by

$$d(A, B) = \min(\text{abs}(I_a - I_b), N - \text{abs}(I_a - I_b))$$

$d(A, B)$ is bound by 0 and $N/2$. We normalise the distance measure by $1/d(A, B)$.

3.3 Trust

Each agent keeps a list of up to K agents he knows and stores the trust t he has in them in it, with $t \in \mathbb{R}$ and $0 \geq t \leq 1$. Full trust is denoted by 1 and no trust by 0. Other agents can be added to this trust table after interactions with them and might get deleted when the list is full and a new agent is added in place (see section 3.5). When an interaction is finished, agents update the value for the opponent in their trust table according to

$$\mathbf{if} (cooperation) \mathbf{then} new_trust = old_trust + \frac{(1 - old_trust)}{2}$$

$$\mathbf{if} (defection) \mathbf{then} new_trust = old_trust - \frac{(old_trust)}{2}$$

This results in a sigmoid curve. If the opponent changes his behaviour, it will have a quick effect on the trust in him. This follows recommendations by Axelrod [3] in that successful strategies should quickly retaliate and quickly forgive. If an agent is removed during an evolutionary update, he is removed from all trust tables.

3.4 Strategies

In our scenarios, we employ some simple strategies to evaluate their interplay. If not mentioned otherwise, all agents can be asked to build up information chains and report their true trust evaluations about agents they know.

Always Cooperate (AC) This strategy simply cooperates in every interaction.

Always Defect (AD) This strategy always defects the opponent.

Ask First (AF) This is the strategy that builds up information chains and makes use of them. AF agents ask other agents about their opponents. In the tradition of Axelrod's research [3], agents with this strategy will default to cooperation if no information on the opponent is accessible or trusted enough to be used. When information from a chain is used, the resulting success has no influence on the trust that an AF agent has in its information source².

Simple Collusion (SC) To put the AF strategy to a test against fraudulent agents, we designed a simple collusion strategy. These agents will behave like AD, i.e. they will defect in each interaction. But when asked about the trust value of another agent A that they actually know and reference with the value $trust_A$ in their trust table, they will return $trust'_A = 1 - trust_A$. This means that they will give each other a good reputation and agents that generally cooperate a bad reputation. If AF agents make use of this information, they might be misled in their actions.

3.5 Network Structure

In the beginning of our simulations, there are no connections between agents. We would like to evolve a Small World Network through interactions. Such a structure can be characterised by low average path lengths L (comparable to L of a random graph of the same size) and a high clustering coefficient C (much higher than C of a random graph). In such a structure, each agent should be able to build a short information chain connecting him to the desired opponent. In particular, we set two goals for a network of size N :

1. The information about neighbours that every single agent maintains should be small ($O(\log N * \sqrt{\log N})$)
2. Retrieving information from the network should be fast ($O(\log N)$ or constant - this is a property given in Small World Networks)

The default way of building a Small World Network structure is to mostly connect neighbouring agents, and sometimes (according to some probability) remote agents [17]. In this model, we follow a more precise approach by Kleinberg [8] to evolve it using a distance metric. This metric works in the sense that agent A should have a higher probability to play against or know agent B than agent C if the distance $d(A, B)$ is lower than $d(A, C)$. Kleinberg calculated that probability with $\frac{1}{d(A, B)^\alpha}$, with the optimal setting for α to evolve a Small World Network at around 2 (high values for α make locality more important). To pick a distance d from all possible distances (with $d \in \mathbb{N}$ and $1 \leq d \leq \frac{N}{2}$), we use a probability vector p , built with

²An AF agent A could react to the information quality, i.e. when he cooperated with agent C because agent B told him C could be trusted, A could lower his trust in B . In this model, however, he does not do so.

Kleinbergs formula:

```

 $p_0 = 0$ 
for  $d$  in  $1 \dots \frac{N}{2}$  do {Kleinberg initialisation}
     $p_d = \frac{1}{d^\alpha}$ 
end for
for  $d$  in  $1 \dots \frac{N}{2}$  do {normalisation}
     $p_d = p_d / \sum_{d'=1}^{\frac{N}{2}} p_{d'}$ 
end for
for  $d$  in  $1 \dots \frac{N}{2}$  do {make sequence converge to 1}
     $p_d = p_d + p_{d-1}$ 
end for

```

In p , we have a list of $\frac{N}{2}$ values that converge to 1. For example, for $N = 10$ and $\alpha = 2$, we have

$$p = [0, 0.7024, 0.878, 0.956, 1]$$

If we choose a random number r with $r \in \mathbb{R}$ and $0 \geq r \leq 1$, we can then choose the the distance as the biggest d so that p_d is still below r .

Our model needs this distance metric for two purposes. One is the building of the network structure: When agent X needs an opponent, our model selects a distance d as described. Then, the index of his opponent for X is $N\%(I_X + d)$. Furthermore, agents need to maintain the network structure by deciding who to keep in their trust table (since the size of the trust tables is limited by K). When for agent X an opponent O is added to the trust table and the size of X s trust table thereby grew bigger than K , X selects a distance d . If the distance from X to O is smaller than d (i.e. O is close to X), a random agent is deleted from X s trust table. Otherwise, O is deleted (because is not very similar to X).

When the same value for α is used for both purposes, it will often happen that agents already have their current opponents in their trust table. To resemble a more realistic situation, our model uses two different values, α and β . We use less bias to locality when building up the network (α , around 1.0) and more bias to locality when agents decide who their closest friends are (β , around 2) - i.e. to decide who to keep in the list of neighbours. This most likely resembles the real world.

The overall goal behind these methods is to evolve and keep a Small World Network structure. In every simulation we ran, the network parameters achieved satisfying values. Small World Networks have $L \simeq L_{random}$, but $C \gg C_{random}$ (see [17] for details). The table below shows the network parameters at the end, averaged over all our simulations.

	L	L_{random}	C	C_{random}
$N = 150, K = 13$	2.46	2.19	0.30	0.08
$N = 150, K = 21$	2.03	1.89	0.29	0.13
$N = 300, K = 16$	2.62	2.33	0.29	0.05
$N = 300, K = 25$	2.20	2.01	0.26	0.08

4 Experimental Evaluation

4.1 Scenario

Our simulations proceed as follows:

```

Initialise N individuals
Initialise the world: According to the scenario, assign strategies uniformly to randomly picked agents
for  $r$  in  $1 \dots N * 4$  rounds do
  for  $i$  in  $1 \dots N$  do
    Select an opponent  $j$  to play against agent  $i$ 3.
    Agents  $i$  and  $j$  try to find out if their opponent can be trusted (if they are AF agents).
    Agents  $i$  and  $j$  decide whether they cooperate or defect.
    Agents  $i$  and  $j$  receive payoffs and update their trust tables according to their opponents' action.
  end for
  if  $(N * 4) \% e == 0$  then
    Evolutionary update: Select two agents randomly and replace the one with lower average payoff
    with a new agent that follows the strategy of the one with higher average payoff. Delete the removed
    agents from trust tables.
  end if
end for

```

4.2 Design and Setup

We used a *simple comparative* experimental design. The simulations were run 20 times for each experimental setting.

Independent Variables

variable	possible values
N: # of individuals	[150, 300]
K: # of neighbours	$[0.7 * \log N * \sqrt{\log N}, 1.1 * \log N * \sqrt{\log N}]$
α : network	[0.9, 1.2]
β : network	[1.9, 2.4]
strategy scenarios	$(\frac{1}{2}AD, \frac{1}{2}AF),$ $(\frac{1}{3}AC, \frac{1}{3}AD, \frac{1}{3}AF),$ $(\frac{1}{2}AC, \frac{1}{2}AF)$
bias to distance when asking	[0.3, 0.7]

Dependent Variables

variable	comment
L	average path length
NP	# of not connected pairs of agents
C	clustering index of network
SN	# of agents of particular strategies
SP	average payoff of the strategy

In effect, we had $2 * 2 * 2 * 2 * 3 * 2 = 96$ different settings.

Fixed Variables

³In each round, we choose an opponent once for each agent and after the game, both opponents will remain in the pool. This guarantees evenly allocation of opponents. In effect, each agent plays at least one game in each round, probably more (for the selection procedure, see also section 3.5).

variable	value
limit of steps in asking	6
chain trust threshold	0.3
opponent trust threshold	0.5
# of epochs	4 * N
e: rounds until evol. update	3
bias to trust when asking	1 - dist. bias
steepness of trust update	2

The payoff matrix is also a fixed variable. It is the standard payoff matrix for a game of Prisoner's dilemma, taken from [3]:

	A	B	A's payoff	A+B payoff
traitor benefit (t)	D	C	5	5
reward (r)	C	C	3	6
punishment (p)	D	D	1	2
sucker payoff (s)	C	D	0	5

4.3 Results

We conducted three experiments, according to the three strategy scenarios listed as independent variables above. In each experiment, all the other independent variable settings were included.

1. The first experiment consisted of 50% AD agents and 50% AF agents. In figure 2, we show the performance of both strategies in the settings with 150 agents (a) and the resulting overall Utilitarian Social Welfare (b), which is calculated by adding up the payoffs all agents achieved in the current round.
2. In the second experiment, we had $\frac{1}{3}$ AD agents, $\frac{1}{3}$ AC agents and $\frac{1}{3}$ AF agents. In Figure 3, we show the performance of the three strategies in the settings with 150 agents, once with normal runtime (a) and once with a runtime of $20 * N$ (b).
3. The third experiment employed 50% AF agents and 50% SC agents. In figure 4, we show the performance of the two strategies in the settings with 150 agents (a) and some information about the number of information chains built and used (b).

4.4 Analysis

This section reviews our research hypotheses and states in what manner we have achieved them.

- **Hypothesis 1: The AF strategy is successful against defection.**

The simple explanation is that this doesn't necessary hold. Simply averaged over all configurations, the AF strategy achieves to represent 55.8% of the population at the end of the simulations. That is not a bad value. If we look closer, we see that the success depends on the particular population scenario.

If we only look at strategy experiment 1, AF versus AD, we find that AF wins by far. We conducted a Welsh Two-Sample T-Test ($t = 51.1728, df = 712.562, p\text{-value} < 2.2e - 16$) to test on the difference of the population representation of both strategies in the end of runs in experiment 1. These results show that significant differences appeared here (the mean population size of AF was 183.99 and the mean population size of AD was 28.41). Figure 2(a) shows the differences over the whole runs. It is interesting to note that if a cooperating strategy like AF takes over the population, this is not only self-beneficiary. Figure 2(b) shows how the overall production of the agent society, the Utilitarian Social Welfare, gets maximised as AF agents take over the population. The high spike in the beginning results from the initially trusting behaviour of AF agents facing unknown AD agents. Quickly, AF agents learn not to trust AD agents and spread that information via information chains to other AF

agents.

But if we look at these two strategies in the 2nd experiment, where also an always-cooperating strategy was involved, the picture is different. The AF strategy loses to the AD strategy (see Figure 3(a)). We conducted another Welsh Two-Sample T-Test like experiment 1, but regarding the second scenario ($t = 28.7613$, $df = 1024.961$, $p\text{-value} < 2.2e - 16$). Here, the AD agents will significantly outnumber the AF agents (the mean population size of AF was 63.47 and the mean population size of AD agents was 142.78. We note that the AF population stays at a stable level.

We concluded that defecting strategies profit much more from the presence of “dumb”, often-cooperating strategies than the AF strategy. In the context of the Prisoners Dilemma, this means that the AF strategy does not exploit cooperators as much as defectors do. The AF strategy should recover once pure cooperators die out (and then the setting is the same as in scenario 1). Figure 3(b) shows the results from reruns of the simulation with a longer runtime ($N * 20$). Indeed, AF agents take over in the long run when AD agents cannot feast on AC agents any longer.

- **Hypothesis 2: The AF strategy is successful against collusion.**

This hypothesis holds. We conducted a Welsh Two-Sample T-Test ($t = 188.57$, $df = 318$, $p\text{-value} < 2.2e - 16$) to test on the difference of the population representation of both strategies in the end of runs in experiment 3 (AF, SC). The results show with strong significance that the AF strategy will outnumber the SC strategy after $N * 4$ runs (see Figure 4 (a)). The mean of AF was 136.07 and the mean of SC was 13.93.

- **Hypothesis 3: The collusion strategy cannot spread its false information along the information chains.**

To understand the setting of AF agents against SC agents better, we monitored the use of information chains. In particular, we protocolled the number of chains that were successfully built (i.e. the original asker, an AF agent, got an answer of someone who had his opponent in his trust table), the number of information chains that were actually used (i.e. the returning trust information exceeded the chain trust threshold) and the number of the used chains which only contained AF agents. Figure 4(b) shows that AF agents will use exclusively use chains that only consists of other AF agents. This may seem strange at first, but it is a consequence of the design. Defecting agents will seldom be chosen for building the next step in a chain as they quickly rank low in trust at all AF agents, due to the retaliating nature of the AF strategy. If they are chosen, they lower the overall returned trust, making it highly unlikely that the asking AF agent will use the chain for his decision. In this sense, the AF strategy could be called unforgiving: Defecting agents will not be asked for information.

5 Conclusions

We proposed and implemented a distributed reputation system, which evolves a Small World Network. Under various circumstances, we tested the performance of a strategy that is generally forgiving but tries to gather information via the network. We showed some simple cases in which it is successful and which cases can be problematic in the short term. By evolving a Small World Network, we showed that this approach can be computationally feasible.

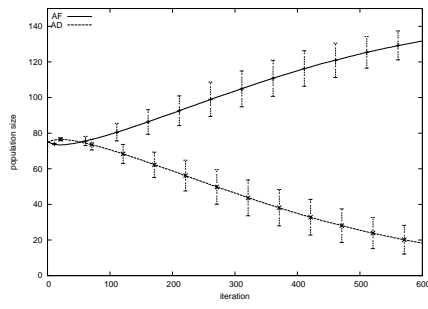
Of course, further research would deal with more various strategy scenarios and experiment with performance against “lying strategies” and possibly cartels. Regarding the distributed nature of the framework,

in which trust information is only passed along local connections, we suspect this to be a fruitful path of research.

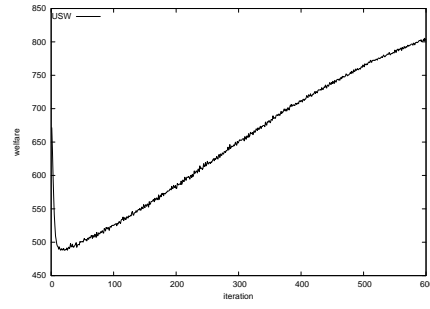
References

- [1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the Hawaii International Conference on System Sciences*, 2000.
- [2] G. Abramson and M. Kuperman. Social games in a social network. *Phys. Rev. E*, 63(3), 2001.
- [3] R. Axelrod. *The Evolution of Cooperation*. Basic Books, New York, 1984.
- [4] D.W. Chadwick. Operational models for reputation servers. In *Trust Management*, volume 3477 of *Lecture Notes in Computer Science*, pages 108–115. Springer, 2005.
- [5] T.S. Ellis and X. Yao. Evolving cooperation in the non-iterated prisoner’s dilemma: a social network inspired approach. In *IEEE Congress on Evolutionary Computation*, 2007.
- [6] A. Josang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43:618–644, 2007.
- [7] Henry A. Kautz, Bart Selman, and Mehul A. Shah. The hidden web. *AI Magazine*, 18(2):27–36, 1997.
- [8] J. Kleinberg. Complex networks and decentralized search algorithms. In *Proceedings of the International Congress of Mathematicians*, 2006.
- [9] S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Computing Science, University of Stirling, 1994.
- [10] S. Marsh. Trust in distributed artificial intelligence. In *Artificial Social Systems*, volume 830 of *Lecture Notes in Computer Science*, pages 94–112. Springer, 1994.
- [11] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.
- [12] L. Mui, M. Mohtashemi, and A. Halberstadt. Notions of reputation in multi-agents systems: a review. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 280–287. ACM Press, 2002.
- [13] M.A. Nowak and R.M. May. Evolutionary games and spatial chaos. *Nature*, 359:826–829, 1992.
- [14] M. Oliphant. Evolving cooperation in the non-iterated prisoner’s dilemma: The importance of spatial organization. In R. Brooks and P. Maes, editors, *Proceedings of Artificial Life IV*, pages 349–352. MIT Press, 1998.
- [15] C. O’Riordan. Forgiveness in the iterated prisoner’s dilemma. In *Proceedings of the UK Workshop on Multiagent Systems*, 2001.
- [16] Travers S. Milgram. An experimental study of the small world problem. *Sociometry*, 32:425–443, 1969.
- [17] J. Duncan Watts and H. Steven Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393, June 1998.

- [18] P. Yolum and M.P. Singh. Engineering self-organizing referral networks for trustworthy service selection. *IEEE Transactions on systems, man, and cybernetics*, 35(3):396–407, 2005.
- [19] B. Yu, M.P. Singh, and K. Sycara. Developing trust in large-scale peer-to-peer systems. In *Proceedings of the First IEEE Symposium on Multi-Agent Security and Survivability*, pages 1–10, 2004.
- [20] B. Yu, M. Venkatraman, and M.P. Singh. An adaptive social network for information access: Theoretical and experimental results. *Journal of the Applied Artificial Intelligence*, 17(1):21–38, 2003.

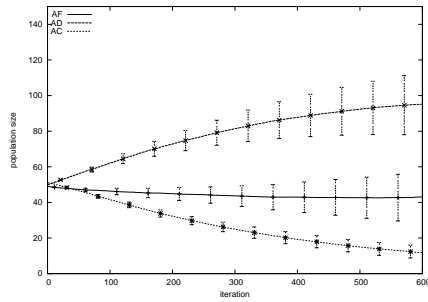


(a) Strategy populations

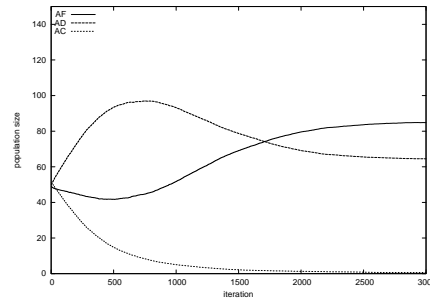


(b) Utilitarian Social Welfare

Figure 2: AD vs AF in experiment 1, (N=150)

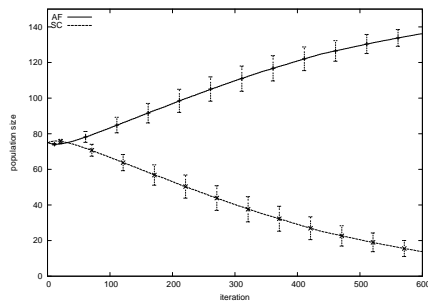


(a) Strategy populations

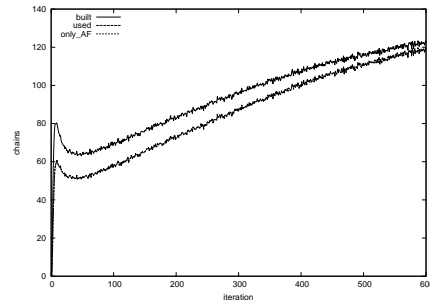


(b) Strategy populations (runtime $N * 20$)

Figure 3: AD vs AC vs AF in experiment 2 (N=150)



(a) Strategy populations



(b) Information chain usage

Figure 4: in experiment 3 (N=150)