The Coevolution of Automata in the Repeated Prisoner's Dilemma

John H. Miller

SFI WORKING PAPER: 1989--003

SANTA FE INSTITUTE

THE COEVOLUTION OF AUTOMATA IN THE REPEATED PRISONER'S DILEMMA

John H. Miller

Santa Fe Institute and

Carnegie-Mellon University

October 15, 1987

Revised July 10, 1989

Abstract

The coevolution of strategies in the repeated Prisoner's Dilemma game is studied under both perfect and imperfect information conditions. Players are required to submit strategies in the form of finite automata in order to participate in the game. By applying recent developments from the study of genetic algorithms in computer science, an explicit environment and selection process are derived. Using this framework, the effect of imperfect information on the development of cooperation and strategic choice is studied. The strategies that emerge are classified and their performances are analyzed. The results of the analysis indicate that information conditions lead to significant differences arising between the evolving strategic environments. Furthermore, they suggest that the general methodology may have much wider applicability.

Acknowledgments

This paper is derived from my Ph.D. dissertation. It has benefited from comments by Brian Arthur, Bob Axelrod, Ken Binmore, John Holland, Stuart Kauffman, and seminar participants at Carnegie-Mellon University and the Santa Fe Institute. A special thanks to Ted Bergstrom for his extensive comments. The financial support of NIH and the Santa Fe Institute's Economic Research Program are gratefully acknowledged. The latter is funded by grants from Citicorp/Citibank and the Russell Sage Foundation and by grants to SFI from the MacArthur Foundation, the National Science Foundation (PHY-8714918) and the U.S. Department of Energy (ER-FG05-88ER25054).

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, most of them fail to incorporate notions of bounded rationality and implementation costs. One new technique for directly confronting these issues is the theoretical study of meta-agents who select a strategy from a well-defined set of available strategies in order to participate in the game. This paper explores the coevolution of strategies in the context of the repeated Prisoner's Dilemma (RPD) game, when the meta-agent's choice is modeled by an evolutionary process.

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* which 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 which have emerged from the recent work of Binmore (1986a). Binmore argues that descriptive concepts of equilibrium may be more important than prescriptive ones. However, current descriptive constructs, for example, the idea of evolutionary stable strategies (Maynard Smith, 1982), 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 game in a tractable manner.

This work assumes that a player's program can be represented by a finite automaton (a Moore machine). The idea of selecting a new program based on the results of previous programs is operationalized through the use of recent results from the study of genetic algorithms in computer science. 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 information. 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 behavior are combined in a general methodological framework. While the major focus is a game theoretic application of this framework, generalizations which capture other social science phenomenon exist, and are discussed in the last section. Empirically, this paper introduces techniques which allow useful experimentation to be conducted on a wide variety of games and other complex systems. Through the use of these methods, problems which were previously inaccessible can now be analyzed.

2. Background

The potential of automata theory for the analysis of games was first suggested in the economics literature by Aumann (1981). Rubinstein (1986) studied an RPD in which both players were required to submit strategies in the form of a Moore machines. 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. Rubinstein did not, however, consider the cost of computing these optimal rules. The fact that no bounds are assumed on the abilities of players to derive the best procedural rules is disturbing and the approach utilized in this study circumvents this inconsistency. Nevertheless, Rubinstein's work is an important formalization of the bounded rationality hypothesis. Rubinstein introduced a dynamic definition of equilibrium in such machine games and found that, in equilibrium, machines will have cycles which never use an internal state more than once during any repetition, and that opposing machines will coordinate their actions. This latter result implies that the potential set of equilibrium outcomes from the game is sharply reduced. In a later paper, Abreu and Rubinstein (1988) weakened the definition of equilibrium and found that a similar class of results holds.

In this paper, the meta-agent's choice of a strategy automaton is modeled through the use of a genetic algorithm (Holland, 1975). Fogel et al. (1966) and Axelrod (1987) have presented related applications of the genetic algorithm. Fogel et al. evolved finite automata which attempted to predict a periodic sequence. Besides the obviously different task, their adaptive plan lacked many of the important features which produce a powerful genetic algorithm (for example, crossover). Axelrod used a genetic algorithm to evolve RPD strategies which based their moves on the game's past three-move history. There are a number of major differences between his work and the one reported here. First, the environment in this work is allowed to vary continuously as the 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 which 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.

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, 1987, 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 for two reasons: because of its wide applicability, and the potential for direct comparisons of the new methodology 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). A typical set of payoffs is presented in Figure 3.1. Given these payoffs, it is easily shown that mutual defection is the only Nash equilibrium (it is also a dominant 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 (Fudenberg and Maskin, 1986), 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

[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.

$$
\begin{array}{c} \\ C_1 \\ D_1 \end{array}
\begin{array}{c} \begin{array}{cc} C_2 & D_2 \end{array} \\ \begin{pmatrix} 3,3 & 0,5 \\ 5,0 & 1,1 \end{pmatrix} \end{array}
$$

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

Figure 3.1. The Basic Prisoner's Dilemma

which 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 mirroring the opponent's last move. The primacy of TFT was somewhat surprising, given the level of sophistication of other strategies entered in the tournament.

While a large amount of analysis exists for the RPD under conditions of perfect information, very little exists for the game under imperfect information. A variety of concepts of imperfect information in these models exist. The concept utilized here is that of noisy 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.[2] The introduction of noise into the system extends the basic RPD game to conditions which model some important situations, for example, price agreements among oligopolistic firms, arms treaties under uncertain verifiability conditions, etc.

Experiments similar to Axelrod's have been conducted for noisy RPD by Bergstrom and Miller (1985). Noise has a significant impact on the efficacy of strategies and the results are very different from the perfect information case. More complex strategies (for example, those based on Bayesian updating) tend to do quite well, while TFT's performance suffers.[3] 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 which discount reported defections due to the noise, may fall victim to strategies which intentionally defect hoping for either a forgiving opponent or a reporting error.

The RPD is a natural choice for inclusion in these experiments. Techniques which 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 the strategies in the game. While a variety of possibilities exist, the use of finite automata for this purpose appears promising.

[2] While this analysis will only consider symmetric noise levels, an interesting extension would be to cases of asymmetric noise. For example, suppose that defections from a price agreement are hard to detect. Then, the likelihood that a firm defecting from such an agreement is reported as cooperating may be far higher than the possibility of a cooperating firm being misreported as defecting.

[3] Intuition about why TFT does poorly can be gained by considering two TFTs playing one another. Without noise, the two strategies will cooperate for all rounds, while with noise, they can easily get out of synchronization resulting in long sequences of $C_1D_2, D_1C_2, C_1D_2, D_1C_2, \ldots$, and thus low payoffs.

3.2 *Finite Automata*

Finite automata mathematically model a system which 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 course, of most interest to this work; however, the vast modeling potential of these techniques hints at a far richer set of potential applications of 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.[4] 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 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 which gives the next internal state that the machine will enter 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 Figure 3.2 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 towards the next state of the machine. The starting state is indicated by the arc labeled S. For example, the first machine in Figure 3.2 always cooperates, regardless of the opponent's actions. The second machine is an automaton which 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, will ensue. The third machine is a trigger strategy, which 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 which will change the automaton's internal state, and therefore it will defect for the remainder of the game. The fourth automaton describes a strategy which 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 which have been of central importance to various earlier studies.

[4] Formally, a Moore machine is described by a four-tuple $< Q, q_0, \lambda, \delta >$, where Q is a finite set of internal states, $q_0 \in Q$ designates the starting state, $\lambda : Q \rightarrow S_i \in \{C, D\}$ 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, $\delta : Q \times S_{\sim i} \rightarrow Q$ ($S_{\sim i} \in \{C, D\}$ is the opponent's reported move last period).

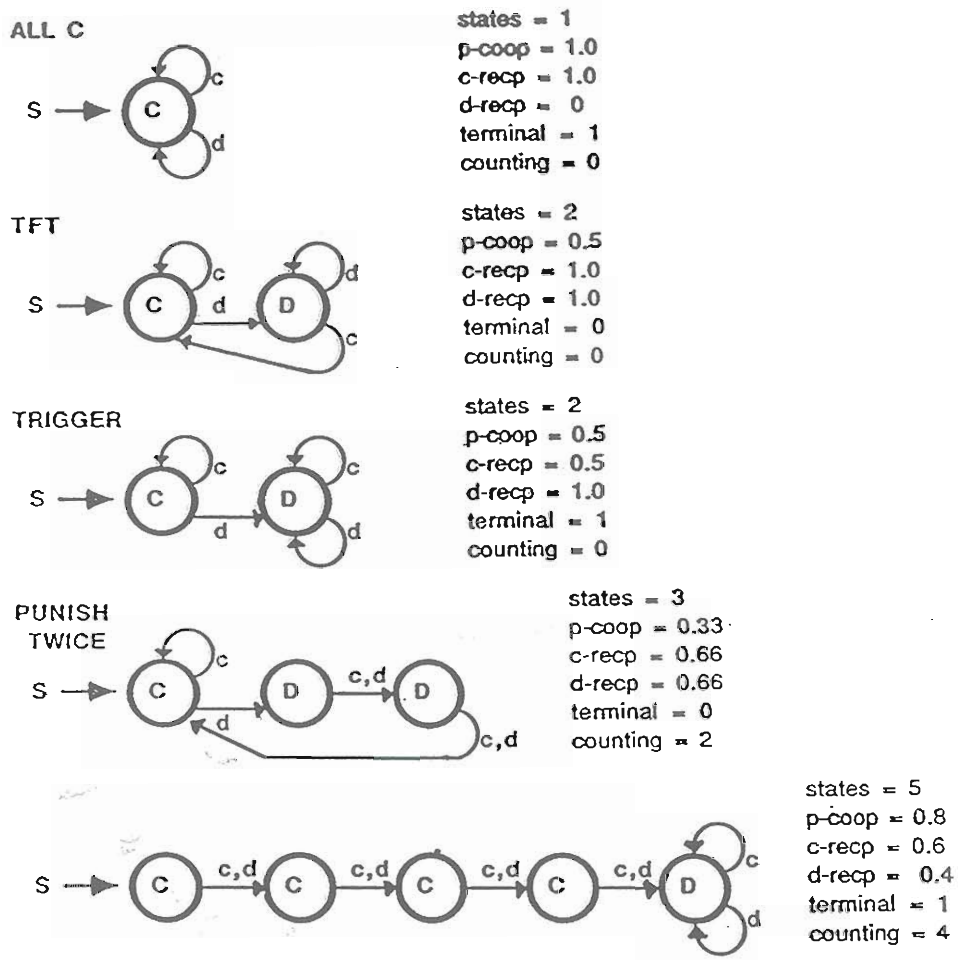

Figure 3.2. Some Possible Automata

An important characteristic of automata is that their memory of the game is embedded in their internal states. The current internal state summarizes all of the relevant history of the game for the automaton. A strategy which 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 which 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 which 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 Figure

3.3). The first four bits provide the starting state of the automaton.[5] 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 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} different possible structures.[6]

A sample automaton:

$$\underbrace{0010110010101100100001000100001\cdots\cdots\cdots0101}_{\text{148 bits}}$$

The structure of an automaton:

$$\underbrace{\#\#\#\#}_{\substack{\text{start}\\\text{state}}}\underbrace{\#\#\#\#\#\#\#\#\#}_{\text{state 0}}\underbrace{\#\#\#\#\#\#\#\#\#}_{\text{state 1}}\cdots\cdots\cdots\underbrace{\#\#\#\#\#\#\#\#\#}_{\text{state 15}},$$

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

$$\overbrace{\underbrace{\#}\;\underbrace{\#\#\#\#}\underbrace{\#\#\#\#}}^{\text{state } i}$$

$$\begin{pmatrix}\text{move when in}\\ \text{state } i.\\ 0 = \text{cooperate}\\ 1 = \text{defect}\end{pmatrix}\begin{pmatrix}\text{transition state if}\\ \text{opponent is observed}\\ \text{to cooperate}\\ \text{BCD} \in \{0,\ldots,15\}\end{pmatrix}\begin{pmatrix}\text{transition state if}\\ \text{opponent is observed}\\ \text{to defect}\\ \text{BCD} \in \{0,\ldots,15\}\end{pmatrix}$$

Using this scheme the sample automaton at the top of the figure defines a Tit-For-Tat strategy, which uses states 1 and 2.

Figure 3.3. The Biology of Automata

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. The previous work utilizing automata offers some theoretical guidance, but lacks some important theoretical links and does not provide an empirical framework. By introducing an evolutionary process, a syntheses occurs which allows the derivation of a consistent theoretical structure as well as a tractable empirical approach.

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

[6] Although there are 2^{148} unique strings, the total number of unique automata is much less than this value. For example, there exists 16! ways to relabel the internal states of each of these strings. Even with this duplication, the number of unique automata is still very large (on the order of 10^{21}, or so).

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 algorithms were developed by Holland (1975) for optimization problems in *difficult* domains. *Difficult* domains are those which have both enormous search spaces as well as 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 behavior (Miller, 1986). Finally, the existing literature from computer science has important analytical and empirical results regarding the algorithm's, and hence the model's, behavior.

Genetic algorithms are a large class of optimization routines which share the following characteristic: a *population* of well-defined structures acts in an environment and receives payoff information on each member's performance, and from this information a new generation of structures is formed by employing a set of genetic operators on the existing structures biased by the performance measures. The genetic algorithm used here is shown in Figure 3.4. Initially, thirty binary structures are chosen at random. Each structure is then tested against the environment (which in this case is composed of the other structures) and receives a performance score. 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 (discussed later).

---

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: $\text{Prob}(i) = \hat{\mu}(i,t)/\sum_j \hat{\mu}(j,t)$.
      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).

Figure 3.4. The Adaptive Plan

The crossover and mutation operations are both important elements of the algorithm, as well as interesting ways to tractably model innovative 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 Figure 3.3 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 attaching them to all of the bits after the $c + 1$ of the second parent. The second child is formed in a similar way using the remaining portions of the two parental strings (see Figure 3.5). Mutation occurs when a bit at a random location on the string changes states.

P_a and P_b are two parents chosen to crossover at point c.

$$P_a = \oplus \oplus \oplus \oplus \oplus \underbrace{\oplus}_{c} \oplus \oplus \oplus \cdots\cdots \oplus \oplus \oplus \oplus$$

$$P_b = \odot \odot \odot \odot \odot \underbrace{\odot}_{c} \odot \odot \odot \cdots\cdots \odot \odot \odot \odot$$

After crossover, the resulting children, C_{ab} and C_{ba}, have the following structures:

$$C_{ab} = \oplus \oplus \oplus \oplus \oplus \oplus \odot \odot \odot \cdots\cdots \odot \odot \odot \odot$$

$$C_{ba} = \odot \odot \odot \odot \odot \odot \oplus \oplus \oplus \cdots\cdots \oplus \oplus \oplus \oplus$$

Figure 3.5. The Crossover Operator

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 Figure 3.6. The first parent describes a strategy which 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 linear 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 result 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 which develops optimized structures by independently manipulating important structural building blocks.

By reproducing structures based on their performances, only the better strategies are allowed to proliferate. Note that future generations are composed of better strategies and thus these new environments put increased performance requirements on future generations. Moreover, structures which are performing better than average are also being sampled more often. Under a system of pure reproduction by performance, existing structures which perform well perpetuate; however, no new structures are introduced. In order to develop new structures the crossover and mutation operators are employed.

Although the crossover operator may appear to haphazardly create new structures, in fact, it is actually able to subtly combine important *patterns* of existing structures into new structures. By explicitly manipulating the population of structures, crossover is implicitly able to recombine those parts of the existing structures which account for better performance. To prevent the adaptive plan from getting trapped by eliminating initially poor performing,

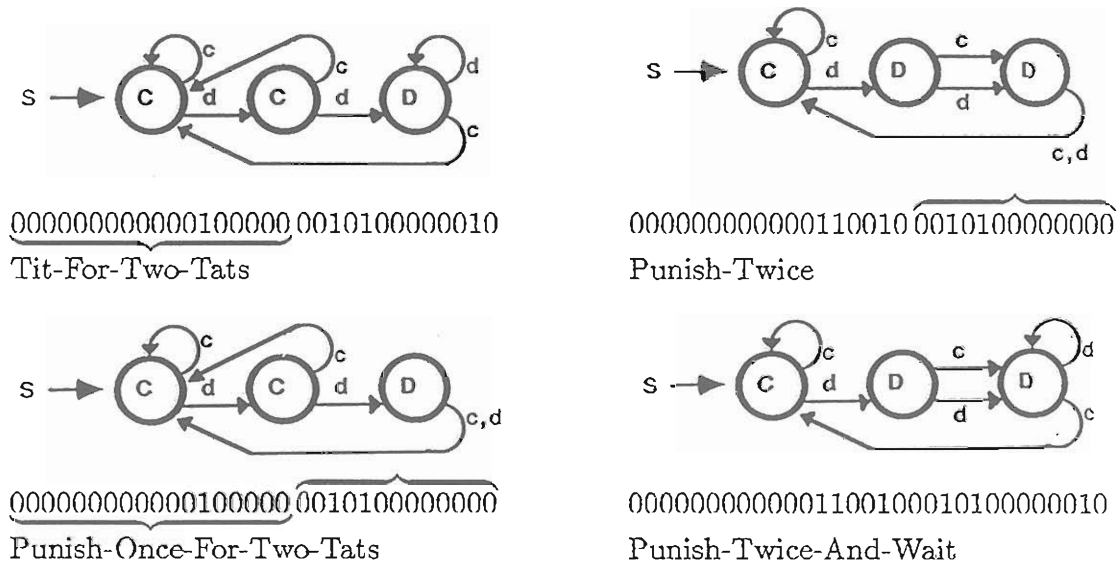

Figure 3.6. An Example of Crossover on Some Automata

but ultimately important patterns, a mutation operator is allowed. Mutation prevents the elimination of potentially valuable patterns. Its value is not in generating new structures to test, since this is equivalent to using an enumerative approach, but rather in the prevention of entrapment on false peaks. Consequently, only a small probability of mutation is required.

The incorporation of the crossover and mutation operators along with reproduction by performance, results in a powerful adaptive plan. Patterns increase or decrease based only upon 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. (These results are explored further in the Appendix.)

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. DeJong found that the algorithm exhibited rapid initial improvement, but that it usually converged towards a point near, but not at, the optimal value. Subsequent analysis showed that this was caused by the phenomenon of genetic drift, where certain important schemata were lost initially and not recovered by the population due to purely stochastic effects. DeJong's analysis also included an exploration of appropriate parameter settings. In general, these results revealed that: (1) larger population sizes produce better long term but have slower initial performance, (2) larger mutation rates prevent character loss and provide better initial but poorer long-term performance, and (3) larger crossover rates also prevent character loss but slow initial performance. DeJong's investigation showed that the version of the algorithm used here performed well across a variety of environments. Finally, Bethke (1981) extended DeJong's results. Through the use of Walsh transforms he demonstrated that those functional forms which confounded the genetic algorithm were also the forms which provided the most difficulty for other commonly used function optimization methods.

The above adaptive plan closely corresponds to the thought experiment discussed in the introduction of the paper. The idea of players resubmitting programs after looking at the scores and programs of the other players is modeled in two ways. The first, is an *imitative* component, which 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).

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 which play an RPD. The advantage of this methodology is that the very complex optimization problem of appropriate strategic choice in the RPD can be analyzed empirically. As previously discussed, the genetic algorithm is a very good optimizer in complex environments. Furthermore, its underlying mechanisms are appealing as appropriate modeling analogs. The use of automata allows a very large set of potential strategies to be easily incorporated into the algorithm. By combining the three elements, controlled empirical experiments can be conducted within a theoretically consistent framework.

4.1 Some Technical Details

Experiments were conducted under various information conditions.[7] 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.[8] Forty runs were conducted under each of the conditions to allow for stochastic variations.

The initial population in 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.[9] Payoffs were then calculated using the values in Figure 3.1. 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). At the end of each generation, the genetic algorithm discussed earlier was applied to form a new population.[10]

---

[7] 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.

[8] The respective variances are 2.95 and 14.15.

[9] Under conditions of imperfect information, five such supergames were performed between every pair to reduce the impact of stochastic variations.

[10] The final scores for each automaton were normalized by taking $\hat{x}_i = (x_i - \mu)/s + \alpha$, where x_i is the automaton's raw score, μ is the sample mean, s is the sample standard deviation, and $\alpha = 2$ is a parameter which determines the importance of relative performance. The $\hat{x}_i$ values below 0 were truncated at 0. This normalization procedure

5. Results

The results of the analysis indicate that the approach outlined above provides interesting insights into the development of cooperation and strategic choice in the RPD. Furthermore, they imply that significant differences between the perfect information (PIE), 1% noise (1%NE), and 5% noise (5%NE) environments occur. 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.[11] Unless otherwise specified, a test based on a one-tailed likelihood ratio technique[12] (see, Freund and Walpole (1980), p. 393) was used to determine whether the means were significantly different from 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.[13]

5.1 The Evolution of Payoffs

Figure 5.1 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 which 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 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%

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 $\hat{x}_i / \sum_j \hat{x}_j$ respectively). Two parents from the old population were randomly selected, where the selection probability for choosing automaton i was $\hat{x}_i / \sum_j \hat{x}_j$. A crossover 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 cth 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.

11 Where alternative approaches are used, they are noted in the text.

12 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.

13 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.

to 5% per generation), the average payoff tends to plateau by the twenty-fifth generation.[14] 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.

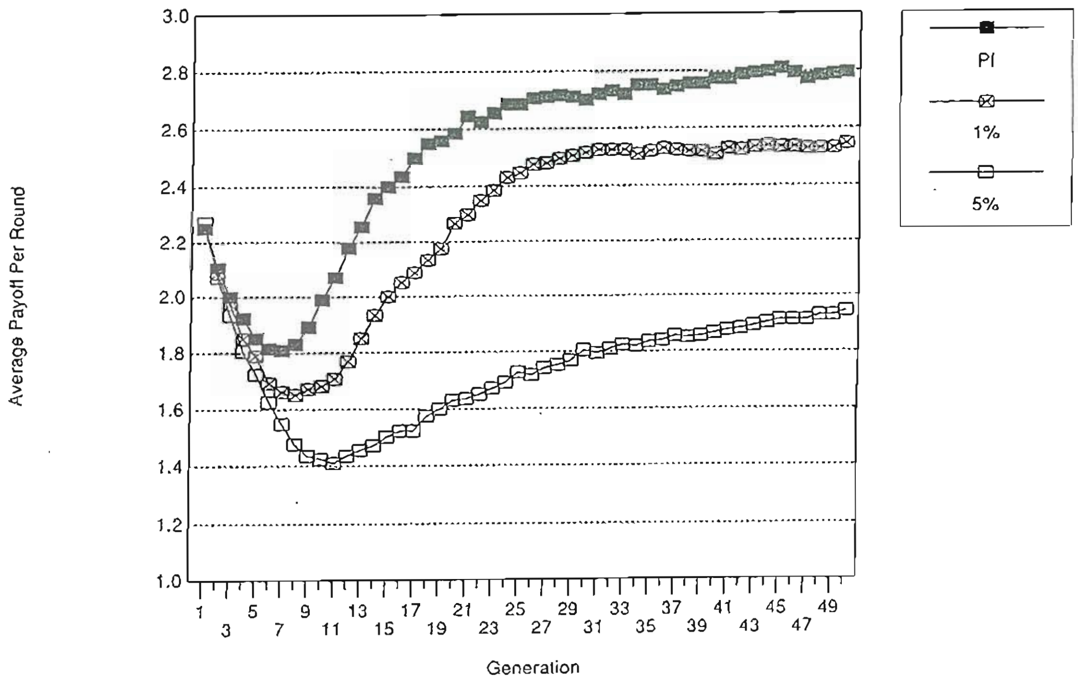

Figure 5.1. Average Payoff per Game Iteration

Figure 5.1 illustrates how cooperation can emerge in these systems. Note that in the early generations the agents tend to evolve strategies which increasingly defect. These conditions do not however persist, and at some point there is an emergence of cooperative strategies which 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 begin to proliferate in the population.

[14] 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.

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 Figure 3.2 for some examples). They may not, however, always prove adequate. For example, the last automaton in Figure 3.2 begins any game by cooperating four times and then defecting for the rest of the game, yet the descriptive measures indicate a relatively "nice" strategy.

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 which will result in a transition to the state. Therefore, even though all automata are defined for 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).[15] 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 Figure 5.2. The randomly chosen automata tend to have a high number of accessible states (about 12.25 verses the maximum limit of 16.0). The number of states in the PIE and 1%NE initially 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. 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. Note that there was no explicit attempt to minimize the automata. Thus, the decline in the size of the automata indicates that the explicit upper bound on the complexity of the machines was not binding. 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. If the number of states is a good measure of complexity, then the conclusion is that less complex strategies are used in noisier environments. While this may appear to be counterintuitive—noisier worlds are in some sense more difficult, and therefore should require more complex strategies—there do exist some theoretical models (for example, Heiner, 1983) which suggest that simple rules of thumb may be one way of coping with uncertainty. Further analysis indicates that the high noise strategies tended to rely on the use of terminal states, thus supporting this hypothesis.[16]

Notions of the actual behavior of a given machine during an RPD are derived from the actions and transitions of each accessible state. The cooperation-reciprocity (defection-reciprocity) is the proportion of accessible states which 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). Terminal states are states which have transitions only into themselves, that is, once a terminal state is

[15] 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).

[16] Not only did the 5%NE have smaller sizes, but they also tended to be less efficient in their construction. By minimizing the machines the 5%NE have about 8% fewer states verses 3% for the lower noise environments. One possible explanation for this is that their is an advantage to maintaining evolvability in more unpredictable environments.

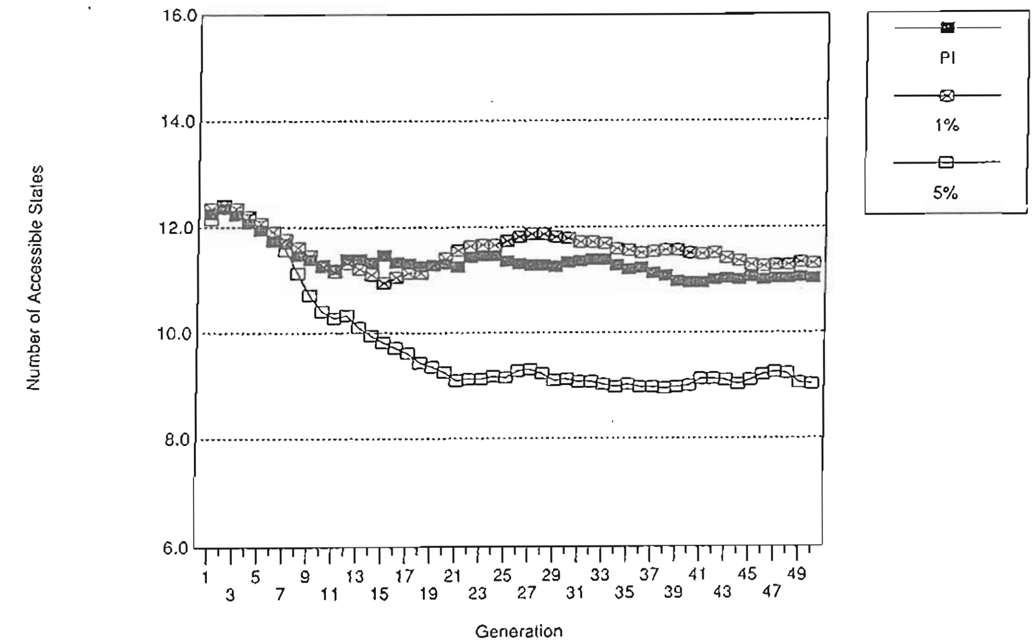

Figure 5.2. Average Accessible States per Minimized Automaton

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).

Figure 5.3 shows the proportion of the total states which return a cooperation by the opponent with a cooperation next period (cooperation-reciprocity), and Figure 5.4 gives the proportion of total states which return a defection by the opponent with a defection next time (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.[17] The differences between the PIE and 1%NE are not statistically significant at the group level of analysis, but are different past about the twentieth generation when the individual is used as the unit of analysis. The final values for the cooperation-reciprocity are .44, .49, and .36 in order of increasing environmental noise. For the defection-reciprocity the corresponding values are .70, .65, and .76. Therefore, with high noise levels, automata do not reciprocate cooperation as much, and are less 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, though never perfectly. Furthermore, at high noise levels

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

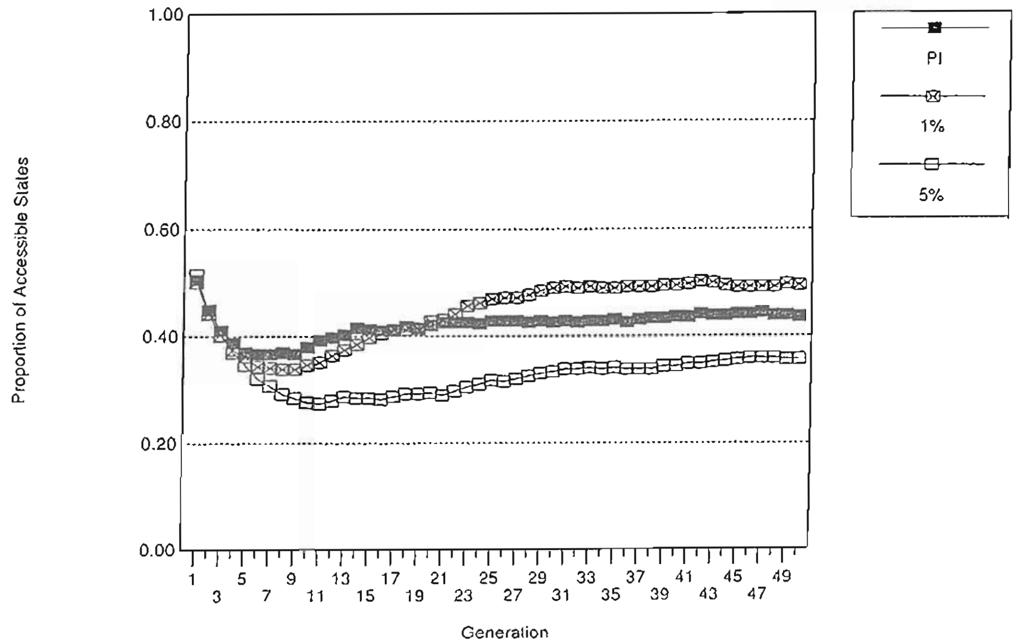

Figure 5.3. Average Cooperation Reciprocity

these patterns become more extreme while at low noise levels there is some evidence, at the individual level of analysis, for them moderating.

Important differences in the average number of terminal states arise. The average number of terminal states is given in Figure 5.5. After a period of rapid growth, the PIE and 1%NE values peak around the eleventh generation, decline, and then stabilize at about .03 and .01 respectively. Under the 5%NE, terminal states continue to rise until around the twentieth generation, at which time they level off to about .21. The 5%NE is significantly different from the other two past the fifteenth generation. 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.[18] 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 read-

[18] These differences are rarely statistically significant.

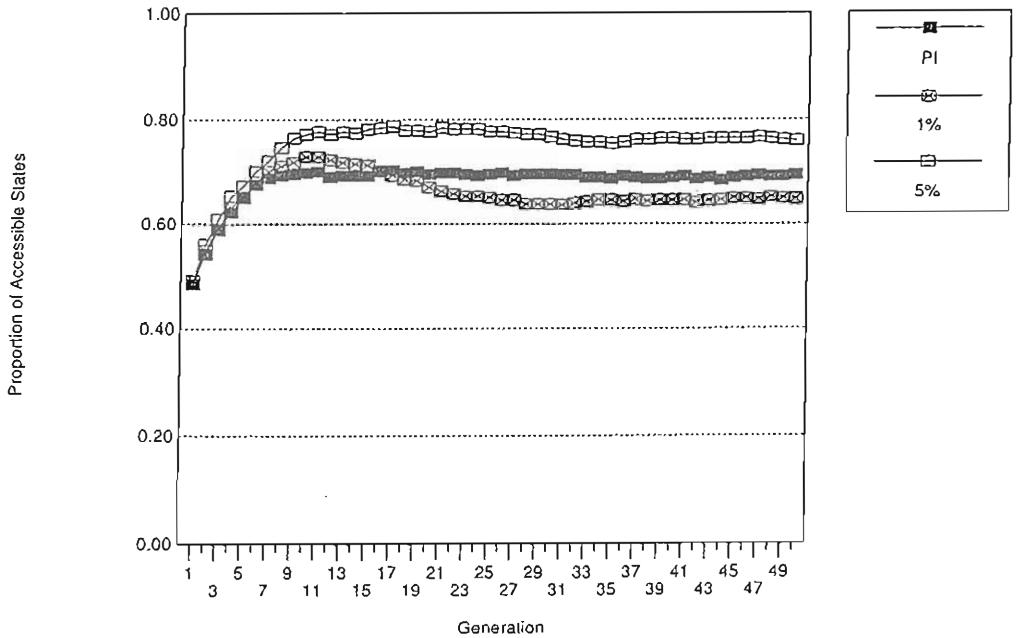

Figure 5.4. Average Defection Reciprocity

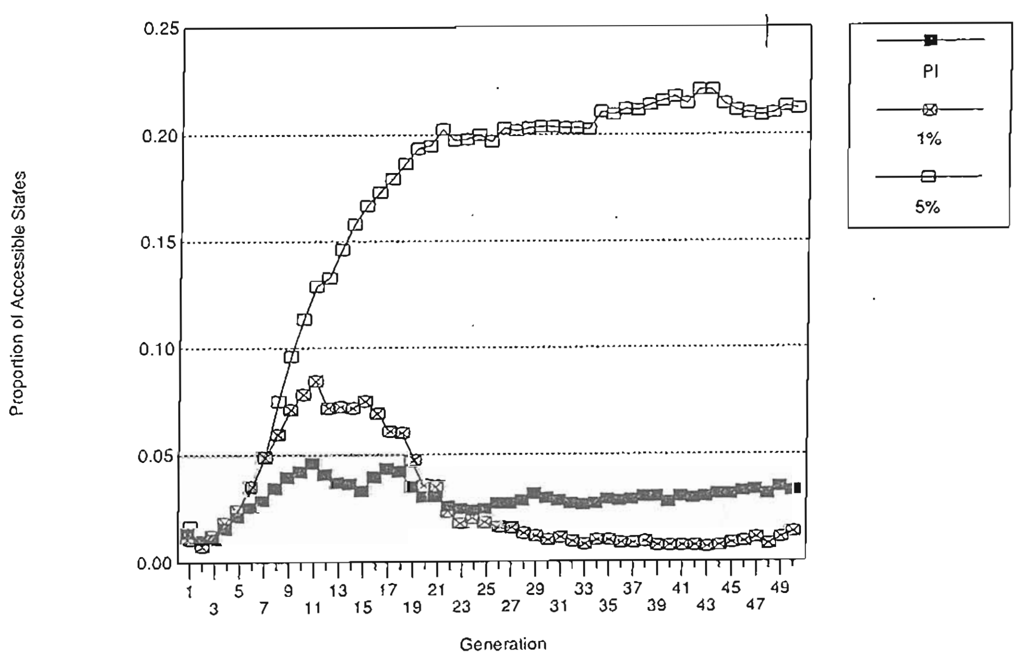

Figure 5.5. Average Proportion of Terminal States

justment 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. This section focuses on two areas: changes in the homogeneity of the population, and differences in survival probabilities over time.

After the selection of the initial structures, the only new structures which emerge are created by crossover and mutation. Given the lack of other newcomers, it is possible that the automata which arise may become highly specific to their individual populations. In order to characterize the extent of specialization, a measure of structural similarity was developed. The similarity value for any given population is the expected proportion of identical bits between two randomly selected members (bits are compared locus by locus, i.e., this is a measure of the expected Hamming distance). As illustrated in Figure 5.6, from an initial value of .50 (random generation) all of the similarity measures rise steadily to around .91 and are not significantly different from one another. Thus, the members of the population do tend to converge towards a homogeneous structure. Nevertheless, 9% of the loci do differ, an amount which cannot be explained solely by the mutation rate. The increasing population homogeneity implies that the individual structures become specialized to their own populations. The full impact of this specialization vis-à-vis new environments is explored in the next section.

While the similarity measures increase over time for any given population, they do not change across the populations. For all three environments, the similarity value for a population consisting of a member from each of the first thirty simulations remained around .50 over all fifty generations. This implies that the structures developing in each individual population are very different from those in other populations. Even though the structures are different, the earlier results indicate that the populations tend to converge on similar qualitative characteristics. It is evident that while general characteristics are important in the choice of strategies, a lot of flexibility in the actual implementation exists.

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 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.

5.4 The Robustness of the Evolved Automata

The structures in each population tend to evolve in isolation from other populations. Given the earlier evidence of specialization from the similarity measure, one wonders whether the observed results are simply due to the automata adapting to their specific environment, or whether more general changes are occurring. 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[19] against: (1) a group of randomly

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

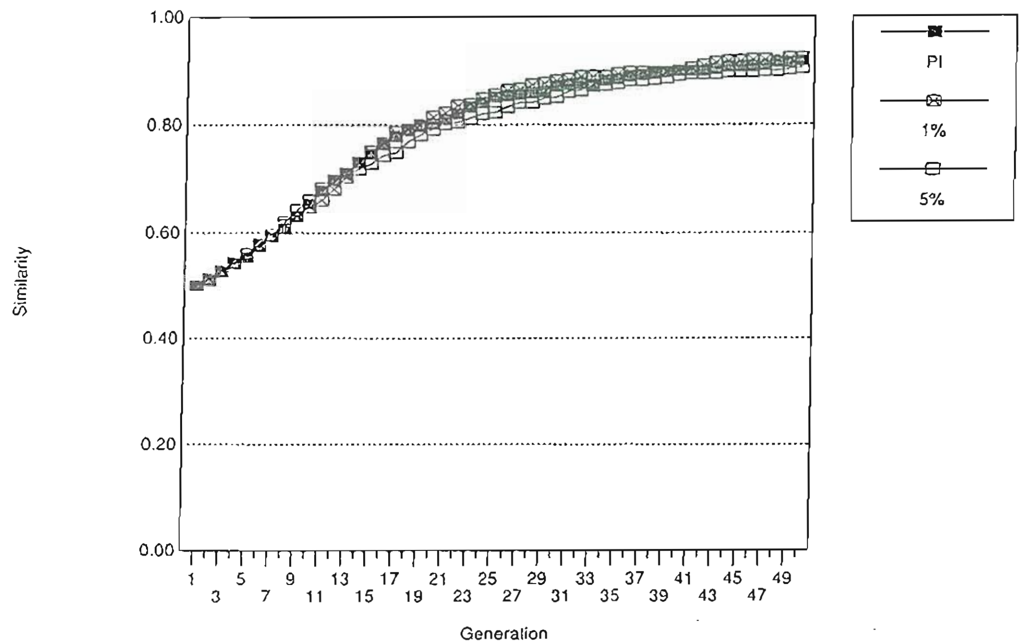

Figure 5.6. Average Similarity

Table 5.1. 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

generated automata, (2) each other, and (3) a representative sample of the strategies submitted for Axelrod's (1984) second human tournament.[20] Table 5.1 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 very similar to the one which 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. The fact that the evolved automata perform significantly better than they did originally in a unique,

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

albeit similar, environment indicates that the evolutionary experience resulted in fundamental changes which enhanced strategic performance. 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. One likely explanation for these results is that, in the lower noise environments, there exist population specific benefits arising from establishing cooperation which 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 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 (1984) second tournament. 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[21] 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 Figure 5.7. The 19% higher score of the evolved verses the random automata was statistically significant. Given that the evolved agents had only evolved against other 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 reliable experimentation in cases where costly expert tournaments are infeasible.

The automaton which tied for first place in Axelrod's tournament had many characteristics which were similar to TFT. Its cooperation-reciprocity was 0.83 which is 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 for 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, and additional experiments with varying levels may be insightful. Furthermore, 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 would also be amenable to experimentation. The general form of the RPD allows a lot of flexibility in the actual payoff

[21] Only the PIE automata were used since these tournaments did not allow the possibility of noise.

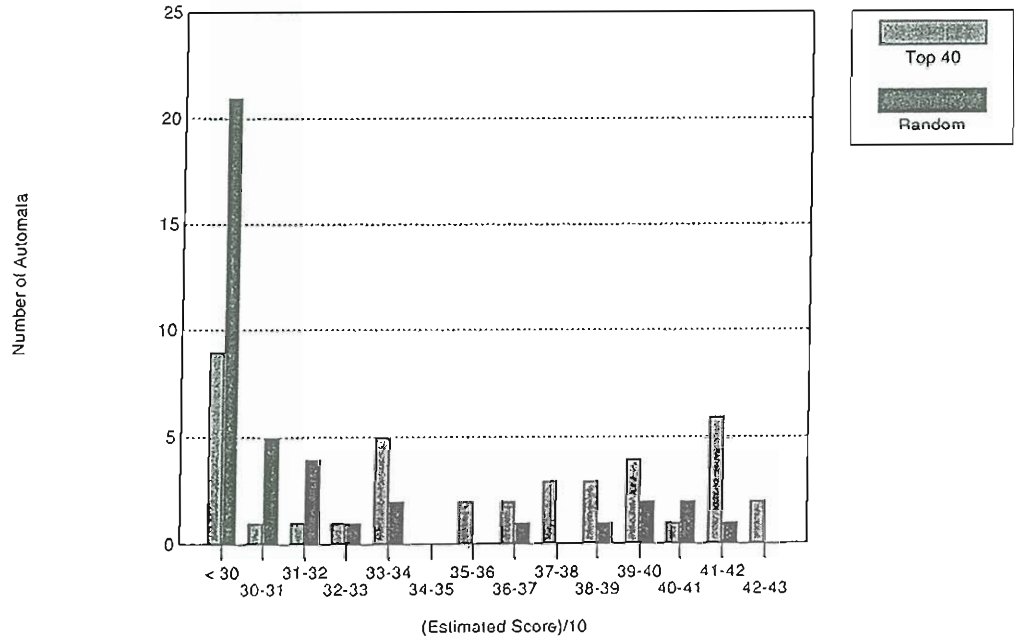

Figure 5.7. Distribution of Scores Against Expert Opponents

values. While the experimental outcomes of certain standard payoff matrices are well known, it is possible that other configurations may alter the qualitative results. Improvements and variations in the specific automaton variables would also be illuminating. Alternative measures of complexity, various weighting schemes based on the transition function, etc., could provide valuable empirical clues. Directly incorporating a cost to the size of automata would yield a better understanding of the role of complexity in these games. Experiments concerning different levels of population size, crossover and mutation rates, could also be of interest. Finally, it would be possible to run two separate populations playing against each other. The previous 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 be developed.

6. Extensions and Conclusions

6.1 Some Specific Applications of the Results

The general results of this analysis indicate that the level of noise in the system has a fundamental impact on the outcome of the RPD. Higher levels of imperfect information are associated with less cooperation and lower payoffs. The effect of noise does not appear to be continuous—definite phase transitions are evident. A variety of direct applications of the immediate analysis are apparent, for example, the importance of verifiability and effective arms treaties.

Negotiations on the limitation of strategic armaments have recently experienced major breakthroughs. Arms treaties are easily modeled as a noisy RPD (two countries can negotiate a treaty and then either comply (cooperate) with the terms of the treaty or violate (defect) it). The ability to verify compliance is directly related to the level of noise in the system. The importance of effective verification technology is highlighted by the results of this paper: the effectiveness of an arms control treaty between purely self-interested nations is directly tied to the accuracy of the verification technology. Highly accurate verification techniques will allow cooperation to be established and maintained, while less accurate ones may easily lead to a breakdown of the agreement.

6.2 Some General Extensions and Conclusions

The elements of this analysis combine to form a methodology well suited for the analysis of strategic choice in games. The basic theoretical model is easily integrated into a useful empirical framework, which allows easy experimentation on a previously inaccessible domain. While the above work explored only a small subset of the potential applications, a rich variety of insights were obtained. The possible extensions are numerous, for example, the impact of changes in information levels and symmetries, population sizes, innovation rates, payoff structures, etc., could all be investigated. Beyond simple parametric experimentation, another obvious extension is to other types of games, for example, games of coordination, multiple player games, etc. Rapoport and Guyer (1966) have identified 78 unique 2×2 player games in their taxonomy. The above methodology could easily be run on each of these games and benchmarks could be developed. These would allow interesting games for further analysis to be identified, as well as provide a database for a broad analysis of strategic behavior.

Automata model an important general class of phenomenan. Thus the above framework is easily extended to problems outside of the domain of game theory with only minor alterations. For example, automata can be used as a model of networks, thereby allowing an analysis of the evolution of organizations. Since the general methodology would be preserved, important linkages between seemingly disparate phenomenon should become apparent, and the potential for an important unification of a large class of problems exists.

While the above model focuses on the evolution of automata, a much more general model is suggested. The adaptive model introduced here has a number of advantages over more traditional models of economic processes: it is inherently dynamic, it works well in complex (difficult) environments, it provides a description of structure which allows tractable notions of innovation and imitation, and agents require little information and minimal processing ability. Along with these benefits, the model retains a strong optimization component and the potential for generating testable hypotheses. The above elements imply that this model may be an appropriate choice for a much more general model of economic behavior.

Appendix

The key to understanding the effectiveness of the genetic algorithm is the idea of schemata. Given a structure which is described by a bit string of length k, $\{0,1\}^k$, a schema can be defined on the set $\{0,1,\#\}^k$. A string is a member of a schema if each of its components match the corresponding elements of the schema which do not specify the # symbol. For example, the following eight-character binary strings: 10111111, 10110000, and 10010101, all match the schema 10#1#### (that is, they are all strings which begin with 10 and have a 1 at the 4th address). The importance of schemata is that although a single structure represents only a small proportion of all potential structures ($1/2^k$), it contains a much larger portion of all of the possible schemata ($2^k/3^k$). For example, given strings of length eight, a single string represents only about .4% of all strings, but contains almost 4% of all schemata. Therefore, an adaptive plan which only manipulates a small number of individual structures has access to a large amount of information about the important schematic building blocks.[22]

The reproduction of structures based on their payoffs encourages a continual improvement in the set of structures. Suppose that the expected number of offspring is proportional to the mean performance, that is

$$\mathrm{E}(N(S',t+1)) = \frac{\mu_t(S')}{\sum_{S\in S(t)} \mu_t(S)/M}\,, \tag{1}$$

where $N(S',t+1)$ is the number of $S' \in S(t)$ at time $t+1$, $S(t)$ is the set of structures in the population at time t, $\mu_t(S')$ is the performance of S' at time t, and M is the number of structures in $S(t)$. Equation (1) implies that those structures which do better than average will increase over time, and that those which do worse will decline. Note also, that if the environment is unchanging the average performance of the population will rise over time, putting increased survival pressure on existing structures. Moreover, structures which are performing better than average are also being sampled more often. Under a system of pure reproduction by performance, existing structures which perform well perpetuate; however, no new structures are introduced.

If structures are only reproduced according to performance, from (1) the expected number of *schemata* of type ξ represented at time $t+1$ will be

$$\mathrm{E}(N(\xi,t+1)) = \frac{\sum_{S'\in S(t):S'\in\xi} \mu_t(S')}{\sum_{S\in S(t)} \mu_t(S)/M} \qquad \forall\, \xi, t.$$

Multiplying and dividing the right-hand side by $N(\xi,t)$ yields

$$\mathrm{E}(N(\xi,t+1)) = \frac{\hat{\mu}_t^{\xi}}{\hat{\mu}_t} N(\xi,t) \qquad \forall\, \xi, t, \tag{2}$$

[22] The schemata defined above represent hyperplanes in the string space and do not cover the set of all possible building blocks. This is easily demonstrated since the set of possible building blocks (patterns) is of size 2^{2^k} which is much greater than the number of potential schemata 3^k. An example of a pattern which cannot be encompassed by the above system is one which matches an even number of bits.

where $\hat{\mu}_t^\xi$ is the average performance of ξ in the sample $S(t)$, and $\hat{\mu}_t$ is the average performance of all of the structures in $S(t)$. Equation (2) implies that with reproduction alone *schemata* in the original subpopulation, $S(1)$, which belong to structures performing better than average will increase in the population. However, as above, reproduction alone does not admit any new structures into $S(t)$, thus no new schemata are tested, and the sample of old schemata is biased towards those which existed in the original population.

In order to develop new schemata and to test old schemata in different structural contexts, the crossover operator is employed. The crossover operator creates instances of new schemata while simultaneously maintaining old ones. Holland (1975, p. 99) showed that if the two originally chosen structures differ at L positions to the left of and at R positions to the right of the crossover point, then either one of the newly formed structures will contain $2^l - 2^{l-L} - 2^{l-R} + 2^{l-(L+R)}$ new schemata (schemata not found in either of the two original structures) and $2^{l-L} + 2^{l-R} - 2^{l-(L+R)}$ old ones, where l is the total length of the string. Crossover also tends to encourage the linkage of schemata. If one looks at only the defining positions of a schema (that is, those positions which specify a specific value at a given address), then the probability of schema ξ being broken by a randomly chosen crossover point is equal to $(l(\xi) - 1)/(l - 1)$ where $l(\xi)$ is the smallest number of loci which contain all of the defining values for ξ. Since $l - 1$ is constant for all structures, schemata which are closer together have a smaller probability of being separated during crossover.

If the crossover operation is introduced into the adaptive plan, the expected number of schemata of type ξ that will be present at time $t + 1$ can be obtained by combining (2) with the lower bound[23] of the probability of crossover not breaking up the schema:

$$\mathrm{E}(N(\xi, t+1)) \geq \frac{\hat{\mu}_t^\xi}{\hat{\mu}_t}[1 - \frac{(l(\xi) - 1)}{(l-1)} P_c] N(\xi, t) \qquad \forall\, \xi, t, \tag{3}$$

where P_c is the probability of crossover. This means that a schema, ξ, will increase its representation in the population as long as

$$\hat{\mu}_t^\xi \geq \frac{1}{[1 - P_c(l(\xi) - 1)/(l-1))]} \hat{\mu}_t \qquad \forall\, \xi, t.$$

This equation implies that longer schemata must have better relative performance than shorter ones to increase in the population. Also, the change in the number of each type of schema is independent of the dynamics of the other schemata in the population.

To prevent the adaptive plan from getting trapped by eliminating initially poor performing, but ultimately important schemata, a mutation operator is allowed. The mutation operator randomly changes existing bits on the string (that is, a 0 becomes a 1 and vice versa). Mutation prevents the elimination of potentially valuable schemata. Its value is not in generating new structures to test, since this is equivalent to using an enumerative approach, but rather in the prevention of entrapment on false peaks. Consequently, only a small probability of mutation is required. Equation (3) can be modified to reflect the influence of mutation on the sampling structure of the space of schemata,

$$\mathrm{E}(N(\xi, t+1)) \geq \frac{\hat{\mu}_t^\xi}{\hat{\mu}_t}[1 - \frac{(l(\xi) - 1)}{(l-1)} P_c](1 - P_m)^{d(\xi)} N(\xi, t) \qquad \forall\, \xi, t$$

where P_m is the probability of mutation and $d(\xi)$ is the number of defining elements in ξ. All of the previous results about schema growth hold with the mutation operator.

[23] This is a lower bound since it is possible, if both parents have the appropriate values, for a schema to remain intact even if the crossover point breaks up the schema.

References

Abreu, D. and Rubinstein, A. "The Structure of Nash Equilibrium in Repeated Games with Finite Automata." *Econometrica* 56(November, 1988):1259–81.

Arbib, M. A. *Algebraic Theory of Machines, Languages, and Semigroups.* New York, New York: Academic Press, 1968.

Aumann, R. J. "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), 1981:11–42.

Axelrod, R. *The Evolution of Cooperation.* New York, New York: Basic Books, 1984.

Axelrod, R. "The Evolution of Strategies in the Iterated Prisoner's Dilemma." In Lawrence Davis, ed., *Genetic Algorithms and Simulated Annealing* (Los Altos, California: Morgan Kaufmann), 1987.

Axelrod, R., and Dion, D. "Annotated Bibliography on *The Evolution of Cooperation.*" Institute of Public Policy Studies, The University of Michigan, 1987.

Bergstrom, T. C. and Miller, J. H. Unpublished results, The University of Michigan, 1985.

Bethke, A. D. *Genetic Algorithms as Function Optimizers.* Ph.D. Dissertation, The University of Michigan, Ann Arbor, 1981.

Binmore, K. G. "Remodeled Rational Players." Working Paper, London School of Economics, 1986a.

Binmore, K. G. and Dasgupta, P. "Game Theory: A Survey." In K. G. Binmore and P. Dasgupta, eds., *Economic Organizations as Games* (Oxford, United Kingdom: Basil Blackwell), 1986b.

Booker, L. B., Goldberg, D. E., and Holland, J. H. *Classifier Systems and Genetic Algorithms.* Technical Report 8, Cognitive Science and Machine Intelligence Laboratory, The University of Michigan, 1987.

DeJong, K. A. *Analysis of the Behavior of a Class of Genetic Adaptive Systems.* Ph.D. Dissertation, The University of Michigan, Ann Arbor, 1975.

Fogel, L. J., Owens, A. J., and Walsh, M. J. *Artificial Intelligence Through Simulated Evolution.* New York, New York: John Wiley and Sons, 1966.

Frantz, D. R. *Non-linearities in Genetic Adaptive Search.* Ph.D. Dissertation, The University of Michigan, Ann Arbor, 1972.

Freund, J. E. and Walpole, R. E. *Mathematical Statistics* third edition. Englewood Cliffs, New Jersey: Prentice-Hall, 1980.

Friedman, J. "A Non-cooperative Equilibrium for Supergames." *Review of Economic Studies* 38(1971):1–12.

Fudenberg, D. and Maskin, E. "The Folk Theorem in Repeated Games with Discounting or with Incomplete Information." *Econometrica* 54(May, 1986):533–554.

Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* New York: Addison-Wesley, 1989.

Harrison, M. A. *Introduction to Switching and Automata Theory.* New York, New York: McGraw-Hill, 1965.

Heiner, R. A. "The Origin of Predictable Behavior." *The American Economic Review* 83(September, 1983):560–95.

Holland, J. H. *Adaptation in Natural and Artificial Systems.* Ann Arbor, Michigan: The University of Michigan Press, 1975.

Hopcroft, J. E. and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation.* Reading, Massachusetts: Addison-Wesley, 1979.

Martin, N. *Convergence Properties of a Class of Probabilistic Adaptive Schemes Called Sequential Reproductive Plans.* Ph.D. Dissertation, The University of Michigan, Ann Arbor, 1973.

Maynard Smith, J. *Evolution and the Theory of Games.* Cambridge, United Kingdom: Cambridge University Press, 1982.

Miller, J. H. "A Genetic Model of Adaptive Economic Behavior." Working Paper, The University of Michigan, March 1986.

Miller, J. H. "The Evolution of Automata in the Repeated Prisoner's Dilemma." Essay in Ph.D. Dissertation, University of Michigan, Ann Arbor, 1988.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. *Numerical Recipes: The Art of Scientific Computing.* New York, New York: Cambridge University Press, 1986.

Rapoport, A. and Guyer, M. "A Taxonomy of 2×2 Games." *General Systems*, 11(1966):203–14.

Rubinstein, A. "Finite Automata Play the Repeated Prisoner's Dilemma." *Journal of Economic Theory* 39(June, 1986):83–96.

Shubik, M. *Game Theory in the Social Sciences: Concepts and Solutions.* Cambridge, Massachusetts: MIT press, 1982.

Tucker, A. W. "A Two-Person Dilemma." Mimeo, Stanford University, 1950.