Reinforcement Learning Rules in A Repeated Game

Ann Maria Bell
*Caelum Research, NASA Ames Research Center, Mail Stop 269-3, Moffett Field, CA 94035-1000 (*abell@mail.arc.nasa.gov*)*

Abstract. This paper examines the performance of simple reinforcement learning algorithms in a stationary environment and in a repeated game where the environment evolves endogenously based on the actions of other agents. Some types of reinforcement learning rules can be extremely sensitive to small changes in the initial conditions, consequently, events early in a simulation can affect the performance of the rule over a relatively long time horizon. However, when multiple adaptive agents interact, algorithms that performed poorly in a stationary environment often converge rapidly to a stable aggregate behaviors despite the slow and erratic behavior of individual learners. Algorithms that are robust in stationary environments can exhibit slow convergence in an evolving environment.

Keywords: reinforcement learning, learning in games, complex systems

1. Introduction

Reinforcement learning (RL) is a powerful machine learning technique that focuses on how agents learn from interacting with their environment. Agents associate rewards with actions on the basis of past experience and then translate those rewards into the probabilities of taking those actions in the future. Recently economists have explored reinforcement learning as a foundation for existing equilibrium concepts in game theory and as an equilibrium selection technique in the presence of multiple equilibria (Fudenberg and Levine, 1998). Reinforcement learning models have also been used to explain experimental data from subjects learning to play repeated games (Roth and Er'ev, 1995), (Er'ev and Roth, 1998), (Er'ev and Rapoport, 1997), (Rapoport, Seale, and Winter, 1998).

One prominent feature of agent-based learning in the context of games and complex adaptive systems is the endogeneity generated by interactions with other learning agents. This paper explores the effects of endogeneity in an adaptive system where agents use simple reinforcement learning rules. The performance of a standard RL algorithm from the computer science literature is contrasted with that of two simpler reinforcement learning algorithms utilized by Roth and Er'ev (1995, 1998). The first part of the analysis examines a single learning agent facing a stationary stochastic environment where all other agents choose their actions based on independent draws from a fixed probability distribution. In a stationary environment details of the problem specification which do not affect the Nash equilibria of the underlying game, such as the initial conditions of the adaptive agents' states and the scaling of rewards, can dramatically affect the performance of the learning rules. The analysis then proceeds to the changes in the dynamic and equilibrium behavior of the system as the proportion of adaptive learning agents increases, creating an endogenously evolving, non-stationary environment. In many cases, the adaptive system as whole rapidly converges to a fixed average or aggregate behavior despite the often slow and erratic convergence of the individual learning rules. But again, the details of the algorithm specification can lead to very different global individual outcomes, especially in the short and medium term. The simulation based results suggest that the specification of the learning algorithm can greatly influence the both the outcomes for individual agents and the dynamics of the entire system.

2. Reinforcement Learning

In reinforcement learning agents formulate policies, or mappings from states to actions, on the basis of the rewards associated with those state-action pairs in the past. In contrast, other approaches to learning focus on forming explicit models of the environment or on searching the action space through evolution and selection (Sutton and Barto, 1998). Best response dynamics implicitly rely on an underlying model of the environment. For example, in fictitious play agents deterministically choose the best response to their beliefs about other agents' actions. Other models of learning in games include mutations or evolutionary search techniques.

The key element of reinforcement learning is specifying a technique for mapping rewards into future probabilities of taking actions. One standard formulation of reinforcement learning (Sutton and Barto, 1998) maps past rewards into future actions through a Gibbs, or Boltzmann, distribution. At time t each learning agent is specified by a vector of weights for the two actions: $w_t = \{w1_t, w2_t\}$, and a vector which records the number of times that each action has been taken: $b_t = \{b1_t, b2_t\}$. In a basic reinforcement learning algorithm for reward maximization agent chooses each action with probabilities:

$$p1_t = \frac{e^{\frac{w1_t}{T_t}}}{e^{\frac{w1_t}{T_t}} + e^{\frac{w2_t}{T_t}}} \qquad \text{and} \qquad p2_t^i = \frac{e^{\frac{w2_t}{T_t}}}{e^{\frac{w1_t}{T_t}} + e^{\frac{w2_t}{T_t}}} \tag{1}$$

where T_t is a "temperature" parameter that declines slowly over time, making it more likely that the action with the higher weight is chosen.

The function determining T_t is:

$$T_{t+1} = Max[\mu \; T_t, \bar{T}] \tag{2}$$

where $\bar{T}$ is a parameter indicating the minimum temperature and μ, $0 < \mu < 1$, is a multiplicative stepsize. The initial value of T_0 is a parameter that indicates the amount of randomness in agents' choices early in the simulation. As T_t approaches 0 the agent becomes more and more likely to choose the action with the higher weight—the agent's choice of action approaches a deterministic best response to the estimated rewards of different actions.

The weights for an action are updated according to the following rules:

$$\begin{aligned} w1_{t+1} &= w1_t + I1_t \; \beta1_t \; (-w1_t + \; r1_t) \\ w2_{t+1} &= w2_t + I2_t \; \beta2_t \; (-w2_t + \; r2_t) \end{aligned} \tag{3}$$

where $I1_t$ and $I2_t$ are indicator variables that equal one when the action is taken and zero when it is not, and $r1_t$ and $r2_t$ are the rewards at time t for taking actions 1 (deterministic payoff) and 2 (random or state dependent payoff) respectively. The parameters $\beta1_t = \frac{1}{b1_t}$ and $\beta2_t = \frac{1}{b2_t}$ for updating the weights correspond to averaging the payoff for that action over all observations. Consequently, the weights change more slowly over time. Note that the rewards may be negative: the probabilities in (1) above are defined for negative and positive weights. We refer to the algorithm determined by (1), (2) and (3) and the β vector as the CS algorithm.

The key free parameters are T_0, μ and $\bar{T}$. The final temperature $\bar{T}$ determines how much experimentation the agents engage in in the long run but does not influence the algorithm before that point. The initial temperature and the stepsize play a crucial role in determining the speed of convergence: a low initial temperature or one that decreases too quickly can lead to slow convergence because the agents are unlikely to take actions that are less preferred and fail to learn the correct value of those actions; a high initial temperature or one that decreases too slowly can lead agents to choose actions without much regard to the average payoffs of those actions in the short run. The initial weights $\{w1_0, w2_0\}$ are also free parameters, but for reasonable initial values of T_0 they have little effect on the behavior of the system: the differences in weights are relatively unimportant when T is high, and the initial weights are replaced with the reward received after the first time an action is chosen.

Roth and Er'ev 1995 and Er'ev and Roth 1998 consider reinforcement learning rules based on the work of psychologists Bush and Mosteller

1955. They also use a vector of weights for each action but the weights are translated into probabilities by:

$$p1_t = \frac{w1_t}{w1_t + w2_t} \qquad p2_t = \frac{w1_t}{w1_t + w2_t}. \tag{4}$$

They consider several different ways to update the the weights, two of which are considered here. In the first case the rewards are added to the weights for each action:

$$\begin{aligned} w1_{t+1} &= w1_t + I1_t \; r1_t \\ w2_{t+1} &= w2_t + I2_t \; r2_t \end{aligned} \tag{5}$$

The weights are the total sum of rewards for each action. We refer to (4) and (5) along with the initial conditions of the parameters as the RE1 learning rule[1]. The free parameters are the two initial weights $\{w1_0, w2_0\}$, which determine the initial probability and the scaling of the stepsize or the extent to which the rewards received change the probability over time. The magnitude of the rewards also affects the rate of change of the probabilities: the algorithm is not independent of the units of measurement. Note that every time an action is taken it is more likely to be taken in the future: high rewards received early in the simulation can have a large affect on the future trajectory of the system. Roth and Er'ev address this by including a "forgetting" parameter in the updating of the weights:

$$\begin{aligned} w1_{t+1} &= (1 - I1_t \; \phi)w1_t + I1_t \; r1_t \\ w2_{t+1} &= (1 - I2_t \; \phi)w2_t + I2_t \; r2_t \end{aligned} \tag{6}$$

where ϕ prevents the weights from growing without bound over time and puts a lower bound on the change in the probability of taking an action for non-zero rewards. We refer to (6) and (4) and the initial parameters $\{w1_0, w2_0\}$ and ϕ as the RE2 rule[2].

3. Description of the Game

Agents repeatedly play a simple congestion game whose pure strategy Nash equilibria require tacit coordination in the absence of communication. There are two actions, one of which returns a deterministic

[1] This is referred to as "the basic reinforcement model" in (Er'ev and Roth, 1998), p. 860. Because the minimum payoff is zero, hence the term they refer to as x_{min} disappears.

[2] This rule is stated on p. 863 in (Er'ev and Roth, 1998) and on p. 175 in (Roth and Er'ev, 1995).

payoff and the other whose payoff depends on the number of players who chose that action.

The game resembles the market entry game where agents choose between entering a market (the payoff depends on the number of entrants) and not entering a market (a fixed payoff). The market entry game has been studied theoretically by Selten and Guth 1982 and Gary-Bobo 1990 and experimentally by Sundali, Rapoport and Seale 1995, Rapoport, Seale and Winter 1998, and Er'ev and Rapoport 1997.

The *El Farol* or Santa Fe bar problem proposed by Arthur 1994 has a similar structure: agents receive a fixed payoff for staying home but the payoff to attending the bar depends on the number of other patrons. The *El Farol* problem has received little attention in the economics literature but has generated a number of papers in computer science. Sethares and Bell (Sethares and Bell, 1998) and Bell, Sethares and Bucklew (Bell, Sethares and Bucklew, 1999) consider the dynamic and equilibrium behavior of simple learning rules in that context. Kephart, Hogg and Huberman 1999 and Fogel, Chellapilla and Angeline 1999 provide more general approaches to analyzing the *El Farol* problem as a complex adaptive system.

Let the deterministic payoff for action 1 be zero. Let M be the total number of agents and $\mathcal{N}$ be a capacity parameter that determines the payoff for action 2. Agents receive a good payoff, g, if the number of agent choosing action 2 falls below the capacity parameter and a bad payoff, b, if the number of agents choosing action 2 exceeds $\mathcal{N}$. The game is then $G = [M, \{S_i\}, u_i(s_i, s_{-i})]$ where S_i consists of two strategies with payoffs determined by $u_i(1, s_{-i}) = 0$ for all s_{-i}, $u_i(2, s_{-i}) = g$ when $\sum_{s_{-i}} \leq \mathcal{N} - 1$, and $u_i(2, s_{-i}) = b$ when $\sum_{s_{-i}} > \mathcal{N} - 1$.

In a deterministic setting where agents utilize only pure strategies a Nash equilibrium occurs when exactly $\mathcal{N}$ agents choose to attend. There are $\binom{M}{\mathcal{N}}$ such equilibria. Each agents' mixed-strategy profile consists of a single parameter p_i which indicates the probability that agent i attends. Let M be the total number of agents, N be the total number of agents who chose action 2, N^{-i} be the total of agents who chose action 2 exclusive of agent i and $\mathcal{N}$ be the maximum capacity parameter.

A mixed-strategy equilibria must satisfy the condition:

$$g \; Pr\left(N^{-i} \leq \mathcal{N} - 1\right) + b \; Pr\left(N^{-i} > \mathcal{N} - 1\right) = 0$$
$$\text{or} \quad Pr\left(N^{-i} \leq \mathcal{N} - 1\right) = \frac{b}{b - g} \tag{7}$$

which states that the expected return to action 1 equals that of action 2. This must hold for all agents simultaneously. Also note that the indifference condition that determines a mixed strategy equilibrium depends on the distribution of the number of agents choosing action

2 which in general depends on the probabilities of taking action 2 for individual agents, not just on the mean of the entire distribution.

For a symmetric mixed strategy equilibria the probability that $\mathcal{N}-1$ or fewer agents choose action 2 is :

$$\sum_{N^{-i}=0}^{N^{-i}=\mathcal{N}-1} \binom{\mathcal{N}-1}{N^{-i}} \left(p^{N^{-i}}(1-p)^{\mathcal{N}-1-N^{-i}}\right). \tag{8}$$

There are no asymmetric mixed strategy equilibria. Consider two agents with differing probabilities of action 2 and, without loss of generality, label them agents 1 and 2 with $p_1 < p_2$. The indifference condition (7) must hold for every agent, which implies that $Pr\left(N^{-1} \leq \mathcal{N}-1\right)$ equal $Pr\left(N^{-2} \leq \mathcal{N}-1\right)$. The density function for N^{-1} can be expressed in terms of the density function exclusive of agents 1 and 2:

$$Pr(N^{-1}=0) = Pr(N^{-1,-2}=0)(1-p_2)$$

$$Pr(N^{-1}=x) = Pr(N^{-1,-2}=x)(1-p_2) + Pr(N^{-1,-2}=x-1)\ p_2.$$

By expanding and combining sums the cumulative distribution that agent 1 faces can be expressed as:

$$Pr(N^{-1} \leq X) = \sum_{x=0}^{x=X-1} Pr(N^{-1,-2}=x) + Pr(N^{-1,-2}=X)(1-p_2).$$

The cumulative distribution function that agent 2 faces differs only by the term $(1-p_2)$ which is replaced by $(1-p_1)$. Consequently, the indifference condition cannot hold simultaneously for two agents with different probabilities of taking action 2.

4. Simulation Results

In the simplest case there is a single learning agent who faces a stationary stochastic environment: all of the other agents choose action 1 with a fixed probability. The behavior of the system is exogenously determined and independent of the agent's actions. The game reduces to a two-armed bandit problem in which the first arm (action 1) returns a fixed payoff and the other arm (action 2) returns a payoff determined by the realization of a multinomial random variable corresponding to the actions of the other agents. This simple scenario provides a baseline measurement of an agent's ability to learn the correct action with the various learning rules.

The situation becomes more complex when two agents are learning at the same time: the environment is still largely exogenous and stationary but the possibility that one agent's actions will influence the future actions of the other agent is introduced. As a larger fraction of the agents simultaneously attempt to learn the value of the two possible actions the environment becomes less stationary, at least initially, and the future behavior of individual agents and the entire system depends on the dynamic interaction of the learning rules. Varying the number of learning agents demonstrates the performance of the learning rules in nonstationary, endogenously evolving environments.

4.1. A single learner in a stationary environment

The initial simulations explore the behavior of the different algorithm specifications when an individual learner faces a stationary environment. The deterministic payoff for action 1 is 1.02, the payoff for action 2 when capacity is not exceeded is approximately 2.02 and 0 when capacity is exceeded. There are 30 agents, the capacity parameter $\mathcal{N}$ is 18. There is one learning agent, the other 29 agents base their actions on independent realizations of a Bernoulli random variable with probability of choosing action 2 of .61. The expected payoff to action 2 is approximately 0.94. Consequently, the best response of the learning agent is to always choose action 1.

The initial parameters for the algorithms determine the probability of action 2 in the first period, this is .60 for all three algorithm specifications. For the CS algorithm the initial weights are $w_0 = \{1.02, 1.32\}$ which leads to a 60% chance of action 2 when combined with the initial temperature $T_0 = .75$. The multiplicative factor used to lower the temperature over time is $\mu = .9975$ until the minimum temperature of $\bar{T} = .025$ is reached. For the RE1 and RE2 algorithms the initials weights are $w_0 = \{.8, 1.2\}$ and the forgetting parameter is $\phi = .001$.

Figure 1 shows the probability of action 2 for the learning agent for 50000 iterations of all three algorithms. The same random numbers were used to determine the action taken and the actions of other agents in all three cases[3]. This suggests the long run behavior of the

[3] The pseudo-random number generator chooses a real number between zero and one. If that number is below the probability of action 2 then the outcome of the Bernoulli random variable is one (action 2), zero (action 1) otherwise. Consequently, when the probabilities are similar the same action is likely to be returned. The pseudo-random number generator acts as an external signal: the differences in the behavior of the algorithms arises from different responses to the same signal. Also, the realizations of the multi-nomial random variable determining the action of the non-adaptive agents is the same across simulations. This provides a more accurate basis for comparison of the performance of the algorithms.

algorithms. The top line in the figure is the RE1 algorithm, the middle line is RE2 and the bottom line is CS. The CS algorithm rapidly trends down to and then fluctuates around the probability associated with the correct estimate of the value of the two actions and the minimum temperature of .025. The RE2 algorithm continues to decline over time. The RE1 algorithm, although apparently stuck at a high probability of action 2, also declines over time: the expected change in the probability of action 2 is negative at every time step but declines rapidly over time as $w1$ and $w2$ increase.

Figure 2 is a close up of Figure 1 which shows the probability of action 2 for 5000 iterations. In this time frame the probability for the RE2 algorithm is slightly higher than the RE1 algorithm; the CS algorithm is still the lower line. The RE algorithms both move in the wrong direction, increasing from the initial probability of .60 to a maximum value of approximately .80. Figure 2 demonstrates the behavior of algorithms in the “intermediate term” that Roth and Er’ev loosely define as the time it takes for the learning curve to become very flat. Comparing the two figures shows the difficulties that arise in identifying the intermediate term. The apparent stability of the RE2 algorithm disappears after the first 5000 iterations even though the stepsize (magnitude of the change in probability) continues to decline over time.

One of the key ingredients in these (and all) adaptive algorithm is the “stepsize” or the amount the probability of choosing action 2 changes at each time step, which is influenced by several parameters in these algorithms. Figures 3, 4 and 5 show the change in the probability of action 2 over time for the three algorithms. Early in the simulations the magnitude of the changes in the probabilities are roughly equal in all three cases[4]. The sum of the absolute values of the change in probability over the first 25 iterations for the CS algorithm is approximately .036, for the RE algorithms, approximately .037. After 500 iterations it is approximately .005 for CS algorithm and approximately .004 for the RE algorithms. (Much later in the simulation the stepsize for the RE2 algorithm becomes larger than that of the RE1 algorithm.) The differences in their behavior are not explained by differing stepsizes, instead, it results from the way the algorithms incorporate the rewards: the CS algorithm decreases the likelihood of action 2 after a bad experience (the zero payoff is averaged into the weight for action 2) but does not increase the likelihood of taking action 1 (the average of the fixed reward doesn’t change after the first few times the action is taken); in

[4] The first observation of approximately .3 for the CS algorithm is not shown on the graph in order to keep the scale the same in the figures for all three algorithms.

contrast, the RE algorithms do not change the likelihood of action 2 after a bad experience (adding a zero leaves the weight for that action unchanged) but decreases the likelihood of action 2 after taking action 1 (the positive payoff is added into the weight for 1). The tendency to (weakly) increase the probability of any action that has been taken can lead the RE algorithms away from the optimal action in the short and medium term.

The performance of the RE1 algorithm in a stationary environment is more problematic than the previous figures suggest: slight changes in the initial conditions can dramatically alter the observed behavior. Figure 6 shows two simulations of the RE1 algorithm with different initial conditions but the same underlying sequence of random numbers as in figure 1. In the first case (upper solid line) the initial weights are $\{.8, 1.0411\}$ with an initial probability of action 2 of approximately 0.565477; the behavior is similar to that observed previously. In the second case (lower dashed line) the initial weights are $\{.8, 1.04105\}$ with an initial probability of ≈ 0.565465; here the RE1 algorithm rapidly tends towards the optimal action,1. The difference in the initial probabilities of action 2 between the two cases is approximately 0.0000118; the difference in the probability at time 10000 is approximately 0.72.

Figure 7 shows the first 50 iterations of figure 1. The divergence of the two simulations occurs in the first iterations when the random number determining the choice of initial actions falls on either side of the of the initial probabilities. These initial choices continue to influence the adaptive agents behavior over the entire course of the simulation. Roth and Er'ev (Roth and Er'ev, 1995) do not consider the initial probability to be one of the free parameters of their model: they set it to 50% in all cases. This assumption can play a crucial role in the behavior of the learning rule.

The free parameter Roth and Er'ev consider is the scaling or magnitude of the initial weights. Increasing the magnitude of the initial weight makes the changes in probability smaller, especially in the first few time steps. Changing the size of the initial weights in this example changes where the divergence in behavior occurs but does not qualitatively change the result: figure 8 shows two simulations with initial weights of $\{8, 9\}$ (upper solid line) and $\{8, 8.75\}$ (dashed lower line). The larger initial weights slowed the movement of both adaptive learners: the difference in the probability at time 10000 is approximately 0.25. A similar situation arises with the RE2 algorithm: figure 9 shows the trajectory of the probability of action 2 starting from initial weights $\{.8, 1.05\}$ (upper solid line) and $\{.8, 1.04\}$ (lower dashed line). Although the RE2 algorithm eventually declines over time, the effects of the initial conditions are apparent after tens of thousands of iterations.

The previous discussion and figures refer to representative simulations. How often do the RE algorithms tend away from the optimal action? The initial probability of .60 in the previous example favors action 2, but the difficulties arise even when 2 is the optimal action. In Figure 10 the dotted lines show the data for one learner using the CS algorithm, the dashed grey lines show the data for the RE1 algorithm and the solid black lines for the RE2 algorithm. The solid line at .60 show the starting point for all the simulations. In all cases the probability of action 2 for the other non-adaptive agents is below .60, so the optimal action for the learning agent is always action 2. Figure 10 shows the data for different fixed probabilities of action 2 for other agents. The lowest dotted line gives the probability of action 2 at iteration 5000 for 100 different runs of the CS algorithm, with the outcomes ordered from lowest to highest. There are twenty-nine other agents with probabilities of .59. The next dotted line gives the same data when the probabilities for the other agents are fixed at .58, and so on though .55. The two highest lines, although not readily distinguished for the CS algorithm, show the data for .45 and .35. The lowest dashed grey line gives the results for the RE1 algorithm and the solid black line gives the results for the RE2 algorithm. When the probability of action 2 is below .60 the process of learning led the agent to favor the action with the lower reward in the first 5000 iterations. For example, despite an initial condition that favored action 2 the RE1 algorithm moved away from the optimal action in about 35% of the simulations. The RE2 algorithm with "forgetting" performs better than the RE1 algorithm, but is still much less likely to take the correct action by time 5000 compared to the CS algorithm. For all three algorithms the lower the probability of action 2 for the other agents, the higher the average reward for action 2 and the easier it is to discern the correct action.

4.2. Complex adaptive systems with multiple learners

When the number of adaptive agents increases, the external environment is not stationary and the actions of the adaptive agents interact over time. Again, the behavior of individual agents and of the system as a whole can differ dramatically depending on the initial conditions of the simulation. In some cases, the most notable feature of the majority of simulations is the rapid convergence of average choice of action 2 to a stable percent, despite the relatively slow convergence of the individual adaptive agents. However, there are also cases where the interaction of the adaptive agents can lead to poor individual and system-wide performance.

Figures 11, 13 and 15 show 25000 iterations of the three algorithms with 15 learning agents and 15 agents who base their actions on independent realizations of a Bernoulli random variable with probability of action 2 of .75. (Figures 12, 14 and 16 are close-ups of figures 11, 13 and 15, respectively.) All of the other initial conditions are the same as the first example. The grey lines are the trajectories of the probability of action 2 for the adaptive agents; the black line is the average probability of action 2 for all agents including those with a fixed probabilities. The behavior of the individual algorithms is qualitatively similar to the case of a single adaptive agent, with the CS and RE2 algorithms converging towards a pure strategy and the RE1 algorithm rapidly approaching a relatively fixed probability. The initial aggregate probability is .675. The mean number of agents choosing action 2 approaches 60% within 10 iterations for all algorithms and the variance declines over time, at least for the CS and RE2 algorithms. The greater variability of the CS algorithm leads to a greater variation in the average choice of action. The endogeneity of the environment also tends to slow down the convergence of the individual learners.

Table 1 summarizes the relationships between the average probability of action 2 and the algorithm specifications when all agents are adaptive. Assuming approximately equal stepsizes or changes in probability across agents and across algorithms the largest change in the average probability is likely to occur when the capacity is exceeded and when CS agents choose action 2 and capacity is not exceeded. For the RE algorithms both actions are reinforced simultaneously when capacity is not exceeded.

Table 1 also suggests that there may be asymmetries between simulations which start with average tendency to choose action 2 above and below .60%. The RE algorithms tend to be somewhat slower to converge (100–150 iterations) to an average probability of .60% when the initial probabilities of action 2 are lower, but otherwise exhibit the same smooth aggregate behavior. The CS algorithm, on the other hand, performs relatively poorly in an endogenous environment.

Figure 17 shows 5000 iterations of the CS algorithm with 15 learning agents and 15 agents who base their actions on independent realizations of a Bernoulli random variable with probability .45. The CS learners rapidly increase their probabilities until all 15 agents choose action 2 every period, and capacity is exceeded 96% of the time. Nonetheless it takes hundreds of iterations for the individual learners to begin to respond to the new environment. The scheme for averaging the rewards into weights is well adapted to a stationary environment but is slow to respond to new external conditions.

Table I. Change in Average Probability of Action 2

learning rule	**capacity not met**	**capacity exceeded**
CS	$N \leq \mathcal{N}$ agents increase probability of action 2	$N > \mathcal{N}$ agents decrease probability of action 2
	$M - N > \mathcal{N}$ agents leave probability of action 2 unchanged, if action 1 has been taken several times	$M - N < \mathcal{N}$ leave probability of action 2 unchanged, if action 1 has been taken several times
net effect	increases	decreases
RE1	$N \leq \mathcal{N}$ agents increase probability of action 2	$N > \mathcal{N}$ agents leave probability of action 2 unchanged
	$M - N > \mathcal{N}$ agents decrease probability of action 2	$M - N < \mathcal{N}$ decrease probability of action 2
net effect	indeterminate	decreases
RE2	$N \leq \mathcal{N}$ agents increase probability of action 2	$N \leq \mathcal{N}$ agents decrease probability of action 2
	$M - N > \mathcal{N}$ agents decrease probability of action 2	$M - N < \mathcal{N}$ decrease probability of action 2
net effect	indeterminate	decreases

When all agents are adaptive the generic behavior of the RE algorithms is relatively rapid convergence of the aggregate probability of action 2 despite the slow movement of individual agents' probabilities. The CS algorithm often rapidly overshoots the capacity of .60%, for initial conditions above and below .60

5. Conclusion

First we consider a single adaptive agent facing a stationary environment. We demonstrate that the simple learning rules proposed by Roth and Er'ev 1995 and Er'ev and Roth 1998 can be extremely sensitive to small changes in the initial conditions and that events early in a simulation can affect the performance of the rule over a relatively long time horizon. In contrast, a reinforcement learning rule based on standard practice in the computer science literature converges rapidly and robustly. The situation is reversed when multiple adaptive agents interact: the RE algorithms often converge rapidly to a stable average aggregate behavior despite the slow and erratic behavior of individual learners, while the CS based learners frequently over-attend in the early and intermediate terms. The symmetric mixed strategy equilibria is unstable: all three learning rules ultimately tend towards pure strategies or stabilize in the medium term at non-equilibrium probabilities of taking action 2. The varied performance of the algorithms in different contexts emphasize the importance of thorough and thoughtful examination of the dynamics and performance of learning algorithms.

References

Arthur, W. B. (1994). Inductive reasoning and bounded rationality: The *El Farol* problem. *American Economic Association Papers and Proceedings*, **84**, 406-411.

Bell, A. M., Sethares, W. A. and Bucklew, J. A. (1999)Coordination failure as a source of congestion in information networks. Working paper. NASA Ames Research Center, http://ic.arc.nasa.gov/ic/people/bell/bell.html.

Bush, R. and Mosteller, F. (1955). *Stochastic Models for Learning.* Wiley, New York.

Erev, I. and Rapoport, A. (1997) Coordination, 'magic,' and reinforcement learning in a market entry game. Working paper. Hong Kong University of Science and Technology Department of Marketing MKTG 97.103.

Er'ev, I. and Roth, A. E. (1998). Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American Economic Review.* **8**(4) 848-881.

Fogel, D. B., Chellapilla, K., and Angeline, P. J. (1999). Inductive reasoning and bounded rationality reconsidered. *IEEE Transactions on Evolutionary Computation.* **3**(2) 142-146.

Fudenberg, D. and Levine, D. K. (1998) *The Theory of Learning in Games.* The MIT Press, Cambridge, MA.

Gary-Bobo, R. (1990) On the existence of equilibrium points in a class of asymmetric market entry games. *Games and Economic Behavior.* **2** 239-246.

Kephart, J. O., Hogg, T. and Huberman, B. A. (1989). Dynamics of computational. *Physical Review A.* **40**(1) 404-421.

Rapoport, A., Seale, D., and Winter, E. (1998) An experimental study of coordination and learning in iterated two-market entry games. Working paper, Hong Kong University of Science and Technology Department of Marketing MKTG 98.116.

Roth, A. E. and Er'ev, I. (1995) Learning in extensive form games: Experimental data and simple dynamic models in the intermediate term. *Games and Economic Behavior.* bf 8 164-212.

Selten, R. and Guth, W. (1982) Equilibrium point selection in a class of market entry games. In *Games, Economic Dynamics, and Time Series Analysis.* Physica-Verlag, Vienna. 101-116.

Sethares, W. A. and Bell, A. M. (1998) An adaptive solution to the *El Farol* problem. In *Proceedings of the Thirty-Sixth Annual Allerton Conference on Communication, Control, and Computing.* Allerton IL.

Sundali, J., Rapoport, A. and Seale, D. (1995) Coordination in market entry games with symmetric players. **Organizational Behavior and Human Decision Processes. 64**(2) 203-218.

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

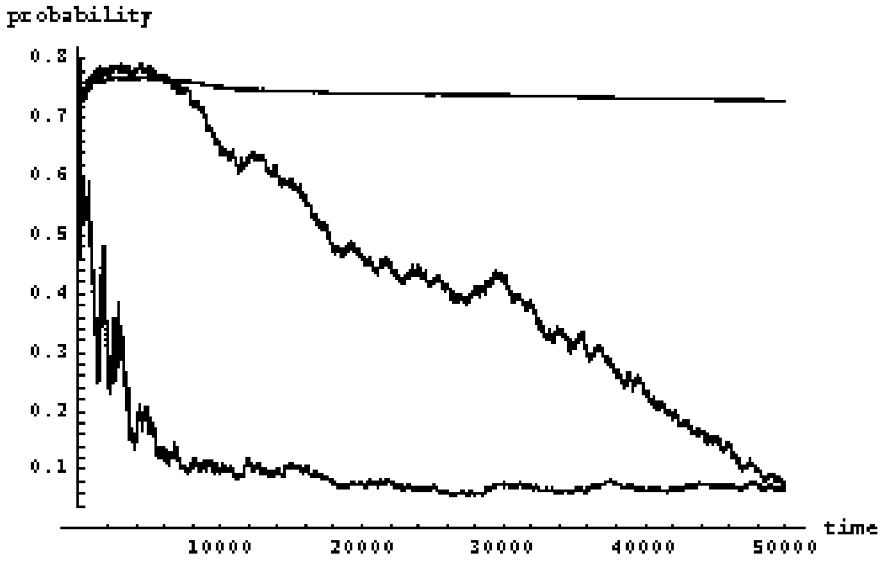

Figure 1. Probability of action 2 for one adaptive agent using the CS (bottom line), RE1 (top line) and RE2 (middle line) algorithms. The fixed probability of action 2 for non-adaptive agents is .61. The optimal action is to choose 1 every period.

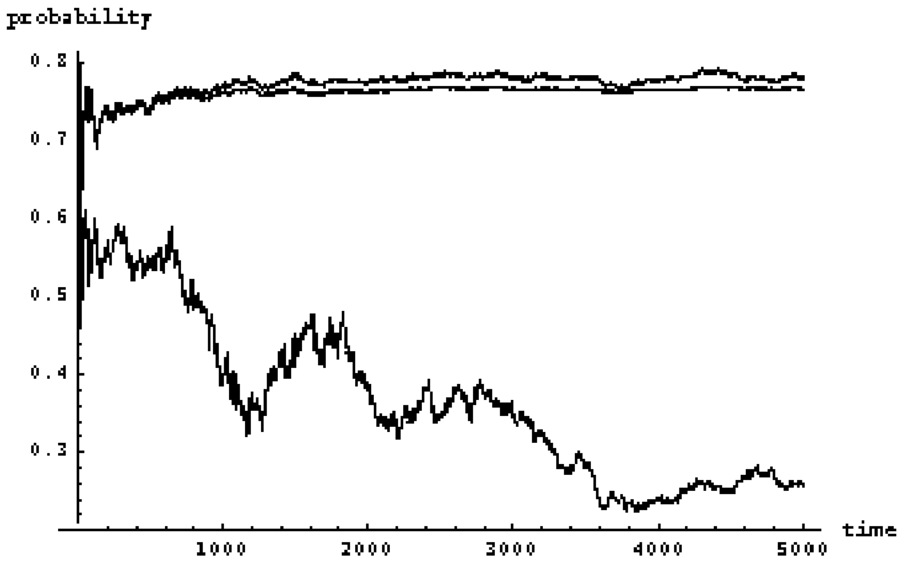

Figure 2. Probability of action 2 for one adaptive agent using the CS, RE1 and RE2 algorithms. First 5000 iterations of figure 1.

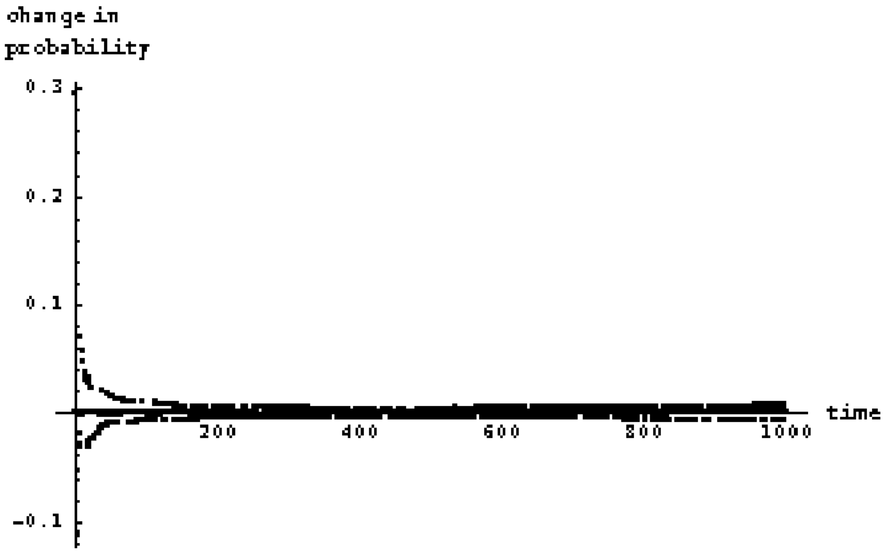

Figure 3. Change in probability of action 2 for the CS algorithm. First 1000 iterations of figure 1.

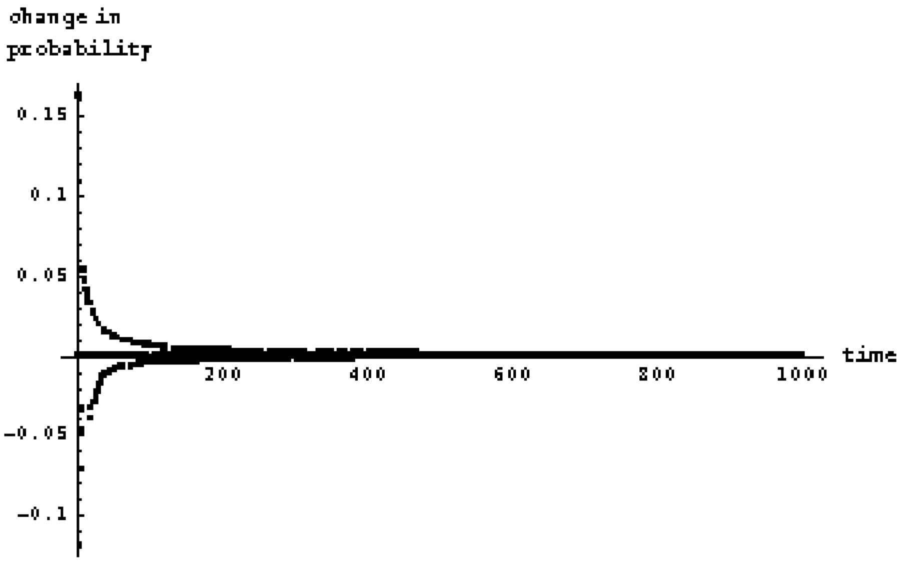

Figure 4. Change in probability of action 2 for the RE1 algorithm. First 1000 iterations of figure 1.

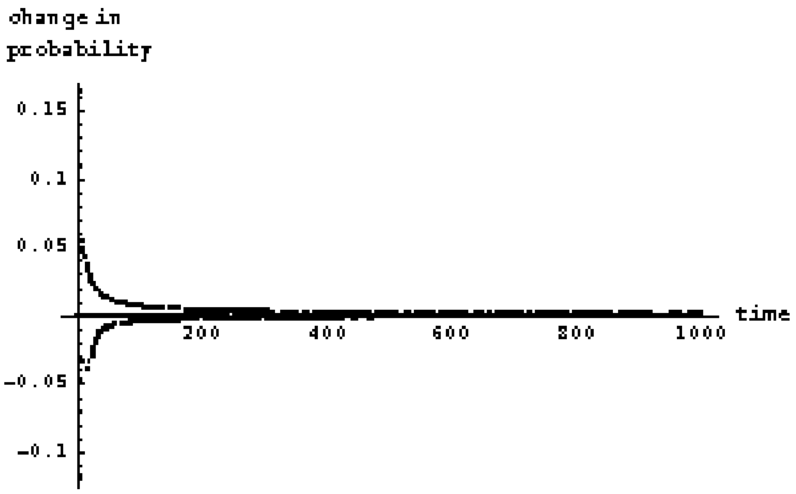

Figure 5. Change in probability of action 2 for the RE2 algorithm. First 1000 iterations of figure 1.

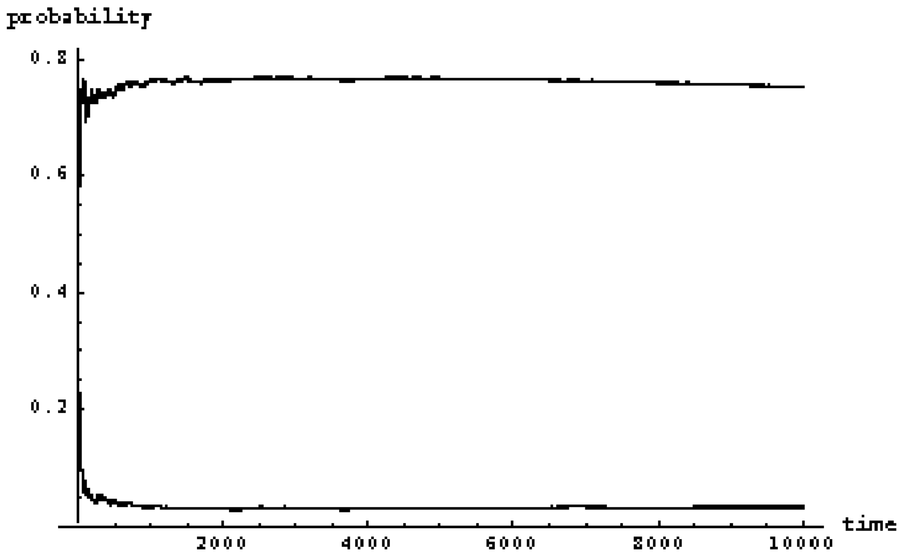

Figure 6. Probability of action 2 for an adaptive agent using the RE1 algorithm with initial weights of $\{.8, 1.0411\}$ (upper solid line) and initial weights of $\{.8, 1.04105\}$ (lower dashed line). The fixed probability of action 2 for non-adaptive agents is .61. The optimal action is to choose 1 every period.

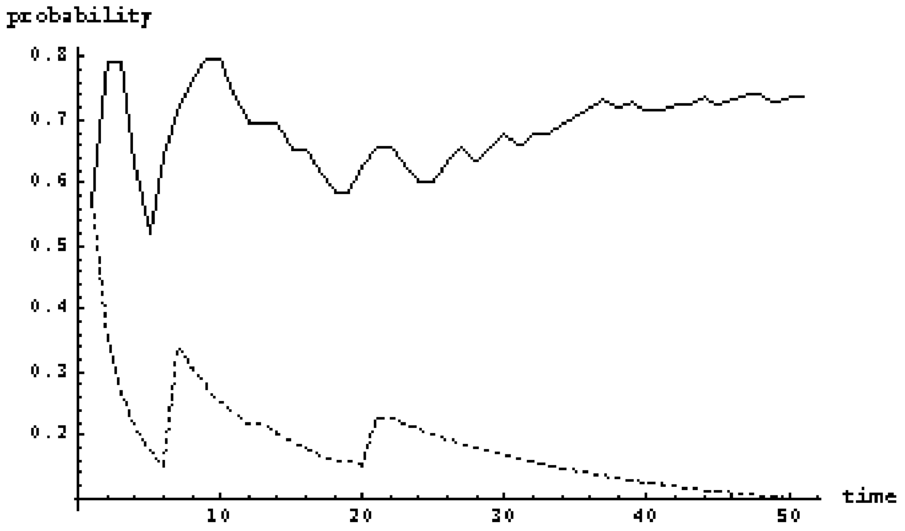

Figure 7. Probability of action 2 for an adaptive agent using the RE1 algorithm with initial weights of $\{.8, 1.0411\}$ (upper solid line) and initial weights of $\{.8, 1.04105\}$ (lower dashed line). First 50 iterations of figure 6.

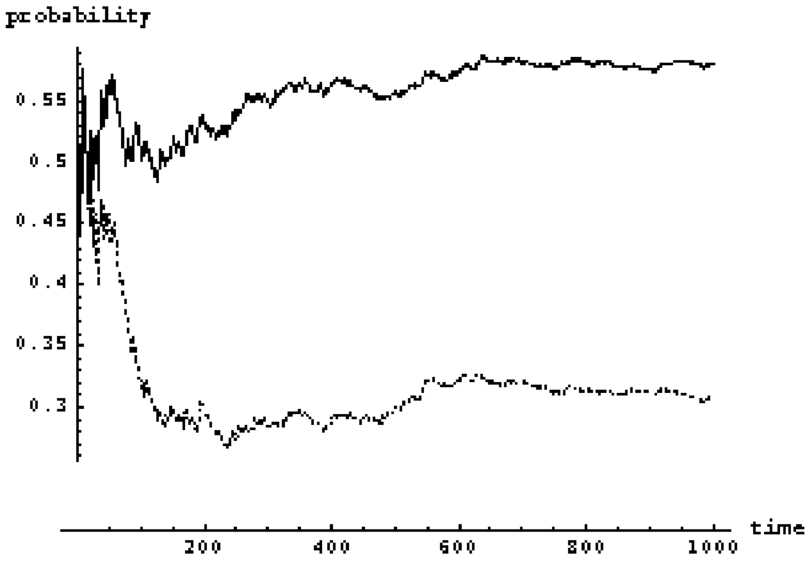

Figure 8. Probability of action 2 for an adaptive agent using the RE1 algorithm with initial weights of $\{8, 9\}$ (upper solid line) and initial weights of $\{8, 8.75\}$ (lower dashed line).

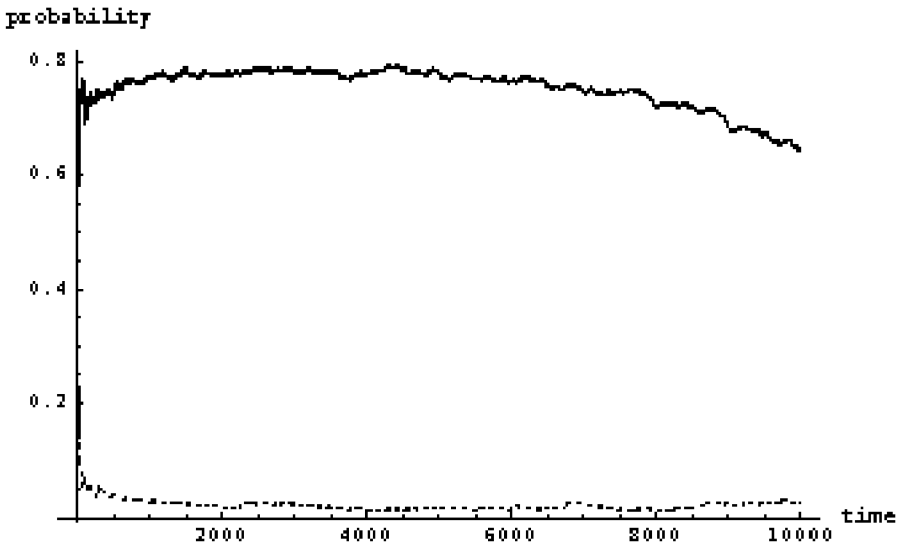

Figure 9. Probability of action 2 for an adaptive agent using the RE2 algorithm with initial weights of $\{.8, 1.05\}$ (upper solid line) and initial weights of $\{.8, 1.04\}$ (lower dashed line). The fixed probability of action 2 for non-adaptive agents is .61. The optimal action is to choose 1 every period.

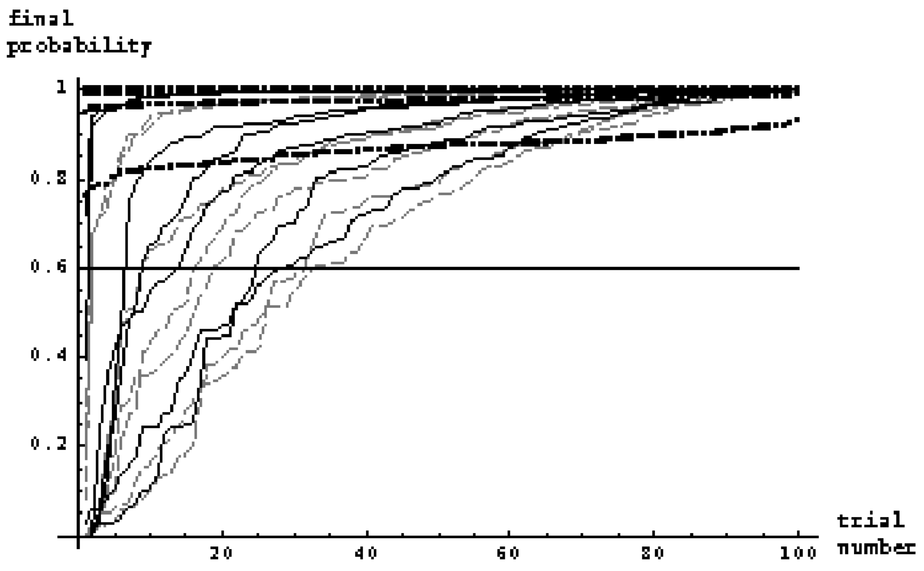

Figure 10. Probability of action 2 for one adaptive agent using the CS (dotted line), RE1(dashed line) and RE2 (solid line) algorithms with 29 non-adaptive agents. Each line represents 100 simulations with differing fixed probabilities of action 2 for the non-adaptive agents, ranging from .59 to .35.

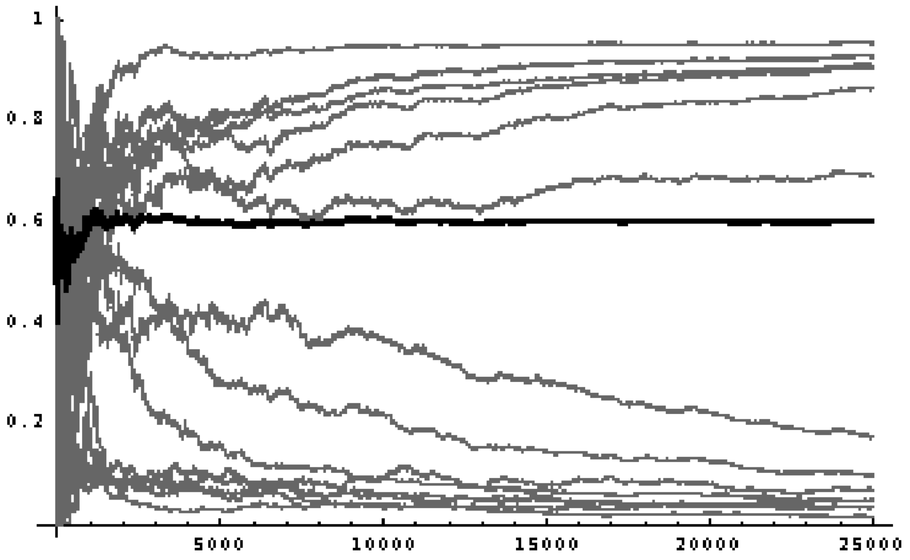

Figure 11. Probability of action 2 for 15 adaptive agents (grey lines) using the CS algorithm with 15 non-adaptive agents with probability of action 2 of .75. Average probability of action 2 for all agents is shown by the solid black line.

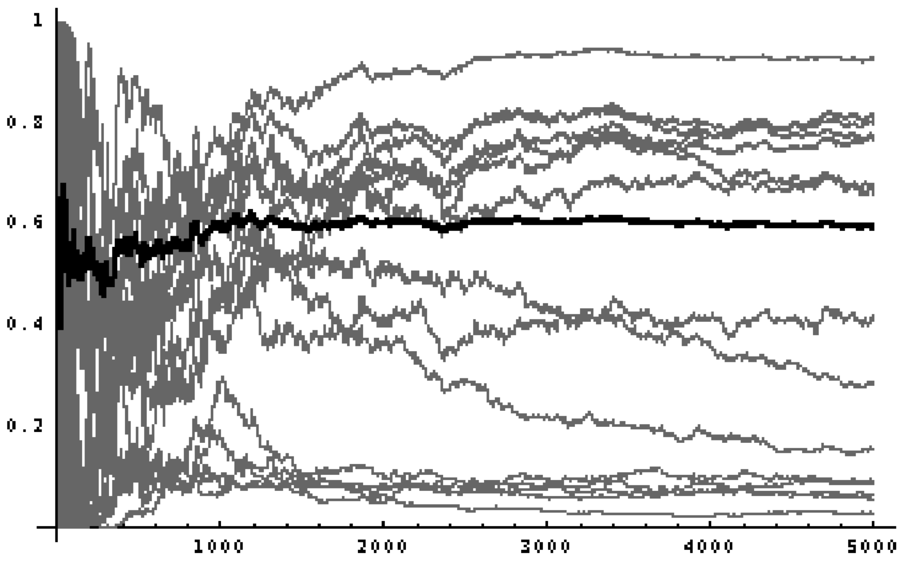

Figure 12. Probability of action 2 for 15 adaptive agents (grey lines) using the CS algorithm with 15 non-adaptive agents with probability of action 2 of .75. First 5000 iterations of figure 11.

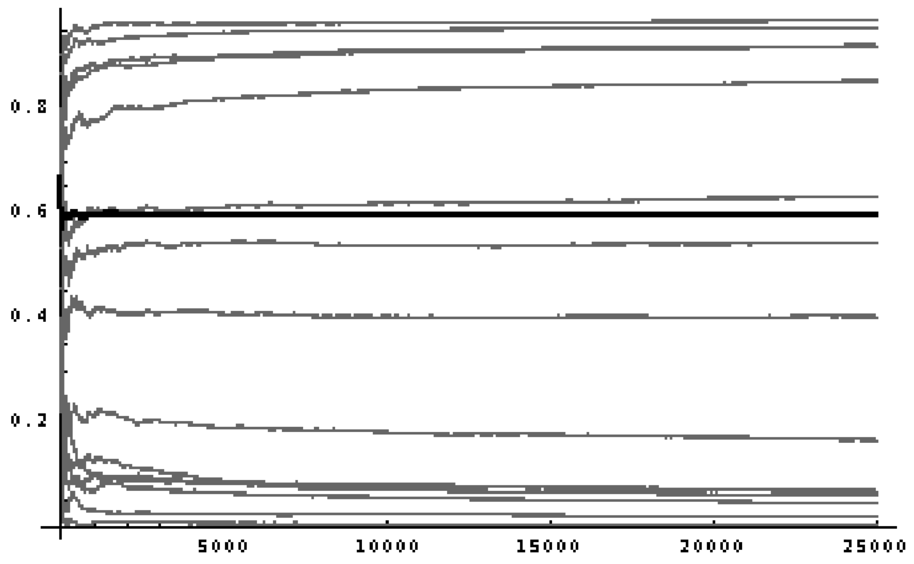

Figure 13. Probability of action 2 for 15 adaptive agents (grey lines) using the RE1 algorithm with 15 non-adaptive agents with probability of action 2 of .75. Average probability of action 2 for all agents is shown by the solid black line.

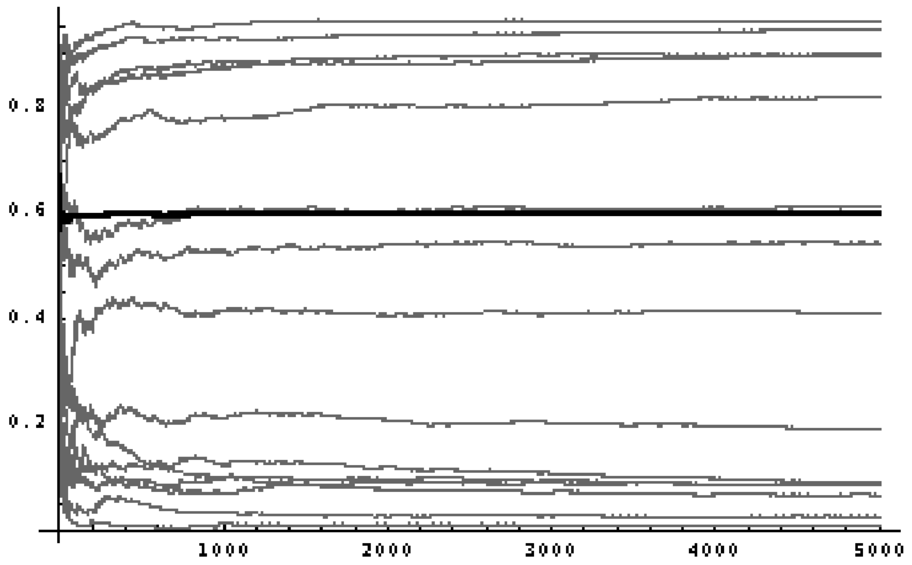

Figure 14. Probability of action 2 for 15 adaptive agents (grey lines) using the RE1 algorithm with 15 non-adaptive agents with probability of action 2 of .75. First 5000 iterations of figure 13.

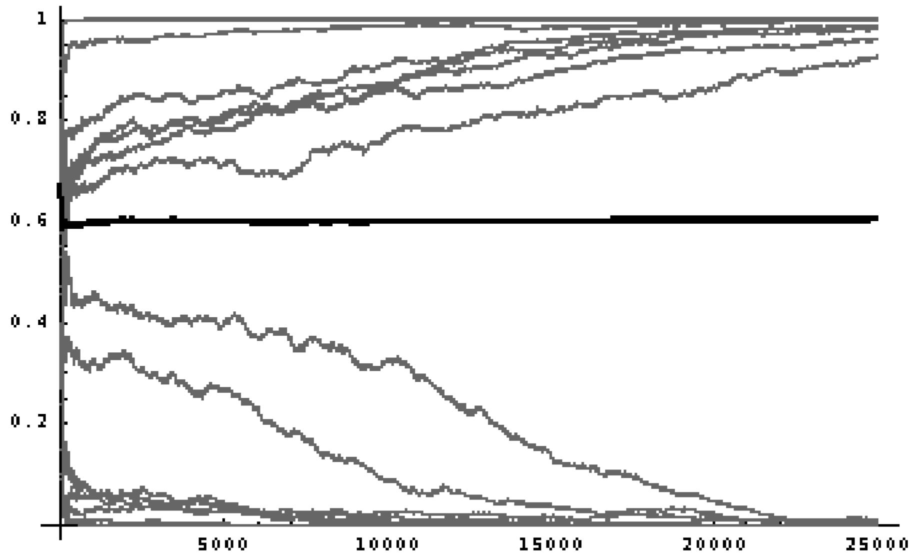

Figure 15. Probability of action 2 for 15 adaptive agents (grey lines) using the RE2 algorithm with 15 non-adaptive agents with probability of action 2 of .75. Average probability of action 2 for all agents is shown by the solid black line.

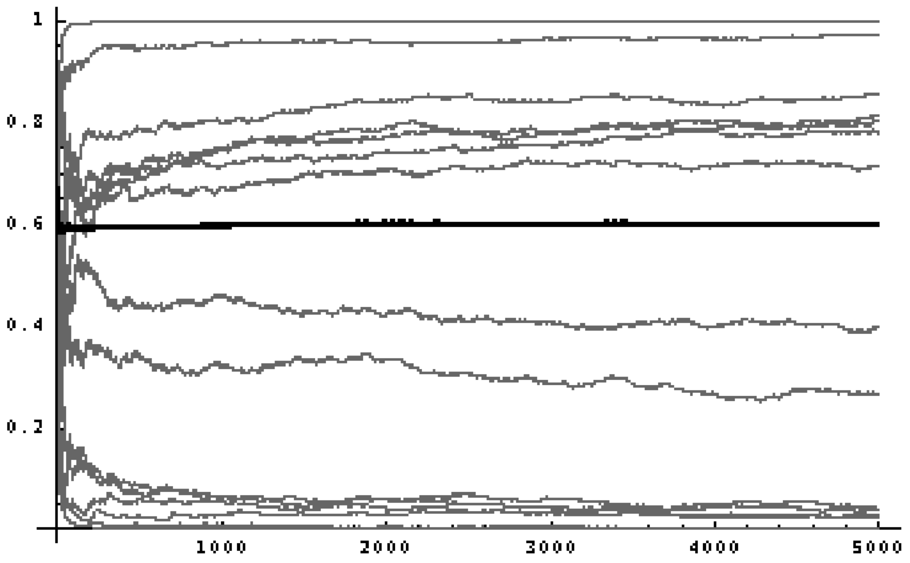

Figure 16. Probability of action 2 for 15 adaptive agents (grey lines) using the RE1 algorithm with 15 non-adaptive agents with probability of action 2 of .75. First 5000 iterations of figure 15.

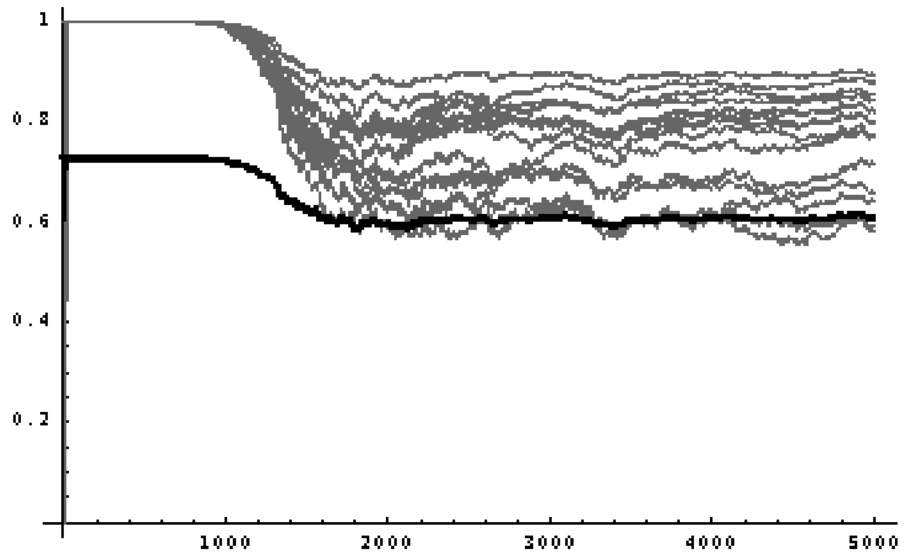

Figure 17. Probability of action 2 for 15 adaptive agents (grey lines) using the CS algorithm with 15 non-adaptive agents with probability of action 2 of .45. Average probability of action 2 for all agents is shown by the solid black line.

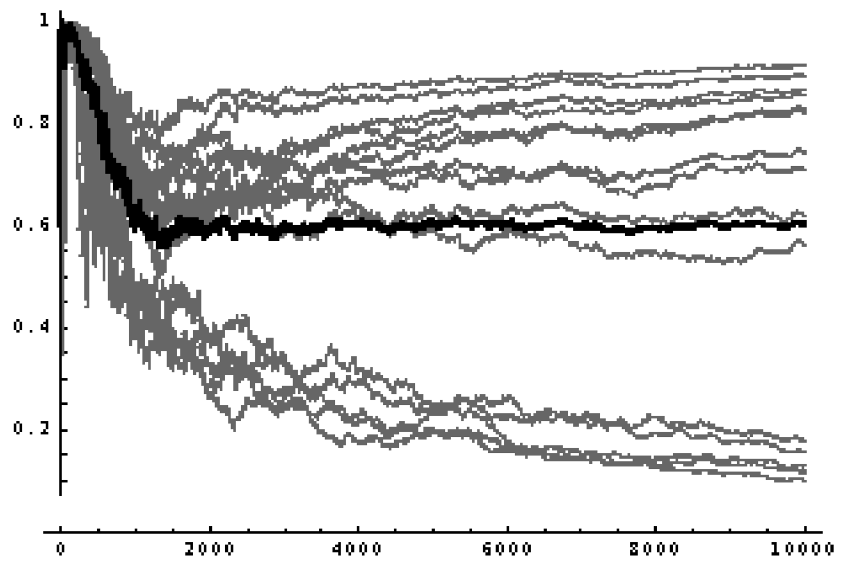

Figure 18. Probability of action 2 for 30 adaptive agents (grey lines) using the CS algorithm with initial probability of action 2 of .60. Average probability of action 2 for all agents is shown by the solid black line.

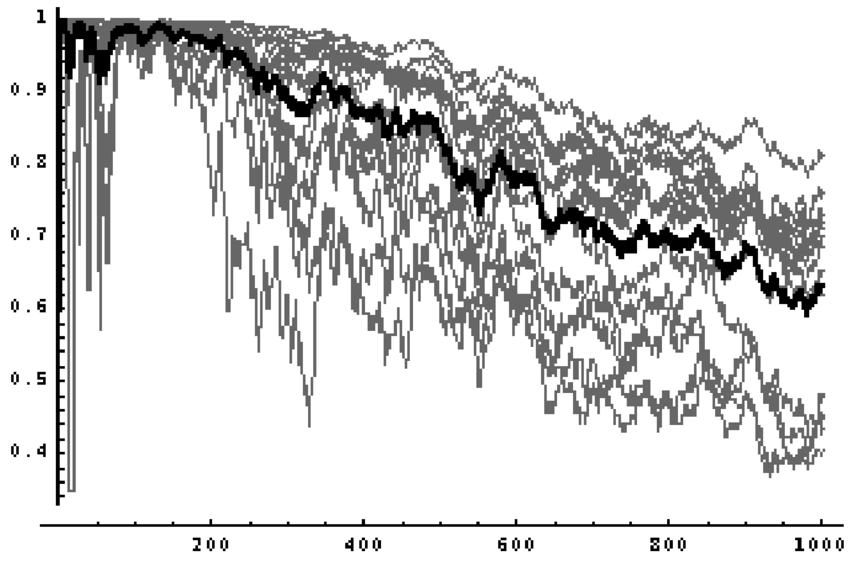

Figure 19. Probability of action 2 for 30 adaptive agents (grey lines) using the CS algorithm with initial probability of action 2 of .60. First 1000 iterations of figure 18.

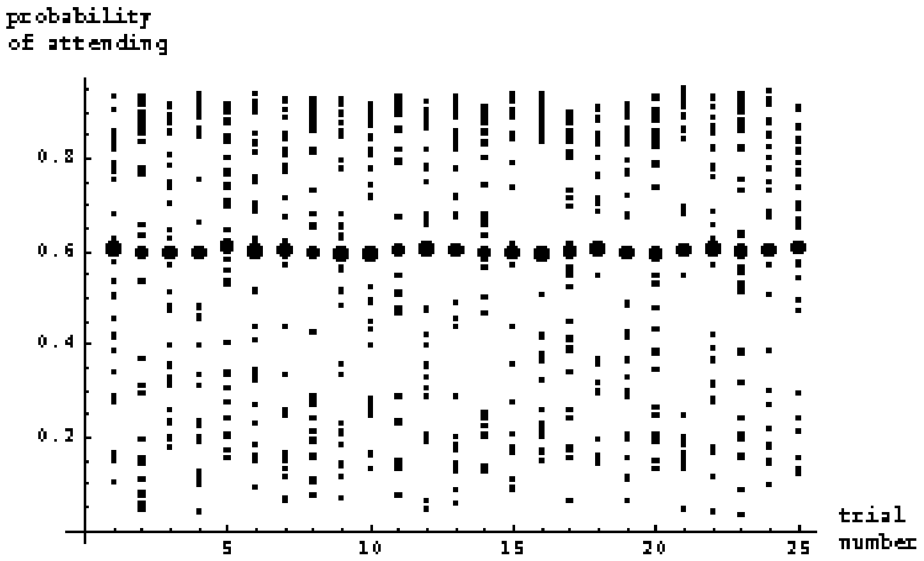

Figure 20. Probability of action 2 for 30 adaptive agents (small dots) using the CS algorithm. Average probability of action 2 for all agents is shown by the large dots. Each vertical column of dots presents the data from one simulation.

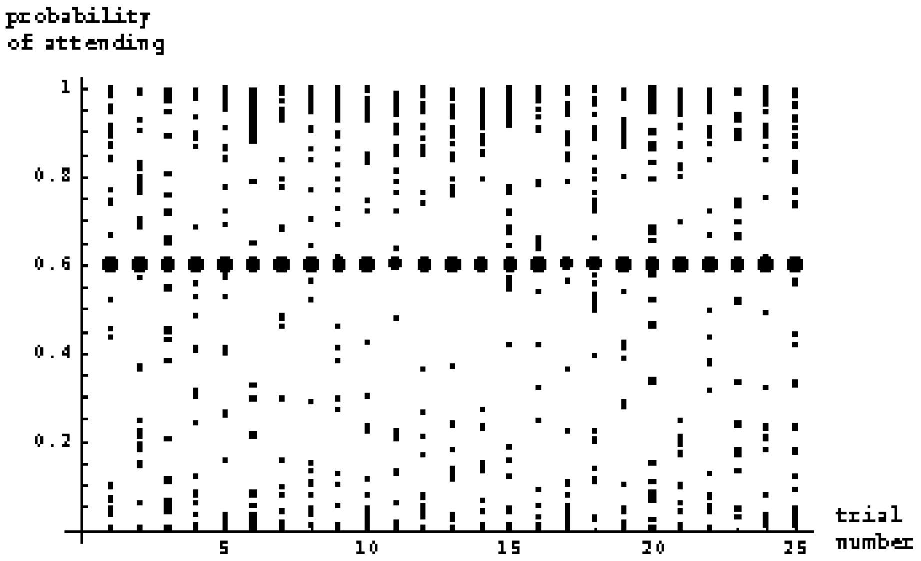

Figure 21. Probability of action 2 for 30 adaptive agents (small dots) using the RE1 algorithm. Average probability of action 2 for all agents is shown by the large dots. Each vertical column of dots presents the data from one simulation.

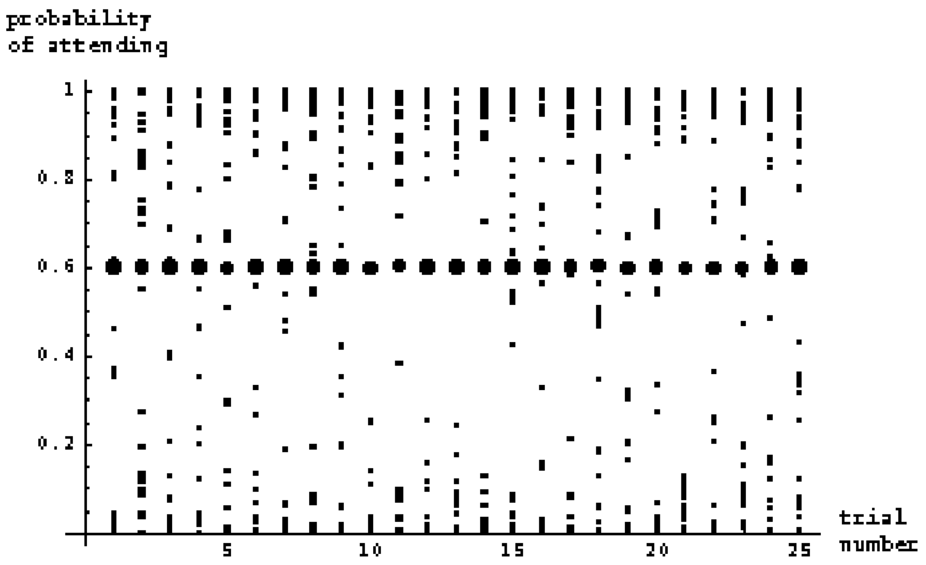

Figure 22. Probability of action 2 for 30 adaptive agents (small dots) using the RE2 algorithm. Average probability of action 2 for all agents is shown by the large dots. Each vertical column of dots presents the data from one simulation.