Group Decision and Negotiation **11**: 433–447, 2002
© 2002 *Kluwer Academic Publishers. Printed in the Netherlands*

Cooperative Agent Systems: Artificial Agents Play the Ultimatum Game

FANG ZHONG AND D.J. WU
LeBow College of Business, Drexel University, Philadelphia, PA, USA

STEVEN O. KIMBROUGH
The Wharton School, University of Pennsylvania, Philadelphia, PA, USA

Abstract

We explore computational approaches for artificial agents to play the ultimatum game. We compare our agents' behavior with that predicted by classical game theory, as well as behavior found in experimental (or behavioral) economics investigations. In particular, we study the following questions: How do artificial agents perform in playing the ultimatum game against fixed rules, dynamic rules, and rotating rules? How do coevolving artificial agents perform? Will learning software agents do better? What is the value of intelligence? What will happen when smart learning agents play against dumb (no-learning) agents? What will be the impact of agent memory size on performance? This exploratory study provides experimental results pertaining to these questions.

Key words: artificial agents, cooperative agent systems, reinforcement learning, ultimatum game

1. Introduction

The ultimatum game has been the subject of wide fascination and study with regard to bargaining and negotiation. (Subsequently, we will not distinguish these very similar concepts.) For example, any trading rule can be characterized as a negotiation over how to split the surplus resulting from a trade (see e.g., Croson 1996). There has been a recent growing interest in the ultimatum game by game theorists, economists, psychologists, and computer scientists, among others (Binmore, Gale, and Samuelson 1995; Bolton and Zwick 1995; Burnell, Evans, and Yao 1999; Güth 1995; Huck 1999; Kagel, Kim, and Moser 1996; Knez and Camerer 1995; Roth *et al.* 1991; Ruffle 1998; Straub and Murnighan 1995). The importance of understanding the ultimatum game extends beyond purely scientific considerations. It is important as well from an e-commerce applications perspective: if we are to field intelligent, (semi-) autonomous agents in conducting bargaining and negotiation in virtual communities (such as in Internet markets), much remains to be learned about the principles of design and management of such artificial agents (Kimbrough and Wu 2001). We have conducted a series of investigations into the dynamics of agents, having varying degrees of intelligence, in strategic situations (or games). These investigations grew out of, and advance, our previous studies of the dynamics of game-playing agents. Strategic situations are ubiquitous. How do and how can learning, deliberating, adaptive agents perform in strategic situations? How do their performances compare with those of strategy-centric agents? How well do different identity-centric agents, using various learning and adaptation regimes, fare against one another in strategic situations? How well do various adaptive mechanisms perform when playing with humans? Are there simple adaptive mechanisms whose play behavior maps well to that of human subjects?

Despite its simplicity, the ultimatum game presents a challenge in understanding negotiation behavior by humans. The relevant literature can be roughly divided into two approaches for addressing the ultimatum game: the classical game theory approach and the experimental economics (or behavioral) approach. The simplest version of the ultimatum game is as follows. Two players, A and B, must decide how to split N dollars. At the beginning of each play of the game, Player A decides how much $x \in [O, N]$ to offer player B. Player B can reject the offer, resulting in both of the players getting nothing; or player B can accept the offer, in which case he gets the x dollars and player A keeps the remaining N-x dollars. For simplicity, we assume x to be integer only. Figure 1 shows the extensive form of the game.

Classical game theory asserts that any rational Player A should offer a tiny amount, say for example, one dollar, to Player B, and player B will always accept this minimal offer. This outcome is indeed sub-game perfect. (It passes the Nash Equilibrium test for every sub game, i.e., sub-trees in the extensive form).

Further, there exists an infinite number of Sub-Game Perfect Equilibria in this game (Skyrms 1996), they are along the line of $x + y = N$ (in the continuous case; an analogous condition holds in the discrete case, which we report here). Classical game theory is silent as to which Nash Equilibrium will be finally selected. Given this weakness in terms of prediction, it is not surprising that experimental economists found that classical game theory predictions (e.g., the one dollar equilibrium) were not supported by empirical evidence when human

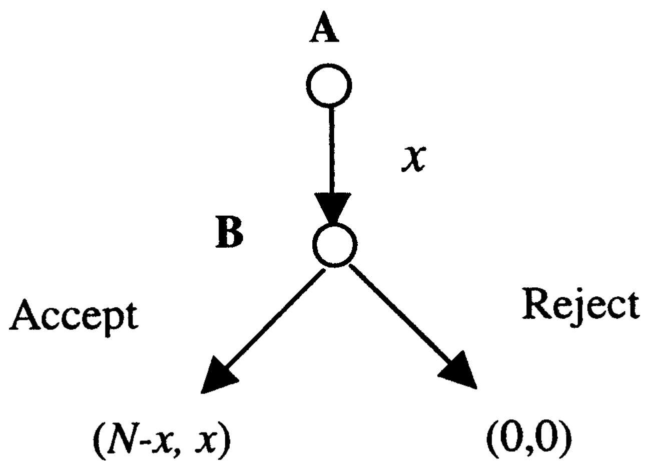


Figure 1. The ultimatum game in extensive form.

subjects were playing the ultimatum game. On the contrary, humans when playing the game in various lab settings, tend to reject offers that do not meet their threshold amounts of share (e.g., Croson 1996; Kagel, Kim, and Moser 1996; Roth *et al.* 1991). These findings, independently reported by a number of researchers (some even consider cultural differences), are consistent and credible. Various explanations and theories have been offered for the deviations from what classical game theory would predict (e.g., Bolton and Zwich 1995; Burnell, Evans, and Yao 1999). For present purposes, these need not detain us.

We conducted an exploratory, but systematic, investigation of ultimatum game play by adaptive agents, agents that form and adjust their strategies in response to experience. The general framework for such agents is *reinforcement learning*, appearing in the machine learning literature (Sutton and Barto 1998). Here we explore computational approaches for artificial agents playing the ultimatum game, and compare our results with classical game theoretical as well as the experimental (or behavioral) economics approaches. In particular, we study the following questions: How do artificial agents perform in playing the ultimatum game against fixed rules, dynamic rules, and rotating rules? How do coevolving artificial agents behave? Will learning software agents do better? What does intelligence do to benefit its holders? What will happen when smart learning agents play against dumb (no-learning) agents? What will be the impact of agent memory size on performance? We provide some initial computer simulation results to these questions.

The rest of the paper is organized as follows. Section 2 outlines our key research methodologies and implementation details. Section 3 reports our experimental findings. Section 4 reports further experiments in order to study the value of intelligence and the impact of memory size on agent performance. Section 5 summarizes and discusses future research.

2. Methodology and Implementations

In our framework, artificial agents are modeled as finite automata (Hopcroft and Ullman 1979; Wolfram 1994). This framework has been adopted by a number of previous investigators. Among them, Binmore and Samuelson (1992), Sandholm and Crites (1995) and many others used the framework to study iterated prisoner's dilemma (IPD). Kimbrough, Wu, and Zhong (2001) used it to study the MIT "Beer Game", where genetic learning artificial agents played the game and managed a linear supply chain. Wu and Sun (2002) investigated the electronic market off-equilibrium behavior of artificial agents in a price and capacity bidding game using genetic algorithms (Holland 1992). These are merely samples to illustrate the validity of this framework in the literature. See Kimbrough and Wu (2001) for a survey.

In this study, however, we depart from previous computational research by integrating several previous standard approaches. First, we study a different game, namely the ultimatum game, rather than other games such as the IPD, Beer, Trust, or Investment games. Second, in studying this game we use evolutionary computation to model agents. This contrasts with the standard approaches. Third, our agents are using reinforcement learning (Sutton and Barto 1998) as a key learning mechanism in game playing while previous studies of the ultimatum game employ no machine learning. Finally, our agents are identity-centric rather than strategy-centric as used in previous studies (e.g., Kimbrough, Wu, and Zhong 2002; Wu and Sun 2002). That is, our agents may meaningfully be said to have individual identities and behavior. They are not just naked strategies that play and succeed or fail. Individuals, not just populations as a whole, learn and adapt over time and with experience. Fielding these kinds of agents, we believe, is needed for e-commerce applications.

We now describe our model and prototype implementations in more detail, using the framework of reinforcement learning (RL). RL is an AI approach to learning whereby an agent tries to maximize the total amount of reward it receives when interacting with an ever changing environment. RL is characterized by trial-and-error (no teacher, in contrast to supervised learning that needs a teacher) and a delayed reward. Key elements of RL includes a policy or rule, a (short-term) reward function, a (long-term) value function and a model of the environment (Sutton and Barto 1998). We discuss how the rules or state-action pairs are embedded in our artificial agents, how the rewards were set up, the long-term goal of the game (returns), the specific reinforcement learning algorithm we designed, and finally our code design using Java.

Rules (State-Action Pairs): In repeated games, each play of a game will be treated as an episode. So within any episode, player A only needs to make a decision at the beginning, and is in only one state during the episode ($s = 0$). The rules for player A only include the action part, which is how much to offer to player B. The value function is simple:

$$Q_A(0, a) \text{ where } a \in [0, N]$$

For player B, the action part is either to accept or to reject, and the state part is the amount offered by player A. So, we have: $a \in$ {Accept, Reject} and $s \in [0,N]$. The value function of Agent B is defined below:

$$Q_B(s, a) \text{ where } s \in [O, N], a \in \{\text{Accept, Reject}\}$$

Rewards: Both of the players get their rewards at the end of each episode. The reward for player A by taking action ***a*** will be:

$$R_A(\mathrm{O}, \mathrm{a}) = \begin{cases} N - a & \textit{when player B accepts} \\ 0 & \textit{when player B rejects} \end{cases}$$

and the reward for player B will be

$$R_B(\mathrm{s}, \mathrm{a}) = \begin{cases} 0 & \textit{when player B rejects} \\ s & \textit{when player B accepts} \end{cases}$$

The value for each state-action pair is simply the average of its rewards during the previous episodes. (Thus the players assume their opponents are playing a stationary strategy.) The value will be updated each time a state-action pair is visited.

Returns: The long-term return is the total payoff each player gets by playing the game repeatedly.

Reinforcement Learning Algorithm: There are two steps in each episode and one decision point for each agent. The decision-making at each point is according to the values of the suitable Q(s, a) and an ε-greedy policy. With probability 1–ε the agent chooses the available action that has the highest Q(s, a) value, and with probability ε the agent chooses randomly and uniformly among the other available actions. The pseudo code for the algorithm is:

```
Initialize $Q_A(0, a) = 0$ and $Q_B(s, a') = 0$
Repeat (for each episode)
Player A: chooses $\boldsymbol{a}$ using ε-greedy policy
Player A: takes action
Player B: observes its state ($\boldsymbol{s = a}$)
Player B: chooses $\boldsymbol{a'}$ from $\boldsymbol{s}$ using ε-greedy policy
Players A and B: get their rewards $\boldsymbol{r}$ and $\boldsymbol{r'}$ and   update their value functions.
$Q_A(0, a) \leftarrow Q_A(0, a)*(n-1)/n + r/n$
$Q_B(s, a') \leftarrow Q_B(s, a')*(n-1)/n + r'/n$
(where $n$ is the total number that the state-action pair has visited)
Until $k$ episodes have been played.
```

Java Coding: In the first experiment, the endowment of player A is set to 63, so its rules as encoded in a binary string have 6 positions, and consequently the rules for player B have 6 + 1 = 7 positions. Model parameters, such as the length of rules for both agents, endowment and the number of episodes to play etc. can be read in from input files. The results of each episode are written into a plain text file.

3. Experimental Results

The artificial agents play a series of ultimatum games, first the repeated one-shot game, second against fixed rules, third against dynamic rules, fourth against rotating rules, and finally against another agent also coevolving under reinforcement learning.

3.1. Repeated one-shot game

In the initial experiment, there is no explicit memory for either agent, i.e. the agents make their decisions regardless of the opponent's last move. Under the current algorithm design, the agents tend to find the game-theoretic result of the ultimatum game, in which player A only offers a tiny amount. The simulation has been multiply rerun using the above configuration and the conclusion was found to be statistically valid. Each time it converges to a different small number less than 10. As for player B, the dominant strategy is to accept. Table 1 is a partial list of the experimental results.

The results are robust to increases in the endowments, such as 127, 255 or 1023, with a larger number of episodes.

These results are not unexpected. Since we set the reward of a reject action by player B to zero, and a number greater than zero when player B chooses to play accept, and since the agents have no memory of previous play, accepting will be the better action under any circumstance. Given B almost always chooses accept, it is better for player A to offer only a small amount. Changing the rewards assigned to player B in a way to favor the reject action when the offer from player A is relatively low will change the results of the experiments.

3.2. Learning agent against fixed rules

Here, the strategies of player B are fixed and punishment is introduced, so that when the last offer from player A is no greater than the current offer, player B will accept it, otherwise reject it. Furthermore, a certain degree of tolerance can be adopted by player B, so that when the proportion of the current offer from player A to the endowment is greater than a certain number p, e.g. 0.8, then B will accept it, otherwise reject it.

```
IF (currentOffer >= $p$* Endowment)
   Accept currentOffer
ELSE
   Reject currentOffer,
```

where p is a model parameter that is defined in the model parameters configuration file.

For player A, if player B accepts his last offer, he will propose an offer no greater than his last offer, i.e. suppose player A proposed d_{t-1} in the $t-1$ time period, then at time t, he will make a selection $d_t \in [0, d_{t-1}]$.

Cooperation emerged. In our experiment where $p = .40$, player A converges on a pro-

Table 1. A partial list of the results

Episode no.	A's offer	A's payoff	B's action	B's payoff
1	19	0	0	0
2	6	0	0	0
3	1	0	0	0
4	37	26	1	37
5	37	52	1	74
486	3	22946	1	5215
487	44	22965		5259
488	3	23025	1	5262
500	3	23685	1	5295

posal of p*Endowment = 40 when the endowment is 100. In other words, player A is capable of learning the policy of player B from iterated games.

3.3. Learning agent against dynamic rules

The lower limit (*p*) of player B is changing as the game is being played. The values and the time of the changes are fixed. Table 2 shows the values of *p* and the occurrence of the adjustment. The first row of Table 2 indicates the number of episodes when a change occurs.

Agent A can track the change very well given enough time periods.

3.4. Learning agent against rotating rules

The lower limit (*p*) of player B is changing with a rotating pattern, say

$$p_{t-1} = .40, p_t = .50, p_{t+1} = .60.$$

We want to see if player A can track such a pattern.

With our current reinforcement learning algorithm, player A converges to a proposal of 60 which is the highest value of *p* * 100. Memory of at least one previous move should be introduced to the algorithm of player A in order for A to track the rotated rules of player B.

3.5. Learning simultaneously

There are no fixed rules for either player A or player B. They will learn to find good solutions for themselves.

In this experiment, both players have memory of their last action. For player A, the state in period *t*, is player B's response in period *t–1*:

$$S^a_t = b_{t-1}$$

The action of player A in time t is still a proposal of the ultimatum, $d_t \in [0, N]$.

For player B, the state in time *t* period has two parameters – the last offer from player A, and his own response in the time *t–1* period:

$$S^b_t = f(d_{t-1}, b_{t-1})$$

Table 2. The values of p in different episodes

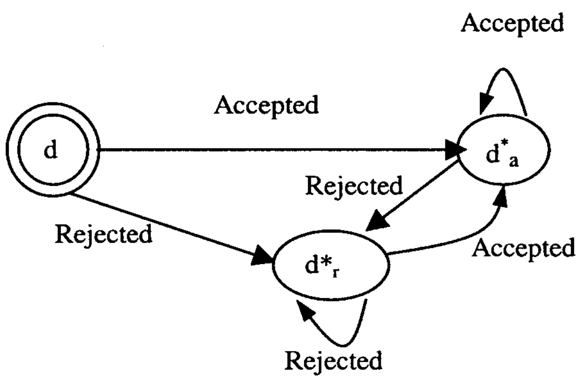
	1	2000	5000	7000	10000
p	0.40	0.35	0.45	0.60	0.40

According to the state signals, player B chooses his lower limit by which he decides whether or not to accept the proposal from A. The decision-making process for player B can be described as follows:

```
IF $b_{t-1}$ is “accept”
THEN
    $p_t = d_{t-1}/N$
ELSE
    $p_t \in [0, N]/N$.
```

The above decision-making process of player A and B is modeled in Figure 2 using finite automata.

Agent A:



Agent B:

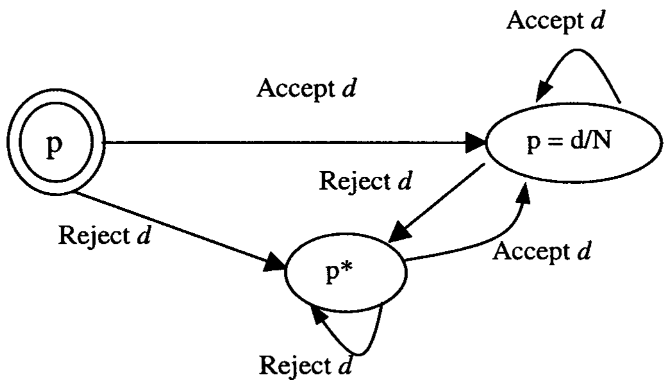


Figure 2. Decision-making processes for indefinite ultimatum game.

Figure 2 should be interpreted as follows. As shown in the top part of the diagram, Agent A starts with a random choice of a possible proposal $d \in [0, N]$. If the proposal is accepted, agent A will choose the best amount to propose according to the value of $Q(s_a, d)$ where $s_a = 1$ (means accept), and continue doing so as long as agent B accepts the proposal. Otherwise, agent A will choose the best amount to propose according to the value of $Q(s_a, d)$ where $s_a = 0$ (means reject), and continue doing so until B accepts.

In the meantime, as depicted in the lower part of the diagram, Agent B starts with a random choice of its lower limit p – acceptable offer. If agent A cooperates, i.e. its offer exceeds this lower limit p, agent B will accept A's offer d and set his lower limit the same as the ratio of agent A's last offer to the endowment and continue doing so as long as agent A meets the threshold. Otherwise, if agent A decreases its offer (we denote this state as s_b), then agent B will set $p \in [0, 1]$ to a new value that maximizes $Q(s_b, p)$, i.e. the optimal p value under the condition that agent A didn't meet B's previous p value. This process will continue until agent A gets back to propose an acceptable offer.

Now we describe the experimental results. Cooperation emerges through co-evolution within 2000 episodes. Player A converges to an offer of 55 or 56, and correspondingly, player B converges to setting his lower limit at 55 or 56. Some open questions arise from this experiment: Why does it converge at 55 and 56 instead of 50? Does the result depend on the initial random choices? We leave this for future theoretical exploration.

4. Further Experiments

We now describe further experiments to investigate the value of intelligence and the impact of memory size on agent behavior. The questions we are interested in are the following. Will learning software agents do better? What good does intelligence do for the agents having it? What will happen when smart learning agents play against dumb (no-learning) agents?

4.1. Value of intelligence

We conducted two experiments to investigate the value of intelligence. In the first experiment (4a), we have one smart (learning) agent and many dumb (no-learning, fixed rule) agents. In the second experiment (4b), we have a population of two types of smart agents and many dumb agents. The goal of these investigations is to characterize the value of intelligence, i.e., whether smart agents would be able to do better through learning.

One Smart Agent vs. Multiple Dumb Agents (4a)

In 4(a), we have three types of dumb agents using different fixed rules. Without loss of generality, we assume the following three rules are being used:

db1: demand/accept 70 or higher;
db2: demand/accept 50 or higher;
db3: demand/accept 30 or higher.

There is a certain possibility (e.g. 25%) that a smart agent can be chosen to play the game. We shall keep a running total of the smart agent's points. The smart agent can learn via reinforcement learning. We would like to see if this smart or learning agent would be able to learn a generally good or reasonable policy. We note here that in designing this learning agent, we have to track the changing population of dumb agents. In each generation, the experiment proceeds as follows:

1. Draw one smart agent with 25 percent possibility; otherwise draw one dumb agent randomly in proportion to their frequency.
2. Draw another dumb agent randomly in proportion to their frequency.
3. Decide the role of each agent, i.e. proposer vs. responder.
4. Two agents play the game against each other.
5. Record the results.
6. Go to the first step until a certain number of games, e.g. 1000 episodes, has been played.
7. Update the frequency of the dumb agent based on observed results of the current or previous time periods.

When a smart agent is chosen to play the game, he will use reinforcement learning to decide how much to propose or the minimum amount he would accept.

The results of this experiment show that the fair agents who propose 50 and accept any amount greater or equal to 50 take over the dumb agent population generation by generation. Figure 3 illustrates such a trend. On the other hand, the smart agents also learn to be fair. Figure 4 illustrates the evolution of the rules one smart agent chose in the experiment, as well as its average point score in each generation, as an indicator of its performance. Although this evolution procedure is different for different smart agents, the graph depicts the aggregate behavior and performance of the smart agent population.

Multiple Smart Agents and Dumb Agents (4b)

In 4(b), we introduce a population of smart agents playing against many dumb agents, and playing against each other as well. Table 3 explains our experimental design. Again, we assume the dumb agents fall into the above three types.

With the same population sizes of smart and dumb agents, the frequency of fair agents increases very slowly. This process is not very smooth for the first 30 generations. We observed decreasing points in this period. Figure 5 shows the evolving process of the frequency of dumb agents within 100 generations.

Again, the behavior and performance of one smart agent is illustrated in Figure 6 to portray the evolution procedure of the overall smart agent population. Figure 7 depicts the distribution of the average points of smart agents.

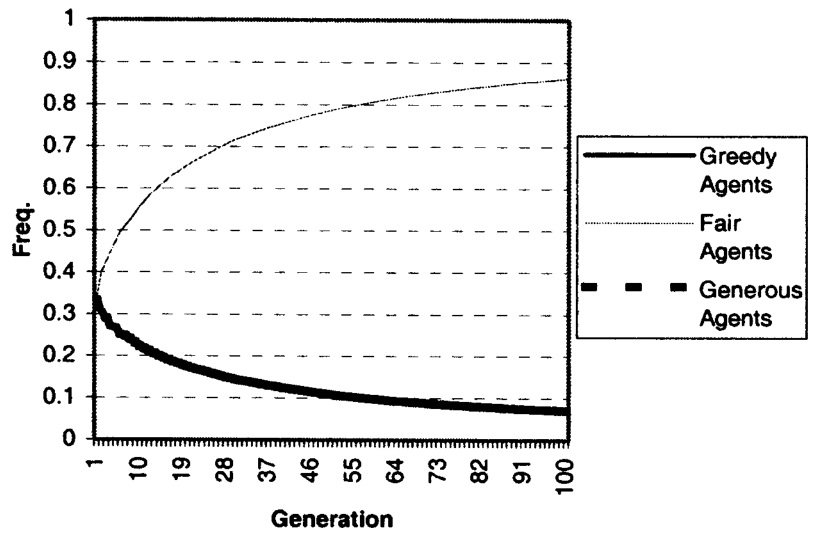


Figure 3. Dynamics of the frequencies of dumb agents. Parameters used for this experiment are: Number of generations = 100, number of plays in each generation = 2000, population size of smart agents = 20, population size of dumb agents = 90.

Table 4 summarizes the average points earned by different types of agent in 4a and 4b. Again, the difference between 4a and 4b is that in the latter, the smart agent could play against another smart agent while in the former case, the smart agent only met with dumb agents (although there are three types of them). It is clear that in either case, smart agents

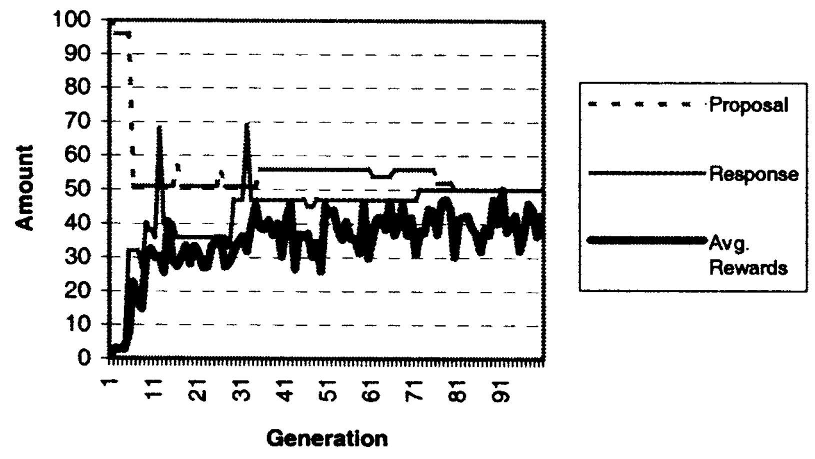


Figure 4. Evolution procedure of the rules used by one smart agent in the smart agent population.

Table 3. Experimental design of 4b

		Responder	
		Smart (y)	Dumb (1–y)
Proposer	Smart (x)		
	Dumb (1–x)		

and fair agents are doing well, while greediness or generosity do not pay. On the other hand, if smart agent plays with another smart agent, then the average points of the smart agent as well as the fair agent decrease a bit – making unusual profit becomes more difficult, other types of agents (the greedy and the generous ones) pick up a bit. In a sense, the distribution of wealth in this community or society becomes a bit "fairer" due to the policy of allowing smart agents to play with smart agents.

4.2. Impact of memory size

Table 5 summarizes our findings on the impact of memory size on agent performance. Where there is only one learning agent (5a), clearly longer memory size helps. Where there are multiple learning agents (5b), and when both have longer memory size, the payoff decreases a bit, while the dumb guys pick up. In any case, it turns out the identity or the type of agent plays an important role. The greedy agents with a longer memory benefit more, while the fair agents remain almost the same (since they try to be fair anyway), and the generous agents give a bit more out (due to their generosity). These findings turn out to be robust across a number of runs.

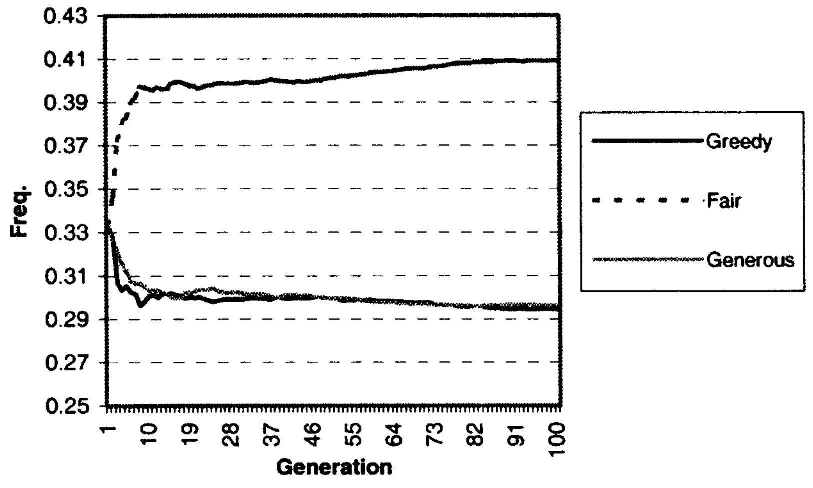


Figure 5. Dynamics of the frequencies of dumb agents when smart agents can play against each other.

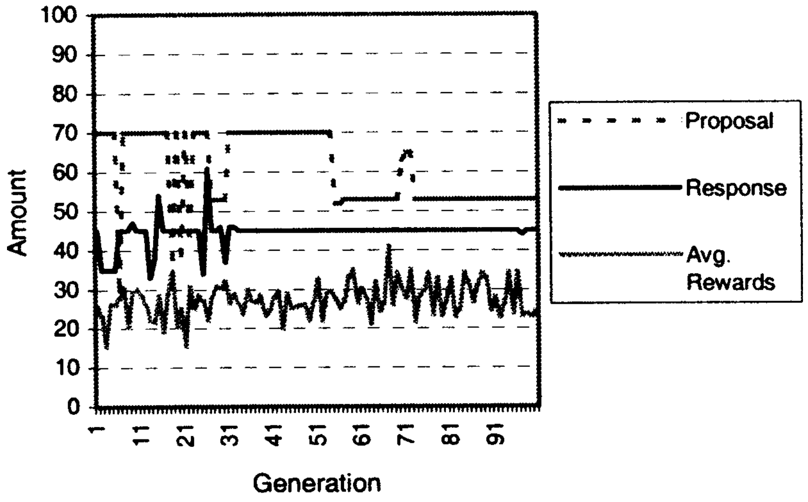


Figure 6. Evolution procedure of the rules used by one smart agent in the smart agent population while smart agents can play against each other.

5. Conclusions and Future Research

Artificial agents using reinforcement learning have been found to be capable of playing the ultimatum game efficiently and effectively. Agent intelligence and memory have been found to have positive impacts on performance. The agent-based approach seems consistent with human behavior. The work here is a first step and sheds light on the design of cooperative agent systems in strategic situations in virtual communities, especially in electronic commerce such as bargaining and negotiations in supply chains.

Following the same paradigm, as outlined in Kimbrough and Wu (2001), we are actively investigating other strategic games, among them are two versions of the trust game: "The classical economic trust game" vs. "The Mad Mex Game" or "The E-Commerce Game", see Wu, Kimbrough, and Zhong (2002) for initial results. These trust games are natural extensions of, yet much more complicated than, the ultimatum game as investigated in this paper. However, the work reported here served as the initial foundation for subsequent

Table 4. Average points of the agents at the end of 100th generation

	Average points Experiment 4a	Experiment 4b
Smart Agents	40.1	30.2
Greedy Agents	19.9	25.0
Fair Agents	43.8	35.1
Generous Agents	20.6	25.1

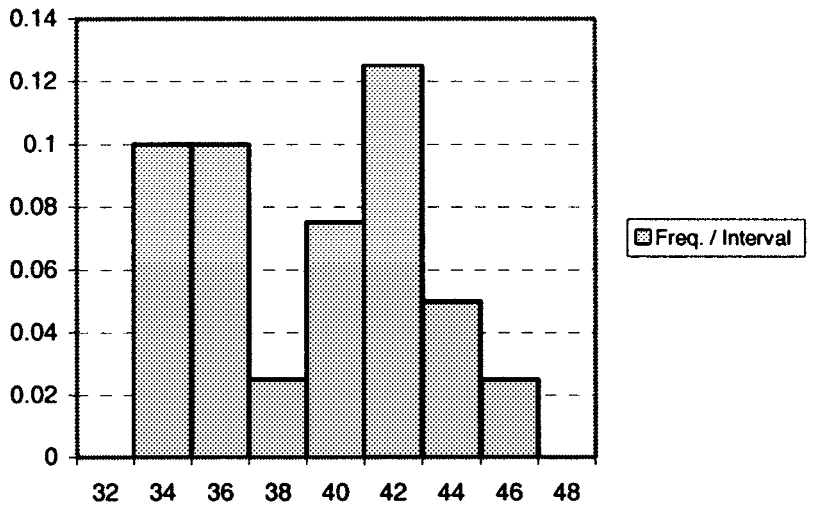


Figure 7. Distribution of the average points of smart agents.

Table 5. Impact of memory size. Parameters used: Number of plays = 2000, dumb agents population size = 90, smart agent population size = 20, number of generations = 100

Agent types	Experiment 5a		Experiment 5b	
	M = 1	M = 2	M = 1	M = 2
Smart w. mem.	33.3	34.1	26.0	23.7
Smart w/o mem.	N/A	N/A	29.9	29.0
Greedy	21.2	22.3	25.7	26.1
Fair	43.2	42.8	34.7	34.1
Gen.	20.7	19.6	25.1	25.1

work. We plan to conduct lab experiments with real agents (humans) playing the ultimatum game and the trust game, including games in which real agents play with our artificial agents. The goals of this investigation are several-fold: to validate our model here, to test the predictive quality of our model, and to explain human decision making behavior. Our ultimate goal is to develop a (semi)-automated negotiation support system in online communities where artificial agents are integrated with human beings to support human decision making in electronic marketplace. We have reported here some first steps in that direction. Much remains to be done.

Acknowledgements

An earlier version of this paper appeared at the Proceedings of the Thirty-Fifth Annual Hawaii International Conference on System Sciences (CD ROM), 2002. We thank the participants for their comments. This material is based upon work supported by, or in part by, NSF grant number SES-9709548.

References

Binmore, K., J. Gale, and L. Samuelson. (1995). "Learning to Be Imperfect: The Ultimatum Game," *Games and Economic Behavior* 8, 56–90.

Binmore, K., and L. Samuelson. (1992). "Evolutionary Stability in Repeated Games Played by Finite Automata," *Journal of Economic Theory* 57, 278–305.

Bolton, G. E., and R. Zwick. (1995). "Anonymity versus Punishment in Ultimatum Bargaining," *Games and Economic Behavior* 10 (1), 95–121.

Burnell, S. J., L. Evans, and S. Yao. (1999). "The Ultimatum Game: Optimal Strategies without Fairness," *Games and Economic Behavior* 26 (2), 221–252.

Croson, R. T. (1996). "Information in Ultimatum Games: An Experimental Study," *Journal of Economic Behavior & Organization* 30 (2), 197–212.

Güth, W. (1995). "On Ultimatum Bargaining Experiments – A Personal Review," *Journal of Economic Behavior & Organization* 27 (3), 329–344.

Holland, J. (1992). *Adaptation in Natural and Artificial Systems.* The MIT Press.

Hopcroft, J. and J. Ullman. (1979). *Introduction to Automata Theory, Languages and Computation*. Reading, MA: Addison-Wesley.

Huck, S. (1999). "Responder Behavior in Ultimatum Offer Games with Incomplete Information," *Journal of Economic Psychology* 20 (2), 183–206.

Kagel, J. H., C. Kim, and D. Moser. (1996). "Fairness in Ultimatum Games with Asymmetric Information and Asymmetric Payoffs," *Games and Economic Behavior* 13 (1), 100–110.

Kimbrough, S. O., and D.J. Wu. (2001). "Designing Artificial Agents for E-Business: An OR/MS Approach," Working Paper, The Wharton School, University of Pennsylvania.

Kimbrough, S. O., D. J. Wu, and F. Zhong. (2002). "Computers Play the Beer Game: Can Artificial Agents Manage Supply Chains," *Decision Support Systems* 33 (3), 323–333.

Knez, M. J., and C. F. Camerer. (1995). "Outside Options and Social Comparison in Three-Player Ultimatum Game Experiments," *Games and Economic Behavior* 10 (1), 65–94.

Roth, A., V. Prasnika, M. Okuno-Fujiwara, and S. Zamir. (1991). "Bargaining and Market Behavior in Jerusalem, Ljubljana, Pittsburgh, and Tokyo: An Experimental Study," *The American Economic Review* 81 (5), 1068–1095.

Ruffle, B. J. (1998). "More Is Better, But Fair Is Fair: Tipping in Dictator and Ultimatum Games," *Games and Economic Behavior* 23 (2), 247–265.

Sandholm, T., and R. Crites. (1995). "Multiagent Reinforcement Learning in Iterated Prisoner's Dilemma," Special Issue on the Prisoner's Dilemma, *Biosystems* 37, 147–166.

Skyrms, B. (1996). *Evolution of the Social Contract.* New York: Cambridge University Press.

Straub, P. G. and J. K. Murnighan. (1995). "An Experimental Investigation of Ultimatum Games: Information, Fairness, Expectations, and Lowest Acceptable Offers," *Journal of Economic Behavior & Organization* 27 (3), 345–364.

Sutton, Richard S., and Andrew G. Barto. (1998). *Reinforcement Learning: An Introduction*, Cambridge, MA: The MIT Press.

Wolfram, S. (1994). *Cellular Automata and Complexity*. Reading, MA: Addison-Wesley Publishing Company.

Wu, D. J., S. O. Kimbrough, and F. Zhong, (2002). "Artificial Agents Play the 'Mad Mex Trust Game': A Computational Approach," *HICSS-35*.

Wu, D. J., and Y. Sun. (2002). "Cooperation in Multi-Agent Bidding," *Decision Support Systems* 33 (3), 335–347.