ELSEVIER

Journal of Economic Behavior and Organization
Vol. 29 (1996) 87–112

JOURNAL OF
Economic Behavior
& Organization

The coevolution of automata in the repeated prisoner's dilemma

John H. Miller

Carnegie Mellon University and the Santa Fe Institute, Department of Social and Decision Sciences, Pittsburgh PA 15213, USA

Received 13 September 1993; revised 4 April 1994

Abstract

A model of learning and adaptation is used to analyze the coevolution of strategies in the repeated Prisoner's Dilemma game under both perfect and imperfect reporting. Meta-players submit finite automata strategies and update their choices through an explicit evolutionary process modeled by a genetic algorithm. Using this framework, adaptive strategic choice and the emergence of cooperation are studied through 'computational experiments.' The results of the analyses indicate that information conditions lead to significant differences among the evolving strategies. Furthermore, they suggest that the general methodology may have much wider applicability to the analysis of adaptation in economic and social systems.

JEL classification: C63; C70; D80; L13

Keywords: Adaptation; Evolution; Learning; Repeated prisoner's dilemma game; Genetic algorithms; Machine learning

1. Introduction

The search for an appropriate way to model the strategic choices of agents has been a central topic in the study of game theory. While a variety of approaches have been used, few of them have explicitly incorporated notions of bounded rationality, learning, and adaptation. Rubinstein (1986) analyzed meta-agents who optimized their selection of strategies constrained by the costs of implementing

SSDI 0167-2681(95)00052-6

such strategies. This paper builds on this approach by modeling the meta-agent's choice through an explicit evolutionary process–a genetic algorithm (Holland, 1975; Goldberg, 1989). By creating artificial adaptive agents (Holland and Miller, 1991) we are able to use 'computational experiments' to study the evolution of strategic choice in games across a general class of adaptive systems. Here, as a simple example of this approach, we analyze the coevolution of strategies in the repeated Prisoner's Dilemma (RPD) game with perfect and imperfect reporting.

Consider the following thought experiment. A group of individuals is about to play a game. In order to participate, players are required to submit a *program* that exactly specifies their moves contingent upon the opponent's reported moves. Initially, the participants have no knowledge of how to play the game, and thus randomly choose their programs. After each round of the game, the actual scores and programs of every player become common knowledge. Based on this information, each person is allowed to adjust his or her program for the next round. Participants submit their new programs, and a new round is initiated. Given such an environment, what types of programs will emerge?

The basic elements of the above scenario encompass important ideas about equilibrium behavior that have emerged from the work of Binmore (1987a, b). Binmore argues that descriptive concepts of equilibrium may be more important than prescriptive ones. However, current descriptive constructs, for example, replicator dynamics and evolutionary stable strategies (Maynard Smith, 1982), often lack the ability to incorporate forms of learning and innovation. The present study removes this restriction, allowing for both learning and innovative processes to enter the model in a tractable manner.

This work assumes that a player's program can be represented by a finite automaton (specifically, a Moore machine–see section 3.2). The idea of selecting a new program based on the results of previous programs is operationalized through the use of a genetic algorithm. Using these elements, the evolving strategic choices of agents are examined under the conditions of a repeated Prisoner's Dilemma game with both perfect and imperfect reporting. The strategies that emerge are classified and their performances are analyzed.

The following research has both theoretical and empirical components. On the theoretical level, elements of bounded rationality and adaptive learning behavior are combined in a general methodological framework. The importance of such models has long been recognized (Simon, 1955, 1959; Day, 1975; and Nelson and Winter, 1982), but tractable models that can display the complex behavior inherent in adaptive learning systems have been difficult to derive. The use of artificial adaptive agents (Holland and Miller, 1991) allows the relatively easy analysis of such complex learning models, and should serve as the basis for the creation of benchmarks that will complement the results from both traditional theory and human experiment. While the major focus of this paper is a game theoretic application of the model, generalizations that capture other social science phenomenon exist, and are discussed in the last section. Empirically, this paper

introduces techniques that allow useful computational experiments to be conducted on a wide variety of games and other complex adaptive economic systems.

2. Background

The potential of automata theory for the analysis of games was first suggested in the economics literature by Aumann (1981). Rubinstein (1986), and Abreu and Rubinstein (1988), studied a RPD in which both players were required to submit strategies in the form of a Moore machines, and found using a dynamic definition of equilibrium that machines will cycle and coordinate their actions, thereby sharply reducing the number of potential equilibria predicted by the Folk theorem (Fudenberg and Maskin, 1986). Moore machines were used to model a form of bounded procedural rationality, wherein players, recognizing that strategies are costly to implement, economize on the size of the machine. Here, we impose additional constraints on the player's ability to *derive* the best procedural rules.

In this paper, the meta-agent's adaptive choice of a strategy automaton is modeled through the use of a genetic algorithm. Fogel et al. (1966), and Axelrod (1987) have presented related applications of the genetic algorithm. Fogel et al. evolved finite automata that attempted to predict a periodic sequence. Besides the obviously different task, their adaptive plan lacked many of the important features required by a genetic algorithm (for example, crossover). Axelrod used a genetic algorithm to evolve RPD strategies that based their moves on the game's past three-move history. There are a number of major differences between his work and this study. First, the environment used here is allowed to coevolve, and varies continuously as the strategic population changes. The major focus of Axelrod's study was on strategies evolving against a fixed environment, one based on eight representative strategies from his earlier tournaments. [1] Second, a wide variety of experiments are conducted in this analysis, most notably, the impact of imperfect reporting. Finally, the use of automata to represent strategies has two major advantages over Axelrod's fixed history strategies: (1) automata are a very flexible description of strategic choice, and thus incorporate many theoretically important strategies that cannot be easily defined under the restriction of the past three-move history (for example, strategies which rely on counting or triggers, etc.), and (2) their analytical possibilities are much richer.

[1] The variable environment case explored here was mentioned in Axelrod's paper, but very little attention was given to it. The last part of Section 5 develops some links between the two approaches.

3. The repeated prisoner's dilemma, finite automata, and evolution

3.1. Repeated prisoner's dilemmas

The game used in this analysis is the repeated Prisoner's Dilemma (RPD). The Prisoner's Dilemma game was first formalized by Tucker (1950), and its current applications span most of social science (see Axelrod and Dion, 1988, for a partial review). Important economic applications include: collusion between firms, trade barriers between countries, and public goods problems. The Prisoner's Dilemma was chosen for this analysis because of its wide applicability, and the potential for direct comparisons of the methodology presented here with the plethora of previous results.

The basic Prisoner's Dilemma is a two-player game, with each player having a choice of either cooperating (C) or defecting (D). The payoffs used in these experiments are the ones typically found in the literature and are presented in Fig. 1. Given these payoffs, it is easily shown that mutual defection is the only Nash equilibrium (it is also a dominant strategy equilibrium). Of course, the intrigue of the Prisoner's Dilemma is that this unique equilibrium is Pareto inferior to the mutual cooperation outcome. If the basic Prisoner's Dilemma is iterated, the resulting supergame is a RPD. If the number of iterations is a *known* finite number, then a simple backward induction argument implies that the only equilibrium is mutual defection in every round. However, if the game is repeated a finite but unknown number of times, or if it is played an infinite number of times with discounting or payoff averaging, then cooperative outcomes can theoretically emerge–in fact, the folk theorem implies that with sufficiently little discounting, any individually rational outcome can be supported as a (subgame-perfect) Nash equilibrium.

The actual behavior of human subjects in the RPD has been widely analyzed (see the references cited in Shubik, 1982, pp. 400–401). Axelrod (1984) conducted two tournaments that used computerized strategies submitted by subjects from a variety of backgrounds. His analysis indicated that the most effective strategy in the tournaments, Tit-For-Tat (TFT), was also the least complicated. TFT begins by cooperating and then mirrors the opponent's last move. The primacy of TFT was somewhat surprising given the level of sophistication of other strategies entered in the tournament.

	C_2	D_2
C_1	3,3	0,5
D_1	5,0	1,1

(The payoffs are ordered Player 1, Player 2.)

Fig. 1. The basic prisoner's dilemma.

While a large amount of analysis exists for the RPD under conditions of perfect information, very little exists for the game under imperfect information. Here we consider imperfect reporting of the opponent's actual moves. That is, a noise level of $\alpha\%$ indicates that $\alpha\%$ of the time an opponent's move is reported to be the opposite of what the opponent actually did, while the remainder of the time the move is perfectly transmitted. For this notion of reporting noise to make sense in the context of a repeated game, we require a delay or aggregation of subgame payoffs (otherwise, the payoff information would be sufficient to reveal the actual move). Examples of situations with delayed payoffs and imperfect verification include arms treaties and oligopolistic production agreements. The presence of noise in the system implies that strategies should not only react to the misreporting, but also try to exploit it. Thus, programs that discount reported defections due to the noise, may fall victim to strategies that intentionally defect hoping for either a forgiving opponent or a reporting error.

The RPD is a natural choice for inclusion in these experiments. Techniques that allow carefully controlled experimentation with the model under a variety of situations will not only increase our current knowledge about the game's characteristics, but also expand the possible set of applications. The RPD is a member of a much broader class of games, and therefore procedures used with this game may be easily transferred into related domains. A key to maintaining this generality is finding a convenient, yet flexible, representation for strategies in the game.

3.2. *Finite automata*

Finite automata mathematically model a system that responds to discrete inputs and outputs. The models arising from finite automata 'capture the notion of a fundamental class of systems, a class rich in structure and applications' (Hopcroft and Ullman, 1979, p. 14). The actual applications of finite-state systems range from the analysis of computational processes and neural networks to a theoretical understanding of costly strategic choice in games. This latter application is of most interest to this work, however, the vast modeling potential of these techniques hints at a far richer set of potential applications for the general methodology developed here.

The specific type of finite automata used here is a Moore machine. A Moore machine designed to play the RPD is described by four elements. [2] The machine consists of a set of *internal states*. One of these states is designated as the *starting state*, and serves as the initial state of the machine. Every internal state has

[2] Formally, a Moore machine is described by a four-tuple $\langle Q, q_0, \lambda, \delta \rangle$, where Q is a finite set of internal states, $q_0 \in Q$ designates the starting state, λ: $Q \rightarrow S_i \in \{C, D\}$ where S_i is the player's move next period, and δ is the transition function which maps the current internal state of the machine and the reported move of the *opponent* into a new internal state, δ: $Q \times S_{\sim i} \rightarrow Q$ (where $S_{\sim i} \in \{C, D\}$ is the opponent's reported move last period).

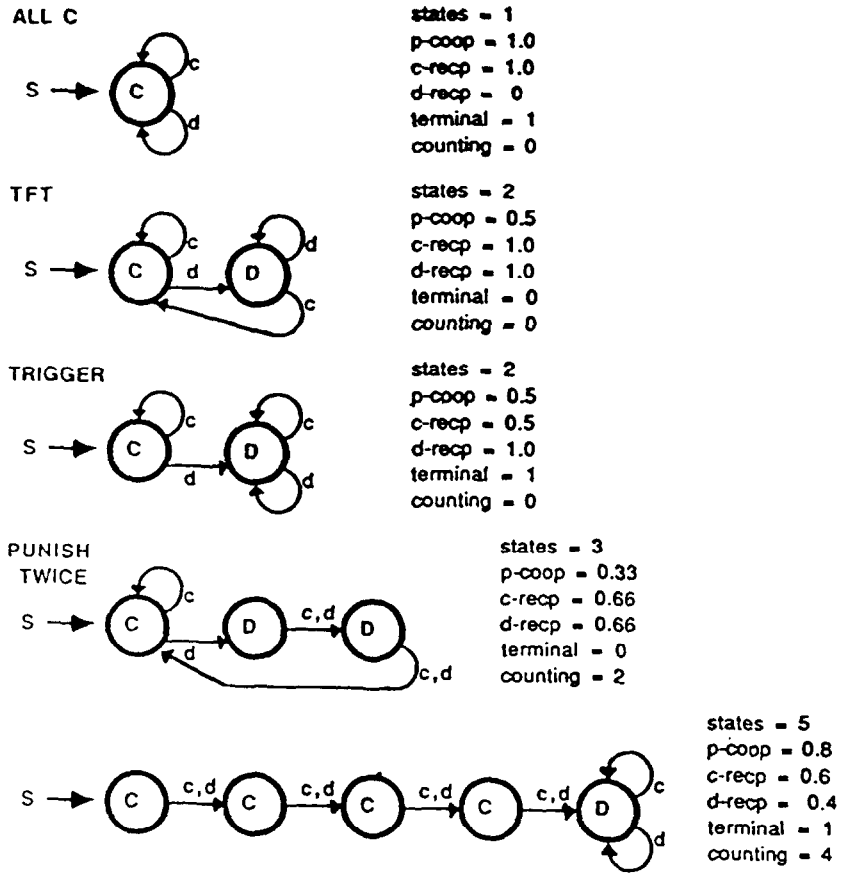


Fig. 2. Some possible automata.

associated with it a single strategic action, thus in the RPD every state indicates whether the machine will cooperate or defect during the next period. Finally, there is a transition function associated with *each* internal state that determines the next internal state given the reported action of the opponent. The transitions may go to any of the internal states (including the current one), and are always conditional on the current state of the machine and the reported move of the opponent. Thus, a machine begins in its starting state and does the action specified in that state (either cooperate or defect). The machine then moves to a new internal state based on the reported move of the opponent, and proceeds with the action specified in the new state. This process will continue until the game ends.

A more intuitive description of an automaton is given by its transition diagram (see Fig. 2 for some examples). The nodes of the transition diagram represent the internal states, with the upper-case labels inside of the nodes showing the move

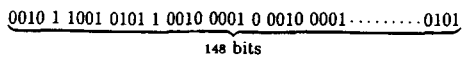
that the machine will make when it enters that state. The transition function is specified by the labeled arcs emerging from each node, where the lower-case label indicates the observed move of the opponent and the arc points to the next state of the machine. The starting state is indicated by the arc labeled *S*. For example, the first machine in Fig. 2 always cooperates, regardless of the opponent's actions. The second machine models TFT. It starts in the left-hand state and cooperates. If the opponent is reported as cooperating, it stays in the left-hand state and again cooperates. However, if a defection is reported, a transition occurs to the right-hand state and the machine issues a defection. The automaton will remain in the right-hand state (and thereby continue defecting) until a cooperation is observed by the opponent, at which time a transition to the left-hand state, and thus cooperation, ensues. The third machine is a trigger strategy, that begins by cooperating and continues to do so unless the opponent defects. If a defection occurs, the automaton enters a terminal (absorbing) state of defection. Once in the terminal state, there are no possible transitions that will change the automaton's internal state, and therefore it will defect for the remainder of the game. The fourth automaton describes a strategy that always cooperates, unless the opponent is observed to defect. If a defection is observed, this strategy will defect for two consecutive turns, and then return to the cooperative state. The final machine begins by cooperating four times in a row and then defects for the rest of the game. As is apparent from the previous descriptions, automata capture a large set of potential strategies, including a number of those strategies that have been of central importance to various earlier studies.

The relevant history of a game is contained in the automaton's current state. A strategy that is based on the past n moves of either the opponent or itself will require a maximum of 2^n internal states. Thus, a TFT strategy, which must only remember the opponent's last move, requires two states, while a strategy that bases its moves on the full history of the last two rounds (including both the opponent's and its own last two moves) requires at most sixteen states. Also note that although an automaton can have, say, sixteen states, only a subset of these states may be accessible given the starting state and transitions inherent in the machine. That is, there may be states in the machine that are impossible to reach during the course of the game.

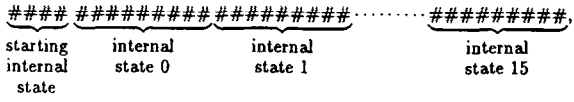
The evolutionary mechanism used in this paper requires strategies to be specified in a well-defined language. Here, each Moore machine is represented by a string of 148 bits (see Fig. 3). The first four bits provide the starting state of the automaton. [3] Sixteen nine-bit packets are then arrayed on the string. Each packet represents an internal state of the automaton. The first bit in a given packet describes the move next period whenever the automaton is in that state (0 = cooperate, 1 = defect), the next four bits give the transition state if the opponent is

[3] A string of four bits can represent $2^4 = 16$ values.

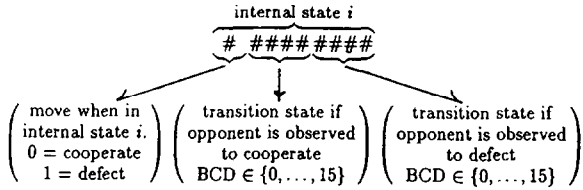
A sample automaton:



The structure of an automaton:



where $\# \in \{0, 1\}$, the starting internal state is a binary coded decimal (BCD) number in $\{0, \ldots, 15\}$, and each internal state i has the following structure:



Using this scheme the sample automaton at the top of the figure defines a Tit-For-Tat strategy, which uses internal states 1 and 2. The machine begins in internal state 2 where it cooperates. If the opponent cooperates, it remains in state 2, while if the opponent defects a transition to state 1 occurs. In internal state 1 the machine defects, and remains in state 1 until the opponent cooperates, in which case it returns to state 2.

Fig. 3. The "biology" of automata.

observed to cooperate, and the final four bits give the transition state if a defection is observed. This scheme allows the definition of any RPD Moore machine of sixteen states or less. Since bits are restricted to two values, there are 2^{148} unique possible structures. [4]

Automata have emerged as a tractable way to model bounded rationality considerations in the theory of games. The class of Moore machines encompasses an interesting and theoretically important class of strategies. By introducing an evolutionary process, a synthesis occurs that allows the derivation of tractable adaptive learning models.

3.3. Evolution

The evolutionary process used in this analysis is derived from a class of optimization routines from computer science called genetic algorithms. These

[4] Although there are 2^{148} possible strings, the total number of unique strategies is much less, since some of the automata are isomorphic, for example, there are 16! ways to relabel the internal states of each of these machines. Even with this duplication, the number of unique strategies is still very large (on the order of 10^{21}, or so).

1) Initial random population of 30 structures indexed by i, $t = 1$.
2) Test each structure against the environment ($\hat{\mu}(i,t)$ = score).
3) Form a new population of 30 structures.
   a) Top 20 from the old population.
   b) Create 10 new structures via crossover and mutation:
      i) Select 2 parents: Prob(i) = $\hat{\mu}(i,t)/\sum_j \hat{\mu}(j,t)$.
      ii) Form 2 children by applying the crossover operator to the parents
      iii) Mutate the newly formed children.
      iv) Repeat (i) through (iii) until 10 new structures are formed.
4) Increment t by 1 (next generation), and iterate (go to Step 2).

Fig. 4. The adaptive plan.

algorithms were developed by Holland (1975) for optimization problems in *difficult* domains. Difficult domains are those with both enormous search spaces and objective functions with nonlinearities (many local optima), discontinuities, high dimensionality, and noise. Genetic algorithms provide a highly efficient mechanism for effectively searching these spaces. Furthermore, their underlying structure indicates that they may be an appropriate model of certain types of adaptive learning behavior (Miller, 1986). Finally, the existing literature from computer science provides important analytic and empirical results regarding the algorithm's, and hence the model's, behavior.

Genetic algorithms are a large class of routines that share the following characteristics: a *population* of well-defined structures acts in an environment and receives information on each member's performance, using this information new populations are formed by selecting the better performing structures and modifying them through genetic operators. The genetic algorithm used here is shown in Fig. 4. Initially, thirty structures (strategies) are chosen at random. Each structure is then tested against the environment (which, here, is composed of the other structures) and receives a performance score (total payoff in the RPD). Given the resulting scores, a new generation of structures is chosen by allowing the top twenty performers to go directly into the next generation. Ten new structures are also created by mating. The mating process occurs by probabilistically selecting two parents from the old population (with the probabilities biased by their scores), and then forming two children through a process of crossover and mutation.

The crossover and mutation operations are both important elements of the algorithm, as well as interesting ways to tractably model adaptive learning behavior. In order to use these operators, structures must be defined in an easily manipulable language. Here, structures are represented as binary strings, with each address on any given string controlling a particular aspect of the final structure (see Fig. 3 for the mapping). The crossover operator works as follows: two structures are chosen as parents and a single crossover point, c, is randomly selected on the bit string. The first child is formed by taking the first c bits from the first parent and concatenating them to the bits following the c'th bit of the

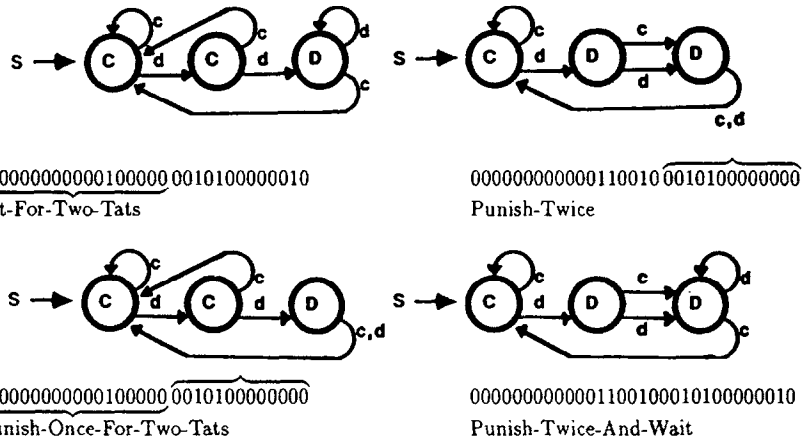


Fig. 5. An example of crossover on some automata.

second parent. The second child is formed in a similar way using the remaining portions of the two parental strings. Mutation occurs when a bit at a random location on the string changes states.

The effect of crossover on the new members of the population obviously depends on the exact location and length of the crossover. The impact of crossover ranges from a simple change in the starting state (even this could have a large impact if previously inaccessible states become active) to a more radical recombination of the two parents. An actual crossover is illustrated in Fig. 5. The first parent describes a strategy that defects after two consecutive defections by the opponent and continues defecting until the opponent cooperates (Tit-For-Two-Tats (TF2T)), the second one cooperates as long as the opponent cooperates, but returns any defection with two rounds of defection (Punish-Twice). When these two parents recombine by a crossover at the indicated locus, the children inherit traits from both parents. The first child punishes once for two consecutive defections by the opponent (a more forgiving strategy than TF2T), and the second child immediately punishes a defection by the opponent with two rounds of defection and then waits for the opponent to reestablish cooperation (a "meaner" strategy than Punish-Twice).

The adaptive plan described above has three major components: (1) reproduction based on performance, (2) recombination (crossover), and (3) mutation. The combination of these three elements results in a very powerful optimization algorithm. At first glance it may appear that the plan is no more than 'random search with preservation of the best [structure]' (Booker et al., 1987, p. 23). However, the algorithm is actually a sophisticated sampling procedure that devel-

ops optimized structures by independently manipulating important structural building blocks.

The incorporation of the crossover and mutation operators along with reproduction by performance, results in a powerful search algorithm. Useful structural patterns increase or decrease based only on their own observed performances–independent of how the full structures are changing. Holland (1975), pp. 121–40, demonstrated that the rate at which patterns are sampled closely corresponds to the optimal sampling path in the canonical n-armed bandit problem, regardless of the form of the payoff function. While the adaptive plan is generating an appropriate sampling plan for the existing patterns, it is simultaneously generating new patterns to test. These modifications are implemented in such a way that high interim performance levels are maintained. Finally, the plan accomplishes this while avoiding entrapment on false peaks. (For a more formal treatment of the workings of genetic algorithms, see Holland (1975), Goldberg (1989), and Vose (1991).)

The performance of the genetic algorithm has been extensively studied. Frantz (1972) showed that the algorithm effectively adapted to highly nonlinear systems. Martin (1973) investigated the asymptotic properties of a similar class of adaptive plans. She found that under certain restrictions the adaptive plan converges to a set of 'good' structures. DeJong (1975) simulated various versions of the algorithm over a variety of environments including: continuous, discontinuous, unimodal, multimodal, convex, nonconvex, low-dimensional, high-dimensional, and noisy functions. His results, later corrected by Bethke (1981), indicated that the genetic algorithm performed better than commonly used function optimization techniques.

The above adaptive plan closely corresponds to the thought experiment discussed in the introduction of the paper. The idea of players adaptively learning when resubmitting their programs is modeled in two ways. The first is an *imitative* component, that allows players to exactly copy the best performing programs. This is implemented when the plan admits the top twenty performers into the next generation. The second component is an *innovative* one, whereby players form new programs by combining different parts of existing programs (crossover), along with some unique modifications (mutation).

Finally, note that genetic algorithms represent a robust and broad class of adaptive algorithms. Such algorithms only require populations of 'solutions' to be reproduced by performance and to have new members created via genetic operators (at least crossover and mutation). The algorithms are extremely robust to actual algorithmic and parametric choices. For example, all of the major qualitative results reported here have been confirmed using a variety of different parameters (e.g., mutation rates, etc.) and algorithmic components (e.g., different strategy representations, selection rules, etc.). Thus, there appears to be a large equivalence class of adaptive behavior that can be captured by genetic algorithms.

4. Methodology

The three ideas discussed in the previous section are combined to form a general methodology: an adaptive plan, based on the genetic algorithm, is used to evolve automata that play a RPD. The advantage of this methodology is that the very complex problem of strategic choice in the RPD can be analyzed empirically through computational experiments. As previously discussed, the genetic algorithm is very good at optimizing functions in complex environments. Furthermore, the algorithm's underlying mechanisms are appealing as appropriate analogs for a model of adaptive learning. The use of automata allows a very large set of potential strategies to be easily incorporated into the algorithm. By combining the three elements, repeatable and recoverable controlled experiments can be conducted within the theoretical framework.

4.1. Some technical details

Experiments were conducted under various information conditions. [5] Three levels of informational accuracy were explored: perfect information, 1%, and 5% noise (where the noise level is the probability that any actual move is misreported). The 1% noise level implies that 3.0 misreports per supergame can be expected while the 5% level is associated with 15.0 misreports. Forty runs were conducted under each of the conditions to allow for stochastic variations.

The game and algorithmic parameter choices were guided by considerations of robustness and computational constraints. A variety of sensitivity analyses have been performed, and they confirm that the results reported here appear robust to reasonable changes in these choices. The initial population for each run consisted of thirty randomly generated automata. Once created, the population was iterated for fifty generations. In every generation, each automaton was matched against each of the other automata and a clone of itself for a 150 round RPD. [6] Payoffs were calculated using the values in Fig. 1. An automaton's final score was the sum of the payoffs for each match (when the clone was played, the player was assigned the average score of the match). The choice of payoff metric was dictated by the existing repeated games literature. Given the size of the machines, it is likely that they enter into steady state behavior early in the game, and thus this measure

[5] All programs were written in Pascal by the author. The major programs were run on an IBM-XT with a Hauppauge 386 MotherBoard (16 MHz Intel 80386 CPU), after compiling the programs in Borland International's TURBO PASCAL version 4.0. Random numbers were generated using the routine supplied by the compiler, and were then shuffled following the procedure outlined in Press et al. (1986), p. 195.

[6] Under conditions of imperfect information, five such supergames were performed between every pair to reduce the impact of stochastic variations. Also note that since the automata were limited to sixteen states, the 150 round game is well beyond their intrinsic memory capacity.

should be closely correlated with long term average performance. At the end of each generation, the genetic algorithm discussed earlier (see Fig. 4) was applied to form a new population. [7]

5. Results

The results of the analysis indicate that the approach outlined above provides insights into the evolution of strategic choice and the emergence of cooperation in the RPD. Furthermore, they imply that significant differences exist among the perfect information (PIE), 1% noise (1%NE), and 5% noise (5%NE) environments. The analysis focuses on the evolution of some important attributes of the individual automata, the population, and also some experiments concerning the robustness of the final strategies. Additional results can be found in Miller (1988).

The majority of the analysis that follows presents the averages over all thirty members of each population and forty simulations conditional on the generation and the noise level in the environment. [8] Unless otherwise specified, a test based on a one-tailed likelihood ratio technique [9] (see, Freund and Walpole, 1980, p. 393) was used to determine whether the means were significantly different from

[7] The final scores for each automaton were normalized by taking $\hat{x}_i = (x_i - \mu)/\delta + \alpha$, where x_i is the automaton's raw score, μ is the sample mean, δ is the sample standard deviation, and $\alpha = 2$ is a parameter that determines the importance of relative performance. The $\hat{x}_i$ values below 0 were truncated at 0. This normalization procedure eliminates difficulties associated with negative scores, and more importantly is immune to affine transformations of the payoff function. The choice of $\alpha = 2$ implies that automata which do worse than two standard deviations from the mean are not allowed to mate. It also determines the importance of relative performance (as $\alpha \to \infty$ and $\alpha \to 0$ the selection probabilities go to $1/N$ and $\hat{x}_i / \Sigma_j \hat{x}_j$ respectively). Two parents from the old population were randomly selected, where the selection probability for choosing automation i was $\hat{x}_i / \Sigma_j \hat{x}_j$. A cross over point, $c \in \{1,\ldots, 148\}$, and length, $l \in \{1,\ldots, 147\}$, were randomly selected and two new automata were formed by exchanging the l bits starting at the c'th position of each parent. This crossover procedure is slightly different from the one previously described, and assumes that the strings are actually circular rather than linear. This eliminates a bias towards preserving the end points which is inherent in the linear procedure. After crossover, each bit was subjected to a 0.5% independent chance of mutation (implying an expectation of 0.74 bit mutations per string with a variance of 0.74 bits). The mating procedure was repeated until ten new members were formed. This new population was then matched as before. Again, note that the above procedures are included for completeness, and that the results are robust to a variety of parametric and algorithmic changes. Subsequent work with alternative selection and reproduction schemes indicate that the results are very robust to these choices.

[8] Where alternative approaches are used, they are noted in the text.

[9] This test does not place excessive requirements on the underlying distribution of the random variables so long as sample sizes (here, samples are of size forty) are large enough for an application of the central limit theorem. The stochastic process describing the genetic algorithm is likely to be very complicated and not Gaussian. Throughout most of the analysis the focus is on the means of the environments. Given the potential for unusual distributions, it is likely that other statistics might also be illuminating.

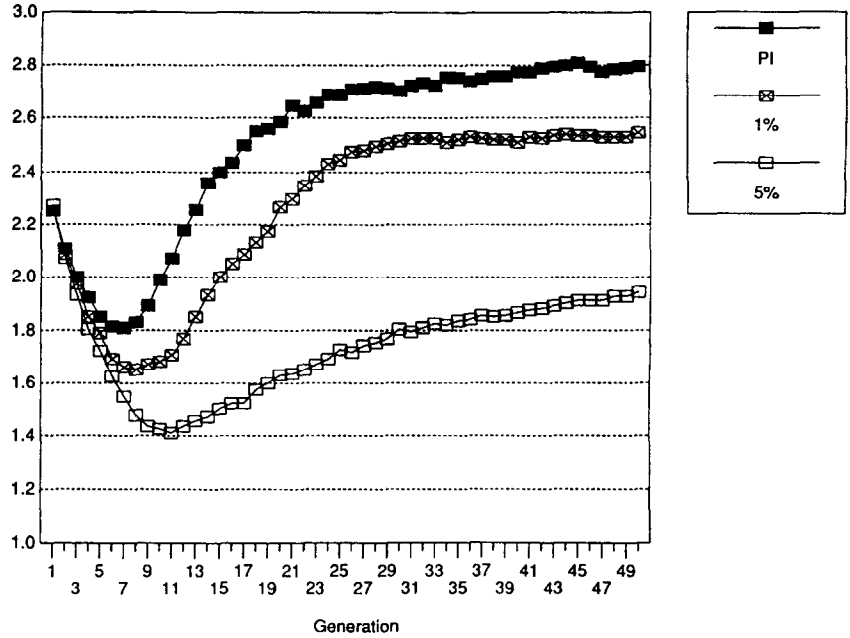


Fig. 6. Average payoff per game iteration.

one another in pair-wise comparisons at a 97.5% level of significance. Given the potential path dependence of any given population, the average characteristics of the population are considered to be the unit of analysis. In essence, this assumption implies that the expected characteristics of a given population as opposed to an individual are important. This perception is consistent with concerns about the overall performance of strategies in a given environment. [10]

5.1. The evolution of payoffs

Fig. 6 shows the average payoff over all of the automata in the relevant populations. The payoff is in terms of a single iteration of the game. If there is always mutual cooperation, then the expected payoff would be 3.0, mutual defection would imply a payoff of 1.0, and strategies that randomly choose moves would expect to receive 2.25. The average payoff path in the three environments quickly trifurcates, and each remains significantly different from the others past the sixth generation. Initially, the average payoff in all three environments is about 2.26. The expected payoff experiences a steady decline in all of the environments

[10] The impact of changing the unit of analysis to the individual is minimal, and in fact, tends to increase the significance levels of the statistical tests.

over the first few generations (with maximum declines of around 7% per generation). Starting with the PIE in the seventh, the 1%NE in the eighth, and the 5%NE in the eleventh generations, each payoff begins to increase after reaching successively lower turning points (1.81, 1.65, and 1.41 respectively). After a period of rapid improvement (with maximum increases ranging from 3% to 5% per generation), the average payoff tends to plateau by the twenty-fifth generation. [11] The final payoff levels are about 2.80, 2.54, and 1.94 for the PIE, 1%NE, and 5%NE respectively. Implying that the expected performance diminishes by about 9% in the 1%NE and 31% under the 5%NE. Under the PIE, payoffs in the final generation tend to be skewed towards the upper end of the distribution. In the 1%NE the final distribution is slightly bimodal, while under the 5%NE an obvious bimodal pattern emerges. The bimodality of the noisy distributions indicates a definite path dependence for these latter populations–after the initial generation a bifurcation occurs in which some populations achieve high payoffs and others do quite poorly.

Fig. 6 illustrates how cooperation can emerge in these systems. Note that in the early generations the agents tend to evolve strategies that increasingly defect. These conditions do not however persist, and at some point there is an emergence of cooperative strategies that tend to proliferate throughout the population under the low noise conditions (a similar result was found by Axelrod, 1987). A relatively simple explanation underlies these dynamics. Initially, the strategies are generated at random, and therefore the best strategy in such an environment is to always defect. Thus, in the early generations the population of strategies tends to evolve towards always defecting. Although always defecting is a good strategy in a random environment, if some strategies can achieve mutual cooperation, they could do quite well. In fact, as later results will confirm, this is exactly what happens–a few strategies begin to reciprocate cooperation, perform well, and proliferate in the population.

5.2. *The evolution of automaton characteristics*

Given the enormous number of possible automata, it is necessary to develop some summary measures of automaton behavior to facilitate the analysis. The measures developed here were guided by results from the existing theoretical and empirical literature, and by no means exhaust the set of possible descriptive statistics. The measures succinctly describe some of the important dimensions of the strategies (see Fig. 2 for some examples). They may not, however, always

[11] Under the 5%NE the leveling off is not as pronounced, however, the estimated exponential growth rate at the final generation using the previous ten generations is approximately 0.34%. If this rate were to continue the average payoff would reach 2.25 in about 44 more generations.

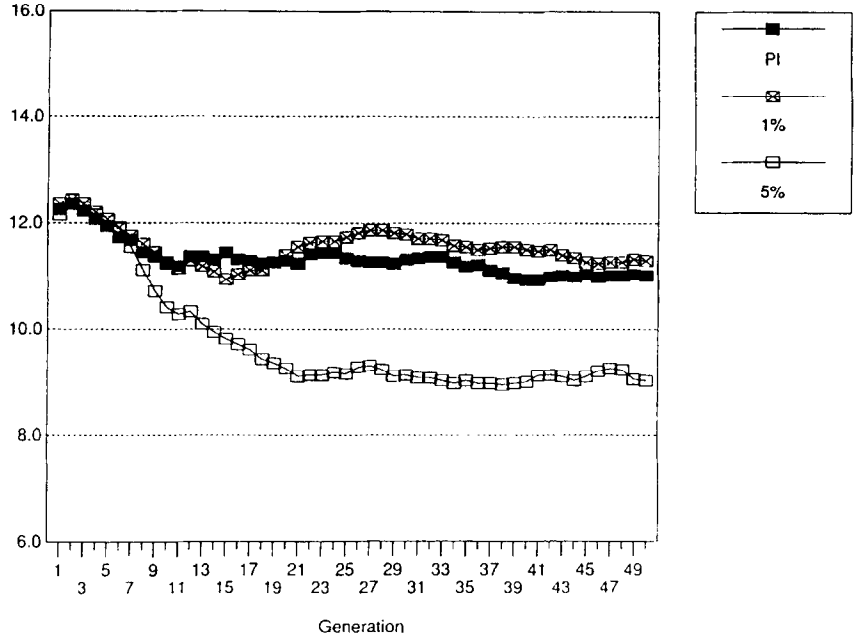


Fig. 7. Average accessible states per minimized automaton.

prove adequate, as they give equal weight to all accessible states (the last automaton in Fig. 2 illustrates potential problems).

The first characteristic is the size of the automaton, which is given by its number of accessible states. A state is accessible if, given the automaton's starting state, there is some possible combination of opponent's moves that will result in a transition to the state. Therefore, even though all automata are defined on sixteen states, some of these states can never be reached during a game. The theoretical literature has often used this variable as a determinant of complexity (Rubinstein, 1986). [12] Automaton theory demonstrates that there exists a minimal state machine for any given behavioral pattern, and that this machine is unique up to an isomorphism (see, for example, Harrison, 1965, Chapter 11). Thus, all of the measures of an automaton's behavior used in this analysis are based on the implied minimal state machine.

The average number of accessible states for the minimized automata is shown in Fig. 7. The initial, randomly chosen, automata averaged about 12.25 states out

[12] Alternative measures of complexity in automata do exist. For example, following the work of Khrone and Rhodes, automaton could be decomposed into their prime components, and then complexity measures based on the type and connections of these components could be developed (see Arbib, 1968, Chapters 3, 5, and 6).

of a possible sixteen. The number of states in the PIE and 1%NE declines until the eleventh generation, at which point it levels off after about a 10% reduction. Under the 5%NE the decline continues until about the twentieth generation, after which time the number of states stabilizes at a 25% lower level than in the first generation. In the final generation the number of states is 11.01, 11.29, and 9.03 in the PIE, 1%NE, and 5%NE respectively. This implies that under the more extreme noise condition, about 20% fewer states develop. All of the declines are statistically significant, as is the difference between the number of states in the 5%NE verses the PIE and 1%NE past the fifteenth generation. If the number of states is a good measure of complexity, then a possible conclusion is that strategic simplification, especially in the presence of noise, is advantageous in the RPD. Theoretical models (e.g., Heiner, 1983) suggest that simple rules of thumb may be superior in the presence of uncertainty. Further analysis indicates that the high noise strategies tended to rely on the use of terminal states, thus supporting this hypothesis. Alternative explanations for the complexity loss are also possible. For example, simple machines may be more likely to arise during early evolutionary dynamics (and, the fifty generation time scale may be too short to observe any rise in complexity), or perhaps effective complex machines require more than the maximum allowed sixteen states.

Notions of the actual behavior of a given machine during a RPD are derived from the actions and transitions of each accessible state. The cooperation-reciprocity (defection-reciprocity) is the proportion of accessible states that return an observed cooperation (defection) by the opponent with a cooperation (defection). These reciprocity measures give only a general notion of a strategy's reactions, since they assume that all accessible states are equally likely. The importance of reciprocity is suggested by the work of Axelrod (1984). Figs. 8 and 9 show the respective evolution of cooperation-reciprocity and defection-reciprocity. Notwithstanding the environment, strategies were always more likely to reciprocate a defection by the opponent with a defection than to cooperate after a cooperation. Similar to the pattern observed throughout the analysis, a period of rapid adjustment is followed by some minor corrections, and then a period of relative stability. Differences between the PIE and 1%NE verses the 5%NE are statistically significant past the seventh generation. [13] The differences between the PIE and 1%NE are not statistically significant at the group level of analysis, but are significantly different past about the twentieth generation when the individual is used as the unit of analysis. The final values for the cooperation-reciprocity are 0.44, 0.49, and 0.36 in order of increasing environmental noise. For the defection-reciprocity the corresponding values are 0.70, 0.65, and 0.76. Therefore, with high noise levels, evolved automata do not reciprocate cooperation as much, and are less

[13] The cooperation-reciprocity measure for the PIE and 5%NE are only significantly different at the 95% level over the final eight periods.

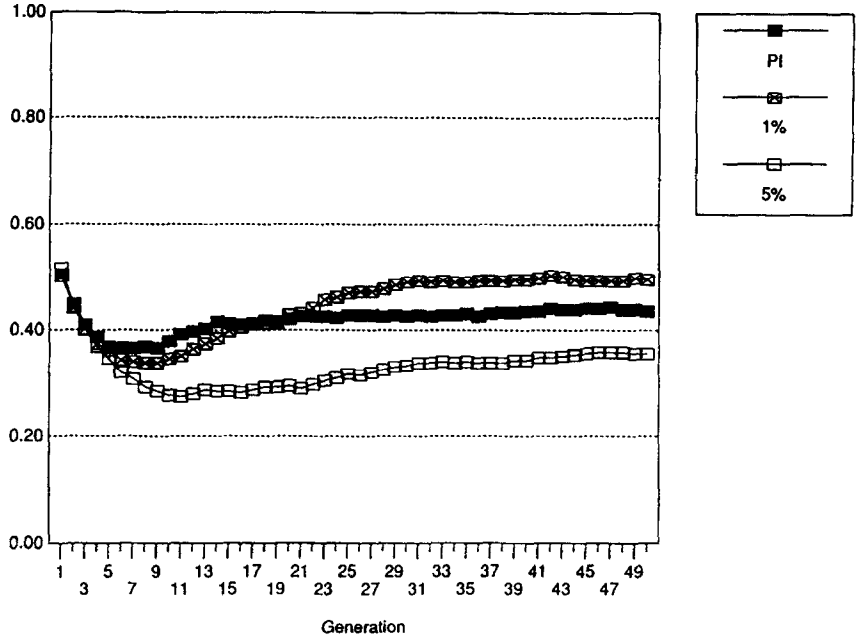


Fig. 8. Average cooperation reciprocity.

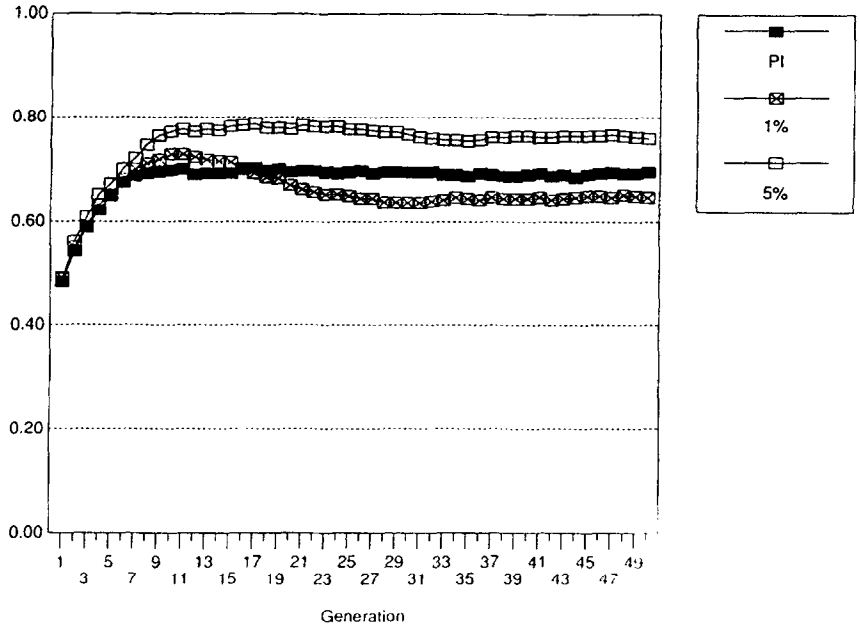


Fig. 9. Average defection reciprocity.

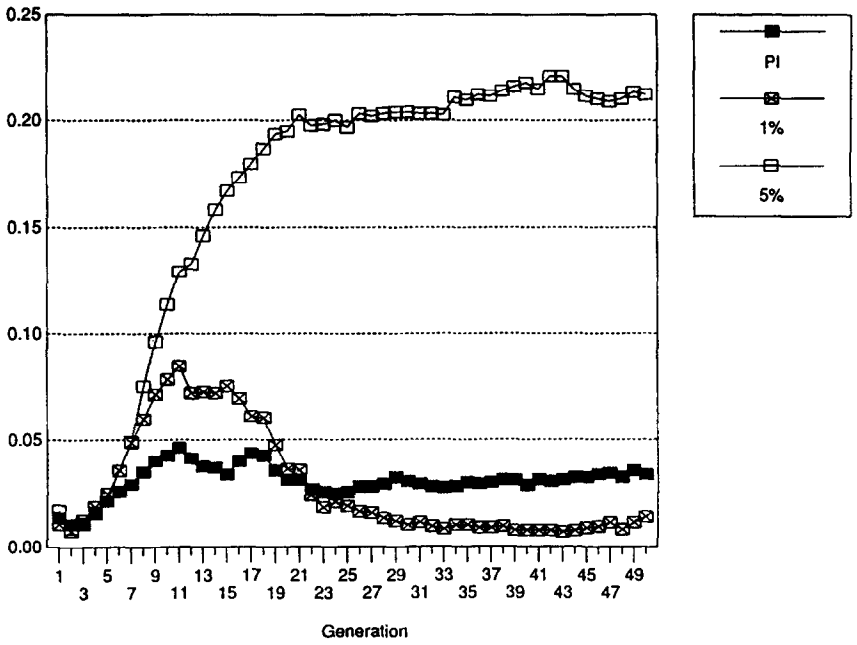


Fig. 10. Average proportion of terminal states.

forgiving of a defection by the opponent than under better information. A rise in the cooperation-reciprocity, implying a greater benefit from cooperative behavior, occurs in both the PIE and 1%NE around the same time that the payoffs under these two conditions begin to increase. Since this measure represents an average over all 30 members of the population, even a slight rise can indicate that a small number of the agents have high levels of cooperation-reciprocity. The general pattern that emerges from the reciprocity measures is that defections are not tolerated and that cooperation is reciprocated at much lower levels. Furthermore, at high noise levels these patterns become more pronounced, while at low noise levels there is some evidence, at the individual level of analysis, for them moderating relative to perfect information.

Terminal states are states that have transitions only into themselves, that is, once a terminal state is reached the automaton remains in the state for the remainder of the game. These states are of interest since they are required for any of the well-known trigger strategies (Friedman, 1971). The average number of terminal states is given in Fig. 10. After a period of rapid growth, the PIE and 1%NE values peak around the eleventh generation, decline, and then stabilize at about 0.03 and 0.01 respectively. Under the 5%NE, terminal states continue to rise until around the twentieth generation, at which time they level off to about 0.21. The 5%NE is significantly different from the other two past the fifteenth genera-

tion. The PIE and 1%NE are significantly different from each other only at the individual level of analysis. The final number of terminal states in the nosiest environment is around ten times the level in the other two. The fact that terminal states are more often used in noisier environments supports the earlier argument that rules of thumb may be important under high uncertainty. In essence, one way to deal with high noise levels is not to deal with them.

Once a terminal state is reached, the automaton's moves are fixed for the remainder of the game. Thus, not only the number of terminal states, but also their behavior is of interest. Defection quickly becomes the predominant terminal action in all of the environments. In the 5%NE almost 100% of the terminal states defect within three generations. The other two environments experience more fluctuations, with the 1%NE tending to have fewer terminal defections. [14] The high proportion of defection in the terminal states is consistent with the types of trigger strategies suggested in the theoretical literature.

The above analysis indicates that a common pattern pervades the evolution of the automata's characteristics. Initially, a period of rapid change occurs. This change quickly slows and plateaus after about ten generations–with the actual leveling off taking longer as the noise in the environment increases. Sometimes in the PIE and 1%NE a short period of readjustment occurs just prior to the plateau. A definite bifurcation appears between the 5%NE and the other two environments. There is also evidence that the impact of low noise levels is qualitatively different from more extreme levels relative to perfect information. That is, a low level of noise may make automata more cooperative and less likely to punish defections than without noise, while higher noise levels have the opposite effect.

5.3. The evolution of population characteristics

Another area for analysis is the evolution of some general population characteristics. Knowledge about the dynamics of the population in the model can suggest important hypotheses concerning the behavior of various systems. Analysis of the creation dates and survival probabilities of the agents indicates that newly created structures have a more difficult time surviving as the population becomes more evolved. Notwithstanding this observation, older structures do not appear to dominate the populations, implying a relatively dynamic environment. Unlike the individual automaton's characteristics discussed in the previous section, the general population characteristics tend to separate the PIE from the other two environments. Under the PIE, structures created in later generations have a more difficult time surviving then those created earlier. Therefore, one major effect of noise appears to be the enhancement of the survival probability of new entrants.

[14] These differences are rarely statistically significant.

Table 1
Average scores of top automata in various environments

	Environment		
	Own population (Final generation)	Randomly generated opponents	Other top performers
PI	2.85	2.56	2.17
1%NE	2.67	2.58	2.08
5%NE	1.99	2.61	1.76

5.4. *The robustness of the evolved automata*

The structures in each population coevolve in isolation from other populations. Thus, the observed results may be due to the automata adapting to their specific environment. To assess the impact of specialization, three experiments were conducted. The experiments matched the top performing automaton from each of the final populations of the forty simulations [15] against: (1) a group of randomly generated automata, (2) each other, and (3) a representative sample of the strategies submitted for Axelrod's second human tournament (Axelrod, 1984). [16] Table 1 summarizes the performance of the top performing automata in their own final generation, as well as in the first two experiments.

The first experiment took each of the top performers and matched them against twenty-nine randomly generated machines. This environment is similar to the one that the automata initially faced in the first generation. On average, the top automata did better than their opponents. Under the PIE and 1%NE the scores were about 15% higher, while they were about 17% higher in the 5%NE. All of these differences were statistically significant. In the PIE and 1%NE the ultimate payoff to the automata in the random environment was less than their payoffs in the final generation, while in the 5%NE, the payoff was higher. The likely explanation for these results is that, in the lower noise environments, there exist population specific benefits arising from establishing cooperation that could not be realized playing random automata. In the 5%NE, evolved strategies are probably able to exploit poor players (via, for example, terminal defection), and thus can do well against random opponents.

The previous experiment indicates that fundamental changes in the evolved automata allow them to excel in a random environment. To subject the best automata to a more challenging world, they were matched against one another. The scores resulting from this experiment were 24%, 22%, and 12% lower than

[15] Only the first thirty simulations were used in the second experiment.

[16] Thanks to Bob Axelrod for supplying the representative eight strategies from his tournament, and to J. Michael Coble for programming assistance.

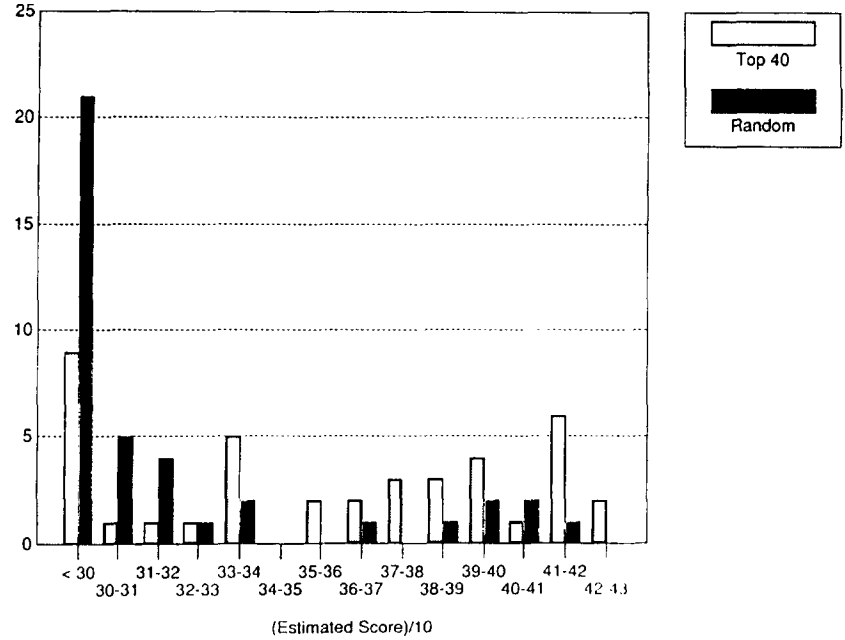


Fig. 11. Distribution of scores against Axelrod's (1984) expert opponents.

the automata's final generation scores in the PIE, 1%NE, and 5%NE respectively. However, only the first two changes were statistically different. The low scores achieved in this environment indicate that population specific effects are somewhat important, especially under better information.

An intriguing test of the top performers in a context other than a game against other automata is provided by matching them against a set of strategies submitted by human subjects for inclusion in Axelrod's second tournament (Axelrod, 1984). The set of strategies used here consisted of eight representatives out of the original 63 entries. These eight accounted for 98% of the variance in the final tournament scores. The forty top PIE [17] automata achieved an average score of 352 points in the 151 round tournament. One of the automata did as well as the top performer in the tournament, TFT, and nine of them were above the estimated median score in the tournament. A control group of forty randomly generated automata were also run against the representative strategies. Their average score was 295 points, and three of the forty scored higher than the median. The distribution of these scores is given in Fig. 11. The 19% higher score of the evolved verses the random automata was statistically significant. Given that the evolved agents had been developed

[17] Only the PIE automata were used since Axelrod's tournament did not allow for the possibility of noise.

only in environments composed of other automata, their relatively good performance in the completely novel environment of human opponents is reassuring. More importantly, however, is the fact that their performance, as well as the general empirical results, seem to be very consistent with the human tournament. Thus, the use of the methodology developed in this paper may provide a basis for benchmark experimentation in cases where costly expert tournaments are infeasible.

The automaton that tied for first place in Axelrod's tournament had many characteristics that were similar to TFT. Its cooperation-reciprocity was 0.83–close to TFT's 1.0 measure. Unlike TFT's perfect defection-reciprocity of 1.0, the automaton's value was 0.42. The automaton had a minimized size of 12 states verses the 2 states necessary for TFT. When played against some standard RPD strategies, the automaton performed very similarly to TFT, with a bit more tolerance of defections for short periods.

5.5. Extensions of the empirical analysis

A wide variety of potential extensions of the empirical analysis exists. The results imply that the level of noise in the environment is quite important. Low levels of noise tend to promote cooperative behavior while higher levels seem to disrupt it. A definite bifurcation occurs under different information levels. One unresolved question is whether the dynamics undergo a smooth transition over various noise levels. The impact of alternative informational configurations, for example, asymmetric noise levels, could also be explored. Questions about the effect of different payoff structures on the likelihood of cooperation developing are also amenable to experimentation. The general form of the RPD allows a lot of flexibility in the actual payoff values. While the experimental outcomes of certain standard payoff matrices are well known, it is possible that other configurations may alter the qualitative results.

The ability to replicate, probe, and recover the computational system also allows a variety of other hypotheses to be interactively tested. Directly incorporating a cost to the size of each automaton would yield a better understanding of the role of procedural complexity in these games. Finally, it is possible to run two separate coevolving populations playing against each other. Results indicate that a single population tends to rapidly lose heterogeneity. By running two simultaneous populations the consequences of this loss would not be as severe, and it is likely that better strategies will evolve.

6. Conclusions

The elements of this analysis combine to form a methodology well suited for the analysis of adaptive strategic choice in games. An obvious extension is the

analysis of the evolution of strategies across large classes of games, for example, the Rapoport and Guyer (1966) 2×2 game taxonomy. Current work searching for generic patterns of adaptive strategic behavior in this class of games is now underway.

While the above model focuses on the evolution of automata, a much more general model is suggested. Systems of artificial adaptive agents create inherently dynamic models of adaptive learning phenomenon that can be easily implemented and analyzed (Holland and Miller, 1991). Thus, such models may be an appropriate choice for a wide variety of economic analysis.

7. For further reading

Binmore and Dasgupta (1986).

Acknowledgements

This paper has benefited from discussions with Ken Arrow, Brian Arthur, Bob Axelrod, Ken Binmore, John Holland, Stuart Kauffman, two anonymous referees, and seminar participants at Carnegie Mellon University and the Santa Fe Institute. Bob Axelrod kindly provided computer programs from his 1984 tournament. A special thanks to Ted Bergstrom for his extensive comments. An equipment grant from Sun Microsystems, and the financial support of the NIH and the Santa Fe Institute are gratefully acknowledged.

References

Abreu, D. and Rubinstein, A., 1988, The structure of Nash equilibrium in repeated games with finite automata, Econometrica 56 (November), 1259–81.

Arbib, M.A., 1968, Algebraic theory of machines, languages and semigroups, New York, Academic Press.

Aumann, R.J., 1981, Survey of repeated games, In: R.J. Aumann et al., eds., Essays in Game Theory and Mathematical Economics in Honor of Oscar Morgenstern, Bibliographisches Institut, Zurich, Mannheim, 11–42.

Axelrod, R., 1984, The evolution of cooperation, New York, Basic Books.

Axelrod, R., 1987, The evolution of strategies in the iterated prisoner's dilemma, In: Lawrence Davis, ed., Genetic Algorithms and Simulated Annealing, Los Altos, California, Morgan Kaufmann.

Axelrod, R. and Dion, D., 1988, The further evolution of cooperation, Science 242, Dec., 1385–1390.

Bethke, A.D., 1981, Genetic algorithms as function optimizers, Ph.D. Dissertation, The University of Michigan, Ann Arbor.

Binmore, K.G., 1987a, Modeling rational players I, Economics and Philosophy 3, 179–214.
Binmore, K.G., 1987b, Modeling rational players II, Economics and Philosophy 4, 9–55.
Binmore, K.G. and Dasgupta, P., 1986, Game theory: A survey, In: K.G. Binmore and P. Dasgupta, eds., Economic Organizations as Games, Oxford, United Kingdom, Basil Blackwell.
Booker, L.B., Goldberg, D.E. and Holland, J.H., 1987, Classifier systems and genetic algorithms, Technical Report 8, Cognitive Science and Machine Intelligence Laboratory, The University of Michigan.
Day, R.H., 1975, Adaptive processes and economic theory, In: R.H. Day and T. Groves, eds., Adaptive Economic Models, New York, Academic Press.
DeJong, K.A., 1975, Analysis of the behavior of a class of genetic adaptive systems, Ph.D. Dissertation, The University of Michigan, Ann Arbor.
Fogel, L.J., Owens, A.J. and Walsh, M.J., 1966, Artificial intelligence through simulated evolution, New York, John Wiley and Sons.
Frantz, D.R., 1972, Non-linearities in genetic adaptive search, Ph.D. Dissertation, The University of Michigan, Ann Arbor.
Freund, J.E. and Walpole, R.E., 1980, Mathematical Statistics, third edition, Englewood Cliffs, New Jersey, Prentice-Hall.
Friedman, J., 1971, A non-cooperative equilibrium for supergames, Review of Economic Studies 38, 1–12.
Fudenberg, D. and Maskin, E., 1986, The folk theorem in repeated games with discounting or with incomplete information, Econometrica 54 (May), 533–554.
Goldberg, D.E., 1989, Genetic algorithms in search, optimization and machine learning, New York, Addison-Wesley.
Harrison, M.A., 1965, Introduction to switching and automata theory, New York, McGraw-Hill.
Heiner, R.A., 1983, The origin of predictable behavior, The American Economic Review 83 (September), 560–95.
Holland, J.H., 1975, Adaptation in natural and artificial systems, Ann Arbor, Michigan, The University of Michigan Press.
Holland, J.H. and Miller, J.H., 1991, Artificial adaptive agents in economic theory, American Economic Review, Papers and Proceedings 81 (May), 365–70.
Hopcroft, J.E. and Ullman, J.D., 1979, Introduction to automata theory, languages and computation, Reading, Massachusetts, Addison-Wesley.
Martin, N., 1973, Convergence properties of a class of probabilistic adaptive schemes called sequential reproductive plans, Ph.D. Dissertation, The University of Michigan, Ann Arbor.
Maynard Smith, J., 1982, Evolution and the theory of games, Cambridge, United Kingdom, Cambridge University Press.
Miller, J.H., 1986, A genetic model of adaptive economic behavior, Working Paper, The University of Michigan (March 1986).
Miller, J.H., 1988, The evolution of automata in the repeated prisoner's dilemma, Essay in Ph.D. Dissertation, University of Michigan, Ann Arbor.
Nelson, R. and Winter, S., 1982, An evolutionary theory of economic change, Cambridge, Harvard University Press.
Press, W.H., Flannery, B.P., Teukolsky, S.A. and Vetterling, W.T., 1986, Numerical recipes: The art of scientific computing, New York, Cambridge University Press.
Rapoport, A. and Guyer, M., 1966, A taxonomy of 2×2 games, General Systems 11, 203–14.
Rubinstein, A., 1986, Finite automata play the repeated prisoner's dilemma, Journal of Economic Theory 39 (June), 83–96.
Shubik, M., 1982, Game theory in the social sciences: Concepts and solutions, Cambridge, Massachusetts, MIT press.
Simon, H.A., 1955, A behavioral model of rational choice, Quarterly Journal of Economics 69, 99–118.

Simon, H.A., 1959, Theories of decision making in economics and behavioral science, American Economic Review 49, 253–83.

Tucker, A.W., 1950, A Two-person dilemma, Mimeo, Stanford University.

Vose, M.D., 1991, Generalizing the notion of schema in genetic algorithms, Artificial Intelligence 50 (August), 385–96.