On Automated Discovery of Models Using Genetic Programming: Bargaining in a Three-Agent Coalitions Game

GARETT DWORMAN, STEVEN O. KIMBROUGH,
AND JAMES D. LAING

GARETT DWORMAN is a Ph.D. candidate in the Department of Operations and Information Management at the Wharton School of the University of Pennsylvania. He studies the application of cognitively motivated information systems to decision problems. Two current research projects are (1) investigations into computational rationality in bargaining situations using evolutionary computation (e.g., genetic algorithms and genetic programming), and (2) developing information retrieval systems capable of high-level pattern discovery in document collections.

STEVEN O. KIMBROUGH is an Associate Professor at the Wharton School of the University of Pennsylvania. His main research interests are in the fields of decision support and artificial intelligence, logic modeling, the computational theory of rationality, and electronic commerce. His active research areas include computational approaches to belief revision and nonmonotonic reasoning, formal languages for business communication, and knowledge-based decision support systems.

JAMES D. LAING is a member of the Decision Processes and the Information Systems faculty at the Wharton School of the University of Pennsylvania. His research interests include cooperative and noncooperative game theory, process models of nonrational behavior, statistical methods, laboratory studies, and evolutionary computation, with particular emphasis on applications in the science of multilateral negotiations.

ABSTRACT: The creation of mathematical, as well as qualitative (or rule-based), models is difficult, time-consuming, and expensive. Recent developments in evolutionary computation hold out the prospect that, for many problems of practical import, machine learning techniques can be used to discover useful models automatically. The prospects are particularly bright, we believe, for such automated discoveries in the context of game theory. This paper reports on a series of successful experiments in which we used a genetic programming regime to discover high-quality negotiation policies. The game-theoretic context in which we conducted these experiments—a three-player coalitions game with sidepayments—is considerably more complex and subtle than any reported in the previous literature on machine learning applied to game theory.

Acknowledgments: We gratefully acknowledge the Research Foundation of the University of Pennsylvania, which supported this work in part. File: JMIS07.mac. Revision: 12 September 1995. This paper is a revised and extended version of a paper presented at the HICSS 1995 conference [13].

Journal of Management Information Systems / Winter 1995–96, Vol. 12, No. 3, pp. 97–125

KEY WORDS AND PHRASES: automatic model discovery, game theory, genetic programming, machine learning.

MODELING IS DIFFICULT, TIME-CONSUMING, AND EXPENSIVE. Examining a real-world system, collecting data, and summarizing the findings in the form of a valid, robust model all require of humans a comparatively well-developed level of training, insight, intellectual acumen, and domain-specific knowledge. How nice it would be if the model formulation process were significantly automated, but surely this is beyond the realm of reasonable hope. One of our purposes in this paper is to suggest otherwise, at least for many modeling situations of interest in commerce and other applications.

Recent work in machine learning has demonstrated that evolutionary computation (including genetic algorithms and genetic programming techniques) can discover useful models for complex systems. Generalizing upon seminal research by John Holland [19, 20, 21], important extensions of this approach have been published by John Koza [24, 25]. Koza's first book on genetic programming [24] contains many examples in which genetic programs are tested against data from known mathematical functions and are able to discover (different) mathematical functions that fit the data.[1] Koza also reports impressive results in finding qualitative, or rule-based, models—for example, that direct artificial ants in finding (artificial) food, and that control robots in various tasks. In his second book, Koza [25] addresses the problem of identifying which portion of a protein is normally situated in a cellular membrane (instead of inside or outside the membrane), using only the primary structure (amino acid sequence) of the protein. This is called the transmembrane problem and it represents an important challenge in contemporary microbiology. Koza's summary of his work is worth quoting [25]:

> The best predicting program evolved by genetic programming for the set-creating version of the transmembrane problem with automatically defined functions has a slightly better error rate than four other published results. This genetically evolved program is an instance of an algorithm produced by an automated technique which is superior to that written by human investigators. [P. 16. See also ch. 16, especially table 18.11, p. 485]

Other examples of automated model discovery can certainly be found, even though this literature is quite young (see [15, 20, 23]). The field of game theory, however, holds perhaps the most significant results to date, as well as some of the best opportunities for automated discovery of models. Among the most impressive findings along these lines are results that have been reported in connection with the problem of finding good strategies for playing Repeated Prisoner's Dilemma (RPD) games. It is, for present purposes, well worth recounting those results.

The Prisoner's Dilemma is among the most intriguing and well-studied games in all of social science. The game is (1) simple, (2) a plausible model for very many situations, ranging from nuclear confrontation to commercial markets, and (3) an enduring challenge to the theory of games. In the basic Prisoner's Dilemma game, two

players face each other. Each may cooperate (*C*) or defect (*D*), and neither may communicate with the other. A typical reward structure works as follows. If player 1 cooperates and player 2 defects, 1 gets a payoff of 0 and 2 gets 5. Here we say: (*C*, *D*): (0, 5). The other possibilities are: (*C*, *C*): (3, 3); (*D*, *D*): (1, 1); and (*D*, *C*): (5, 0). The payoffs are denominated—for example, in dollars—so that more is better. Other payoff structures are possible, but it is essential to the problem that when both players cooperate, both do better than when both defect, but that each player has an incentive to defect, given any fixed choice by the opponent.

Prisoner's Dilemma has a certain tragic element to it: if the players could make an enforceable bargain to cooperate, both would be better off than if both defected; yet in the absence of such a deal, both have a compelling incentive to defect, producing a worse outcome for all involved.[2] Suppose, however, that our prisoners play the game more than once. Might they, by their behavior in repeated plays, effectively communicate and arrive at a jointly beneficial bargain? Game theorists do not agree about which strategy is optimal in this repeated contest. Aiming to develop an empirical theory for the Repeated Prisoner's Dilemma (RPD), Robert Axelrod [6] held two contests in which participants submitted computer programs for playing RPD. The first contest attracted fourteen entrants. The winning policy, by far, was TIT FOR TAT: cooperate on the first game with an opponent; on all subsequent rounds with that opponent, do (*C* or *D*) what the opponent did (*C* or *D*) in the previous round. Axelrod publicized the results of this contest in the game theory community and held a second contest. This time sixty-two policies were submitted and again TIT FOR TAT won. Significantly for present purposes, only one person submitted TIT FOR TAT in the second contest—and that was the same person (Anatol Rapoport) who submitted it in the first.

TIT FOR TAT is evidently a policy for RPD with much to recommend it. No person was able to think up a better one, and only one person submitted it for these contests. Axelrod, however, reports that a genetic algorithm was able to beat TIT FOR TAT. In a subsequent study, Axelrod [7] applied a genetic algorithm in an environment consisting of eight entries from the second tournament. In that tournament "98 percent of the variance in the tournament scores is explained by knowing a rule's performance with these eight representatives" [7, p. 33]. Here is Axelrod's summary of the results:

> The results are quite remarkable: from a strictly random start, the genetic algorithm evolved populations whose median member was just as successful as the best rule in the tournament, TIT FOR TAT. While most of the runs evolve populations whose rules are very similar to TIT FOR TAT, in eleven of the forty runs, the median rule actually does substantially better than TIT FOR TAT. In these eleven runs, the populations evolved strategies that manage to exploit one of the eight representatives at the cost of achieving somewhat less cooperation with two others. But the net effect is a gain in effectiveness. This is a remarkable achievement. [7, p. 36f]

We agree that this is indeed a remarkable achievement. A computer program was able to discover strategies superior to those discovered by humans in this tournament of one of the most important and well-studied games in social science. This work, along with other investigations in evolutionary computation (e.g., [18, 23, 30]),

suggests the hypothesis that boundedly rational (artificial) agents, operating with fairly elementary computational mechanisms, can adapt within challenging decision environments to develop effective strategies (internally represented as mathematical or qualitative models) that approximate optimal behavior. The results for Prisoner's Dilemma, among other results, hold out the promise that such boundedly rational artificial agents may actually make original discoveries of models (or strategies) for a wide variety of important problems.

We are conducting a series of investigations whose primary objective is to demonstrate that boundedly rational agents, operating with fairly elementary computational mechanisms, can adapt to achieve approximately optimal strategies for bargaining with other agents in complex and dynamic environments of multilateral negotiations that humans find challenging. We have achieved considerable success in these investigations through pilot studies during the last two years, demonstrating that these artificial agents can generate effective strategies that approximate optimal solutions, and can do so under rather difficult circumstances. Moreover, we have found that the behavior of these artificial agents rivals that achieved by humans in laboratory studies. Our principal goal in this paper is to report on these results.[3]

Background

Research Perspectives on Limits to Human Rationality in Bargaining

RESULTS FROM A RAPIDLY GROWING COLLECTION OF LABORATORY STUDIES indicate various ways in which humans depart in multilateral negotiations from the perfect rationality envisioned in the theory of games (e.g., [5, 8]). We interpret these results to indicate the following general scenario (see [2]). People approach somewhat novel multiperson conflict situations, not *de nova*, but from perspectives of tentative rules of thumb developed from their previous experience. They cope with this new environment by keying on prominent features of the situation and employing simple heuristics to form an initial, but tentative, assessment of the bargaining problem. As suggested by behavioral decision theory (see [10, 36]), this "first-cut" analysis produces the starting points ("anchors") from which subsequent adjustments are made, whether via cognitive or adaptive responses, to the competitive process of negotiations. This adjustment process is vulnerable to vagaries in the coevolution of the parties as they respond to one another. It is only with considerable experience in the situation and through intensive negotiations that players begin to approach outcomes that approximate a game-theoretic equilibrium. Even then, people have difficulty articulating a well-defined, comprehensive strategy. Their behavior seems to be based on rules of thumb that represent, at best, mere fragments of the type of complete and optimal strategies identified in the theory of games for negotiating coalition agreements (see [27, 33]). Players apply these strategy fragments myopically, and have great difficulty anticipating the other players' moves. Thus, although there is some learning with intensive experience in the situation, much of the movement toward the

game-theoretic solution appears to result from adaptive, rather than cognitive, processes.

In response to these results from laboratory studies, recent progress in game-theoretic reasoning injects into the analysis some elements of nonrational behavior. For one example, although he did so to refine Nash's concept of strategic equilibrium [31], Selten [32] introduced small probabilities that each player might make a mistake ("tremble") when implementing (but not computing) a strategy. Selten assumes that these "tremble" probabilities are common knowledge among the players, and that each player takes them into account in computing an optimal strategy by solving a thereby constrained maximization problem. For another example, to account for the fact that real people can cooperate in the finitely repeated Prisoner's Dilemma, Kreps, Milgrom, Roberts, and Wilson [26] introduced the possibility that one player believes that another player is, with some small probability, a "crazy type" who is programmed always to play a particular strategy, such as TIT FOR TAT. The rational player takes into account the probability of a "crazy type" in computing an optimal response, and may conclude that it pays to emulate the "crazy type," at least for a while. Paradoxically, such injections of a little nonrationality into the analysis greatly complicate the decision problem facing any player who is computing a best response. Such work testifies to the increasingly sophisticated body of rigorous ideas we call game theory that continues to yield significant insights into the structure of mixed-motive situations. Yet such attempts to incorporate elements of nonrationality into game theory just nibble at the edges of the basic problem, because they do not generate a plausible causal explanation of how finite agents, such as humans, cope with the complexities of mixed-motive situations.

Evolutionary Dynamics, Finite Automata, and Genetic Algorithms

A fundamentally more direct approach is to assume that boundedly rational players address complex game situations, not by solving the game *de nova*, but rather by employing fairly simple rules of thumb that help to govern behavior but are too incomplete to constitute a well-formulated strategy. Aumann [5] calls this "rule rationality," in contrast to "act rationality." From this perspective, the rules may evolve with experience over a sequence of games, and help to determine the starting point of deliberations within any bargaining game. The process of negotiations from this initial stage within the play of the game is guided by these rules and the dynamic processes of competition. Thus, in contrast to previous game theory's almost exclusive focus on equilibrium and scant attention to the processes through which equilibrium is reached in rational play, we think the time is ripe to investigate the adaptive coping processes through which boundedly rational agents adjust to complex, dynamic, and reactive environments.

Biological Models

Some research in animal behavior (e.g., [16]), inspired by the seminal work of the biologist Maynard Smith [29], is based upon such a boundedly rational process model.

This approach assumes that each animal is born genetically endowed with specific pure strategies that govern its actions. If this animal's strategies are effective in coping with the current environment, then they increase the individual's fitness and thus, through Darwinian selection, are more likely to be inherited by members of the next generation. If the replicator dynamics across generations reach an equilibrium, the vector of population proportions of animals who play various pure strategies constitutes a mixed-strategy (Nash) equilibrium in the population as a whole. This approach can be useful in understanding not only the behavior of animals, but also that of markets and formal organizations [9].

Finite Automata and Genetic Algorithms

A related approach is to represent alternative, complete strategies as a set of finite-state automata that compete with each other in a tournament of, say, repeated two-agent, binary choice [*C*(cooperate) versus *D*(defect)] Prisoner's Dilemma games (e.g., [1, 18, 30]). The strategy TIT FOR TAT, for example, can be represented formally by a finite-state automaton (Moore machine) that begins in state 1 (play *C*) and remains there if the opponent plays *C*. Otherwise it moves to state 2 (play *D*) and continues in this state if the opponent plays *D*, but returns to state 1 if the opponent plays *C*. For example, Miller [30] begins his analysis with a set of randomly constituted strategies, represented as Moore machines. These structures play a tournament of repeated Prisoner's Dilemma games with each other. The strength (biological fitness) of each structure increments or decrements in accordance with the payoff this strategy wins in each game it plays. At the end of the tournament, the next generation of structures is created by probabilistically retaining the stronger structures, and creating new structures (offspring) as follows: Select, with probabilities in proportion to their strengths, pairs of old structures for mating. Each of the couples produces two offspring who, after being exposed to the genetic operations of crossover and mutation [15, 19, 20], constitute new structures that join the next generation. Then repeat this process through a series of generations.

Miller's results from various computer simulations indicate that, after an initial adjustment period from the random start, the frequency of cooperation and the average payoffs realized from the repeated game encounters increase across generations until a plateau is reached. With experience, the best-performing structures tend to become less complex. Moreover, the top performers that emerge from Miller's computer runs perform credibly against strategies submitted by game theorists for Axelrod's [6] second tournament of Repeated Prisoner's Dilemma games.

Two Problems

These results indicate that adaptive processes can generate relatively simple but reasonably successful strategies for dealing with difficult environments. Yet studies such as Miller's (also see [3]), however promising and provocative, have certain limitations. The first limitation, which we call the *complexity* problem, is that their

artificial agents play in fairly simple environments. How will they perform in more complex environments, especially environments with a richer variety of (perhaps noisy) relevant information and in the presence of multiple other, coevolving agents? The second limitation, which we call the *encoding* problem, arises from the fact that all these experiments employed a specific, usually fairly clever, representation of the agent's problem. The investigator imposed this encoding, rather than permitting the agents to discover a useful formulation of their problem and its solution.

Our work, described below, constitutes the beginning of a systematic investigation into how these general limitations (and others not described herein) may be overcome. However promising these techniques may be for the automated discovery of models, their promise will not be fulfilled until at least these two limitations are relaxed. We now turn to the story of our investigations in that regard.

Design of the Investigation

THE MOTIVATION OF OUR RESEARCH IS TO AUGMENT EVOLUTIONARY COMPUTATION'S ability to automatically discover models. Earlier projects [11, 14] focused on the complexity problem, using a genetic algorithm approach to discover models of adaptive agents playing characteristic function games—a much more complex environment than previously reported in applications of genetic algorithms. The current project uses genetic programming, a new technique in evolutionary computation, to solve the encoding problem in this complex environment.

The Complexity Problem: Characteristic Function Games

To deal with the complexity problem, we are investigating an appreciably more heterogeneous and dynamic environment than has been addressed in the previous literature employing an evolutionary computation approach: a cooperative game with side-payments. In this paper we focus on a special case of such a game, in which three agents negotiate competitively to decide which two-agent coalition forms and to agree on a division between the coalition partners of the points constituting this coalition's value. The agent who is excluded from the coalition wins zero. The game is not symmetric, in that each coalition (ij) of two players has a unique value: $v(AB) = 18$, $v(AC) = 24$, $v(BC) = 30$. This is a dynamic environment in which three agents are coevolving throughout while they negotiate a considerably more difficult decision problem than has been addressed in the previous research cited above.

The process of these negotiations is governed by formal rules embodied in this *mechanism*. The mechanism awards the floor to one of the three agents to initiate negotiations. Technically, the mechanism makes this agent a null offer. If the agent accepts, the game ends and all three agents win zero payoff. Otherwise, the *initiator* selects a potential coalition partner as *responder* and proposes a two-way split of their coalition's value. If the responder accepts, the game ends, and the coalition partners win the payoffs as specified in the agreement, while the third player wins zero. If the responder does not accept, then it becomes the initiator and makes a proposal, tacitly

rejecting the previous offer.[4] The game continues in this fashion until an agreement is reached, or until a specified maximum number (here, six) of offers have been rejected, in which case the mechanism ends the game by default, and awards all three agents a payoff of zero.[5]

This environment is more complex in three ways than games that were previously modeled with genetic algorithms. First, in this situation *three* agents interact with each other and coevolve; thus, the coevolutionary dynamics become more complex. Each agent's performance depends not only on its reactions to the other two agents, but also on their reactions to it and to each other, and so on.

Second, characteristic function games involve many more possible contingencies and options than the games previously studied from an evolutionary computation perspective. In the Prisoner's Dilemma players have a single binary choice: *defect* or *cooperate*. The environment studied by Marimon, McGrattan, and Sargent [28] presents more options, but not many: an agent must decide whether or not to trade its current good, and then decide whether or not to consume any good it owns. In our environment, the agent holding the floor can accept the proposal or make an offer. In generating a proposal, the agent must choose one of the two other agents to be the responder and also must decide on the proposed division of payoffs. Agent A has the fewest alternatives in this game. In addition to accept, A can offer $[0 - 18]$ to B or $[0 - 24]$ to C. Thus, A selects from forty-five mutually exclusive alternatives, while agents B and C have fifty-one and fifty-seven options, respectively.

Third, the greater number of options also implies that players must react to a greater number of contingencies. If each agent in a two-player Repeated Prisoner's Dilemma game conditions its actions only on the outcome of the preceeding game, then it need react only to four contingencies, the Cartesian product of the agent's and opponent's previous binary move. In the exchange economy studied by Marimon, McGrattan, and Sargent [28], agents react to the information of what item they own and what item is offered them. Thus, the number of input states is the square of the number of items. The number of items is never greater than five, so the number of input states never exceeds twenty-five. In contrast, each agent in our environment must respond to every proposal that includes it as coalition partner. Thus, agents A, B, and C, respectively, face forty-four, fifty, and fifty-six possible contingencies.

The Encoding Problem: Genetic Programming

In genetic algorithms, models are represented by fixed-length strings in which each field of the string has a meaning preset by the programmer. The programmer must write instructions for translating these strings in order to evaluate the models. Genetic programming (GP), developed by John Koza [24, 25], manipulates flexible, tree-structured computer programs rather than fixed-length strings.

A benefit of genetic programming over traditional genetic algorithms is that players evolve not just the values of variables in their models but also the structure of their models. In genetic algorithms, how inputs are processed, how output is determined, and how inputs are mapped to outputs is preset by the programmer who designs the

fixed-length string representation of the models. In genetic programming, the programmer predetermines what information is available to the model and what output the model provides. However, the structure that determines how the input information is combined, and how the input information relates to the model's output is left to the evolutionary process. As Angeline [4] observes,

> In genetic programming, the interpretation of a particular expression tree is performed without consideration to the position of the allele. Thus, as in computer programs, the interpretation of the allele "if" in a genetic program is the same regardless of its position in the expression tree. The arbitrarily complex association between an expression and its interpreted behavior allows for an unlimited variety of dynamics to emerge naturally from the evolutionary process. Again, while there is in principle no reason why a standard genetic algorithm could not use a similar interpretation [of] functions, this has not been the practice. [P. 83]

Genetic operators in traditional genetic algorithms operate on the underlying representation of the models—the bit string—without regard to its interpretation. In contrast, the genetic operators in genetic programming are designed to preserve the syntactic correctness of the evolving programs. Therefore, genetic programming operators work more closely with the evaluated representation of a model. Again, as Angeline [4] puts it,

> In actuality, GP's use of a crossover operator that preserves the syntactic constraints of the representation language is the only true distinction between standard genetic algorithms and genetic programming. More importantly, the conceptual conveniences of a dynamic representation, non-positional interpretations, and direct manipulation of representation, highlights the narrow focus of previous genetic algorithms research. Genetic programming's most significant contribution to the field may be to re-emphasize the variety of options not yet explored. [P. 83]

System Design[6]

Model Representation

In determining the representation of a genetic programming (GP) model, one must supply three ingredients:

- Program syntax: constrains (but does not determine) the structure of the program trees to assure meaningful, and executable, models.
- Function set: specifies the class of operators to be permitted as nonterminal nodes in the program trees, thus determining which functions the models can employ.
- Terminal set: defines the symbols that comprise the leaf nodes of the program trees, thereby determining what contingencies the models can possibly recognize, and what output they can produce.

The next few subsections discuss our specifications for these three components of a GP representation. Our models are Lisp programs. However, GP systems have been implemented in other languages such as C, Prolog, and Pascal. They have also been implemented using object-oriented languages such as Lisp with CLOS, C++, and

SmallTalk. Figure 1 presents the full representation for our models. Figure 2 gives an example of a strategy for each agent. In the first example, agent A will accept any proposal in which it is offered between two and eight points (inclusive). Otherwise, it will reject the offer and propose an $((A\ 6)(C\ 18))$ split of $v(AC) = 24$.

Program Syntax

Essentially, each strategy model is a nested if-then-else statement. An if-then-else statement consists of a condition and two action statements. If the condition statement evaluates to true, then the first of the two action statements is evaluated. Otherwise, the second action statement is evaluated.

The condition statements can be a condition or a Boolean combination of conditions. Action statements are either accept, make an offer (which implies a rejection of the previous proposal), or another if-then-else statement. Therefore, the models can represent strategies of almost arbitrary complexity by nesting Boolean condition statements and if-then-else action statements.[7]

Function Set

There are two sets of functions, one for condition statements and a second for action statements. Functions for condition statements consist of the two-place Boolean operators "and," "or," and "not." Each function returns true or false by applying the Boolean operator on the two arguments. "Not" ignores its second argument (which is included to provide closure under the genetic operations).[8] The only function for action statements is if-then-else.

Terminal Set

Like the function set, the terminal set is divided into condition-statement terminals and action-statement terminals. The conditional terminal and offer terminals consist of a triple of values and a pair of values respectively. We call them *complex data structures* (CDS) to distinguish them from the single-valued terminals, true, false, and accept. A coalition structure consists of two pieces of information: a set of players in the proposed coalition, and a distribution of the coalition's value over those players. Therefore, combinations of player names and payoff amounts—which a CDS combines—are essential building blocks for constructing strategies. We designed special genetic operators, discussed later, to manipulate the CDS terminals.

The only condition terminal is the conditional complex data structure. These terminals specify contingencies for the strategy. They evaluate to true or false. A conditional CDS—written as {Player, Lower Bound, Upper Bound}—evaluates to true if, in the proposed agreement, the specified player would receive a payoff between Lower Bound and Upper Bound (inclusive). For example, suppose B offers 5 points to A. Taking $v(AB) = 18$ into account, the mechanism would record this as $((B\ 13)(A\ 5))$, listing the initiator's payoff first. Then the following conditional complex data

SYNTACTIC STRUCTURE	
strategy model::	if-then-else statement
if-then-else statement::	(IF-THEN-ELSE condition action action)
condition::	(condition-fn condition condition) | condition-terminal
action::	if-then-else statement | action-terminal
SYMBOLS	
condition-fn::	AND | OR | NOT
condition-terminal::	{PLAYER, LOWER-BOUND, UPPER-BOUND}
action-terminal::	ACCEPT | {PLAYER, AMOUNT}

Figure 1. Representation of the Strategy Models

Agent A	(IF-THEN-ELSE {A 2 8} ACCEPT {C 18})
Agent B	(IF-THEN-ELSE (NOT {B 10 30}) {C 15} ACCEPT)
Agent C	(IF-THEN-ELSE {B 10 30} (IF-THEN-ELSE {C 0 16} {B 12} ACCEPT) (IF-THEN-ELSE (OR {C 20 30} {A 12 18}) ACCEPT {A 5}))

Figure 2. An Example of a Strategy for Each Agent

structures in A's rules would evaluate as follows:

{A 5 14} true: A's proposed payoff of 5 points lies in the closed interval [5, 13].
{B 2 10} false: B's payoff of 13 would exceed the Upper Bound.
{C 0 12} true: C would get 0 (implicitly).

The action terminal set consists of the accept symbol, and the offer complex data structure, which signifies a rejection and subsequent alternative offer. The action terminals are the outputs of the program and indicate a player's move. An offer CDS—written as {Player Offer}—represents a proposal to the specified respondent (Player) who would receive the payoff stipulated by Offer. The initiator's own proposed payoff equals the coalition's value minus Offer. Therefore, if Player C makes the offer {A 10}, then the mechanism records this as the proposal ((C 14)(A 10)).

Model Evolution

Genetic Operators

We employ four genetic operators to create new straegy models, and constrain these operators to ensure that the syntactic structure is maintained. Crossover swaps subtrees from two parents. Mutation changes the symbol at a randomly picked function or terminal node in the tree. If a terminal node is chosen, then a new CDS may be created

for that node. We provide two additional operators—CDS crossover and CDS mutation—to evolve the genetic material inside a CDS. Crossover and mutation operate on entire subtrees or nodes of a program tree; they do not operate on the values inside a CDS. Therefore, without CDS-specific operators, the values inside a CDS would never change, and the agent would be prevented from evolving new offers or conditions.

Both of the CDS operators work on a randomly chosen field within a CDS. CDS mutation toggles a random bit in the chosen field, thereby creating a new value. CDS crossover selects two CDS, one from each parent, then picks a random point in the chosen field's bit string and swaps the right-hand sides of that field's bit string in the two CDS. Again, both CDS operations are constrained to ensure legal syntactic structures.

The Evolutionary Process

At generation 0, all strategies are created randomly. Care is taken to assure that programs of differing shapes and sizes are created. This way there is much diversity of population structure and content when a simulation begins. After the strategies are evaluated (a process described in the next subsection), new populations are created. Most of a new population (between 50 percent and 80 percent, depending upon parameter settings) is created by copying old strategies. The old strategies are selected probabilistically for copying based upon their scores in the evaluation. The higher the score, the greater the probability of being selected for copying. Selection is performed with replacement, so a single strategy may be copied more than once. When a strategy is copied, mutation or CDS mutation may be applied to generate a modified version for the next generation.

The rest of the new population is created using crossover. Two parents are selected probabilistically, crossover is performed on them, and the children are placed into the new population. Again, parents are chosen based upon their scores in the evaluation, and selection is with replacement, so a strategy may serve as a parent more than once. After crossover is performed, the children may be changed further by mutation, CDS crossover, or CDS mutation.

Parameters control the evolution of the population. They include specifications for the percentage of the population created via copying rather than via crossover and the probabilities of the other three genetic operators being performed on any given child.

The Game Simulation

Our research simulated negotiations in the three-player cooperative game defined above. Each agent is represented by a population of fifty strategies.[9] A tournament of games is used to evaluate all the strategies. Ideally, every strategy in an agent's population of rules would play each combination of opponent agents' strategies and do so three times, so that each of the players in the combination is given a turn to begin the negotiations. Unfortunately, that would require $3(50^3) = 375{,}000$ games per generation, and thus is not feasible. Instead, selecting randomly without replacement,

First Initiator:	Bargaining Sequence:
A	A: ((A 6)(C 18)) C: ((C 19)(A 5)) A: ACCEPT
B	B: ((B 15)(C 15)) C: ((C 18)(B 12)) B: ACCEPT
C	C: ((C 19)(A 5)) A: ACCEPT

Figure 3. Three Games Played by the Strategies Displayed in Figure 2

we match each strategy of a player with exactly 250 combinations of the other two players' strategies. Each selected combination then plays three games, each initiated by a different player. Thus, each strategy plays 750 games in the tournament.

A strategy's strength (fitness) is based on its score in the tournament, which we calculate as the average payoff (points) won by the strategy in the outcomes of games in which, sometime during the negotiations, it got the floor. In accordance with the rules of procedure governing the bargaining process (described above), the strategies displayed in Figure 2 generate the three bargaining sequences shown in Figure 3.

Comments about the Models

It is important to point out the limitations we are imposing on these artificial agents. Their strategies are constrained to be of the simple form "If x is the current proposal, then do y, else do z." Such rules are contingent only on the current offer, and thus are entirely oblivious to the history of the negotiations, and even to the value of the various coalitions. As output, they can only accept or make a proposal. The agents are prevented from developing expectations about other agents in the game. Moreover, these agents begin in the first generation with randomly constructed strategies that are likely to be simply inane. Indeed, the first generation often contains rules such as "If A offers me 15, then request 5 from A."

The next section presents results indicating that, quite remarkably, these simple agents, so severely limited cognitively, nevertheless can discover strategies that cope well with this complex and dynamic environment.

Results

THIS SECTION ANALYZES DATA FROM SIMULATIONS THROUGH 4,000 GENERATIONS. The simulations are based on a twelve-cell experimental design consisting of a Cartesian

product of parameter values for the genetic programming representation.[10] In each of these twelve cells, separate runs were conducted for three different seeds of the pseudo-random number generator. The data result from 12×3 = 36 simulations. We recorded data at the random start (generation 0), and then every five generations in the 4,000-generation runs; thus, the results for each of the 36 runs are based on variables contained in 801 reports, each of which averages the results from up to 3(players) × 50(rules/player) × 750(games/rule) = 112,500 distinct bargaining sequences played in the tournament for that generation.

This paper focuses on the results from two cells in the twelve-cell design, which we will refer to as run 1 and run 2.[11] On average, the results for these two runs are very similar. Later, we shall see that the results for these two cells are in fact representative of the other cells in the design. Also, in discussing these results, we shall compare where possible the behavior of these artificial agents with the behavior of human subjects who were observed in a laboratory study of a game that is comparable to ours ([22]: game IV, experiments 1–3 combined, $n = 48$ triads).[12]

Game Length and Agreement Frequencies

Some of the variables adjust quickly from the random start, so that the averages for the first 1,000 generations differ little from later generations. The first pair of rows in Table 1 indicate that, on average, less than two rounds is needed to complete the bargaining. The second pair of rows report that roughly 84 percent of the games in run 1, and 87 percent in run 2, result in a coalition agreement (rather than ending by default), both in the first 1,000 generations and in generations $1{,}000^{+} - 4{,}000$ combined. Although this is a high level of agreement, the artificial agents are somewhat less successful than human subjects in forming agreements: all forty-eight of Kahan and Rapoport's triads succeeded in forming a coalition.

The bottom six rows of Table 1 break down the overall frequency of agreement into the frequencies with which each two-agent coalition forms. From detailed examinations of the graphs of coalition frequencies across generations (not shown), we note that, in a typical generation, most of the agreements involve just two of the three coalitions, but that the pair of coalitions that currently is prevalent changes across generations. Nonetheless, the overall averages for the 1,000-generation subpopulations displayed in Table 1 indicate that the BC coalition is somewhat disadvantaged in these data, particularly in the second run. However, no coalition is frozen out of these negotiations indefinitely.

This contrasts with the oft-observed proclivity of human subjects in characteristic-function games to opt for the "most valuable" coalition: the forty-eight coalitions observed by Kahan and Rapoport are distributed $n(AB) = 6$, $n(AC) = 7$, $n(BC) = 35$. Yet, if payoffs conform to the quota solution, then each player should be indifferent as to which coalition that player joins. Thus, it might seem that, at equilibrium, all two-person coalitions should be equally likely to form in this game.[13] Apparently the artificial agents, ignorant of this game's characteristic function, were not distracted by the fact that (BC) is the "most valuable" coalition.

Table 1.

n	Generations (thousands)				
	0–1	1^+-2	2^+-3	3^+-4	1^+-4
	(201)	(200)	(200)	(200)	(600)
Mean game length					
run 1	1.843	1.763	1.851	1.860	1.825
run 2	1.767	1.765	1.876	1.862	1.834
Agreement frequency					
run 1	0.842	0.854	0.836	0.836	0.842
run 2	0.873	0.882	0.861	0.858	0.867
Coalition frequency					
AB					
run 1	0.255	0.360	0.274	0.233	0.289
run 2	0.379	0.336	0.360	0.328	0.342
AC					
run 1	0.350	0.299	0.359	0.360	0.339
run 2	0.324	0.429	0.345	0.389	0.388
BC					
run 1	0.237	0.195	0.203	0.242	0.213
run 2	0.169	0.116	0.156	0.140	0.137

Coalition Payoffs and the Quota Solution

Various solution concepts from cooperative game theory prescribe for this game a *quota solution* (e.g., see [35], p. 177n), akin to a vector of equilibrium prices, which specifies the equilibrium values of the three agents' positions in this situation when each agent seeks to maximize its total payoff under perfect competition. Specifically, the quota solution to this game is $q = (q_A, q_B, q_C) = (6, 12, 18)$. It prescribes that if coalition (ij) forms in this game, then the payoffs should be divided between coalition partners in accordance with their quotas, while the third player must, in this game, win zero. For example, the quota solution prescribes that if A and B form a coalition, then they should split $v(AB) = 18$ such that A wins $q_A = 6$ points and B gets $q_B = 12$. Note that $q_A + q_B = v(AB)$.

We are interested particularly in determining the extent to which these simple agents, so cognitively limited, can coevolve in this rather complex bargaining environment to produce coalition agreements that roughly approximate those prescribed by the theory of cooperative games. *Note well:* the rules are being selected in the evolutionary process on the basis of their fitness as measured by the average payoffs they win in the tournament, *not* by how closely their payoffs approach the quota solution. For this purpose, to avoid concentrating on rules that are outliers, we focus on the average payoffs won in coalition agreements by the rule of each agent that, for the generation being reported, earned the median overall payoff among the agent's population of fifty rules. To provide a summary measure, let us define the *mean absolute deviation from the quota*: to wit, we compute for each agent the absolute value of the difference

between the agent's quota and the average payoff won in coalition agreements by the agent's median rule, then average the results across the three agents. Figures 4 through 6 (respectively, 7 through 9) plot the 801 values of this summary measure for each of the three seeds used in the first (resp., second) run.[14]

Figures 4 through 9 display some very encouraging results. In Figure 4, after a substantial departure from the quotas in the early generations, the median rule approaches the quota solution by the nine-hundredth generation, then remains within about one to two points of the quota until a medium strength random shock to the system occurs at roughly generation 2,200. Beyond this point, the system recovers to within a point of the quota in generation 3,000 and again, after further small shocks, in generation 4,000. Note that the genetic operations can produce enormous changes in just one generation. For example, even if identical parents mate, subtree crossover can produce strikingly different offspring, and this can reverberate throughout the coevolving system of the three agents' populations of rules. Nonetheless, although the graphs depicted in Figures 7 through 9 are somewhat less dramatic (as is consistent with the milder parameter values used in run 2), in all six figures the systems recover rather nicely from these shocks and return to the neighborhood of the game's quota solution.[15]

Although the graphs indicate differences in details, the six sets of data (two runs, three seeds each) exhibit very similar results when we compute the average across each 1,000-generation segment. Table 2 presents the mean absolute deviation from the quota (and standard error of the mean) for the two respective runs, averaged within each 1,000-generation segment and across the three seeds. In both runs, the deviation from the quota is less in subsequent generations than during the first 1,000-generation adjustment period, averaging over these subsequent generations to about 2.6 and 2.1 points, respectively, from the quota solution.

Table 3 presents for the two runs the average payoffs (and standard error) won as a coalition member by each agent's median rule, again averaged across seeds within 1,000-generation segments. The means across the last 3,000 generations indicate that agent A is averaging one to two points more than its quota ($q_A = 6$), whereas both agents B and C are averaging within less than a point of their quotas ($q_B = 12$, $q_C = 18$) in the tournament of games. Thus, on average, these systems of simple rules coevolve into the neighborhood of the cooperative game-theoretic solution.

Equilibrium

We now describe the extent to which, at "equilibrium," the artificial agents reach agreements approximating those prescribed by game theory. In these simulations, "equilibrium" is at best transitory. As demonstrated in Figures 4 through 9, the behavior of the three agents remains stable for a while, becomes somewhat chaotic as random changes shake the system, and then settles down again. This cycle between stable and nonstable states is common to all thirty-six simulations in the twelve-cell design. To measure the extent to which the system is presently in a state that approximates an equilibrium, we use the following operational construct. Let $x(t)$ denote any variable x at a point t in the sequence of generation reports. In particular,

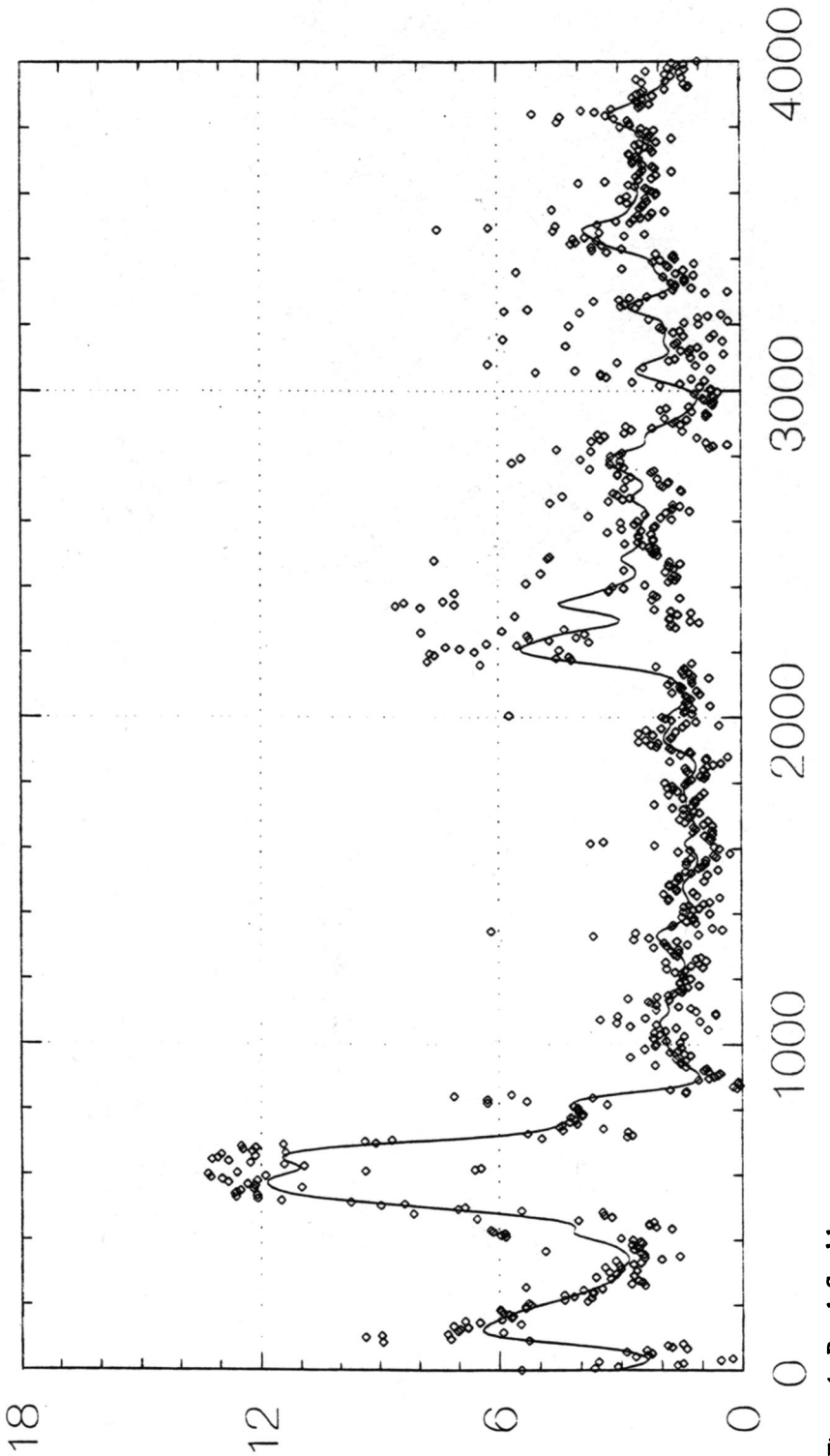

Figure 4. **Run 1, Seed 1**

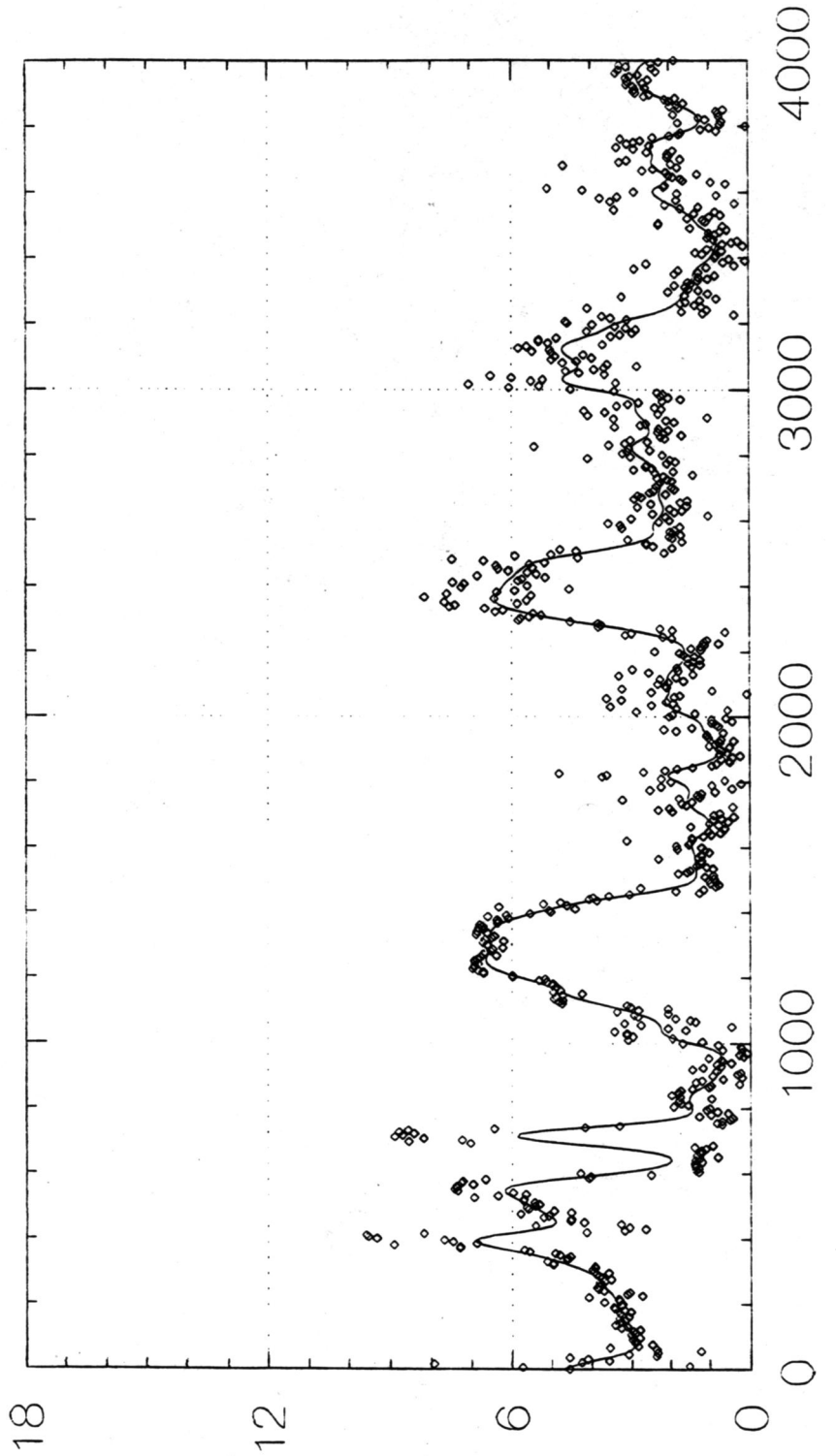

Figure 5. **Run 1, Seed 2**

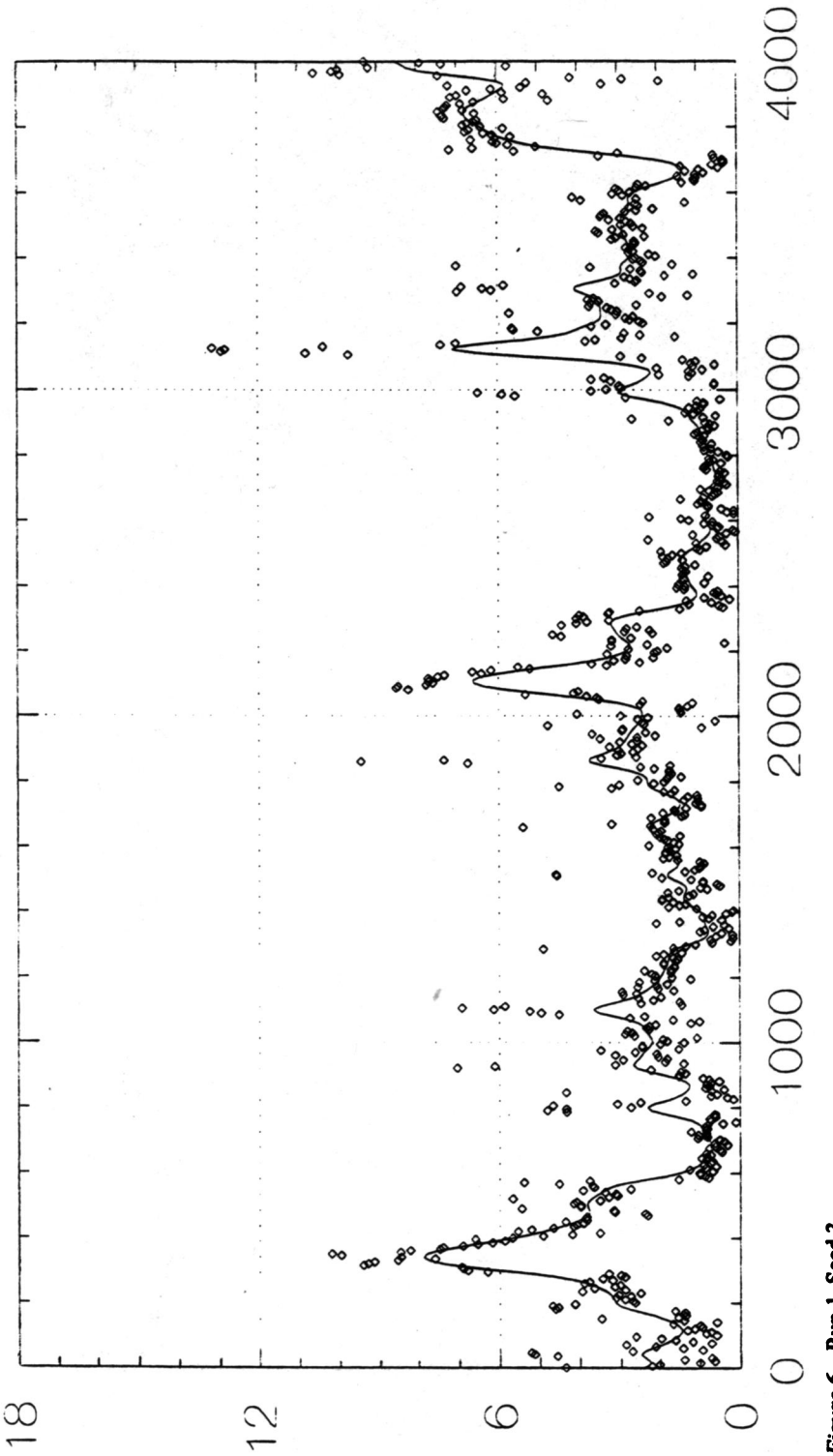

Figure 6. **Run 1, Seed 3**

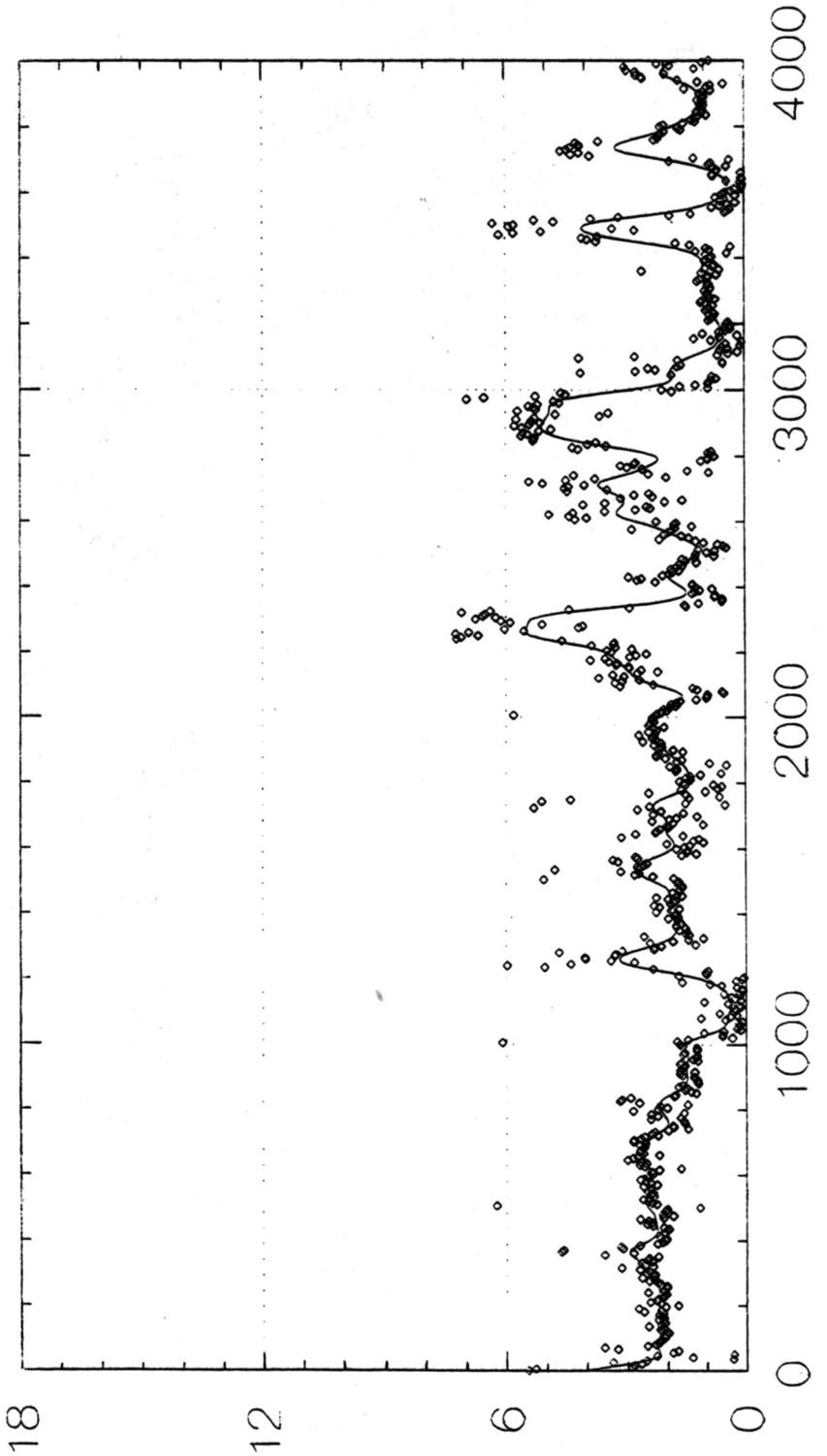

Figure 7. **Run 2, Seed 1**

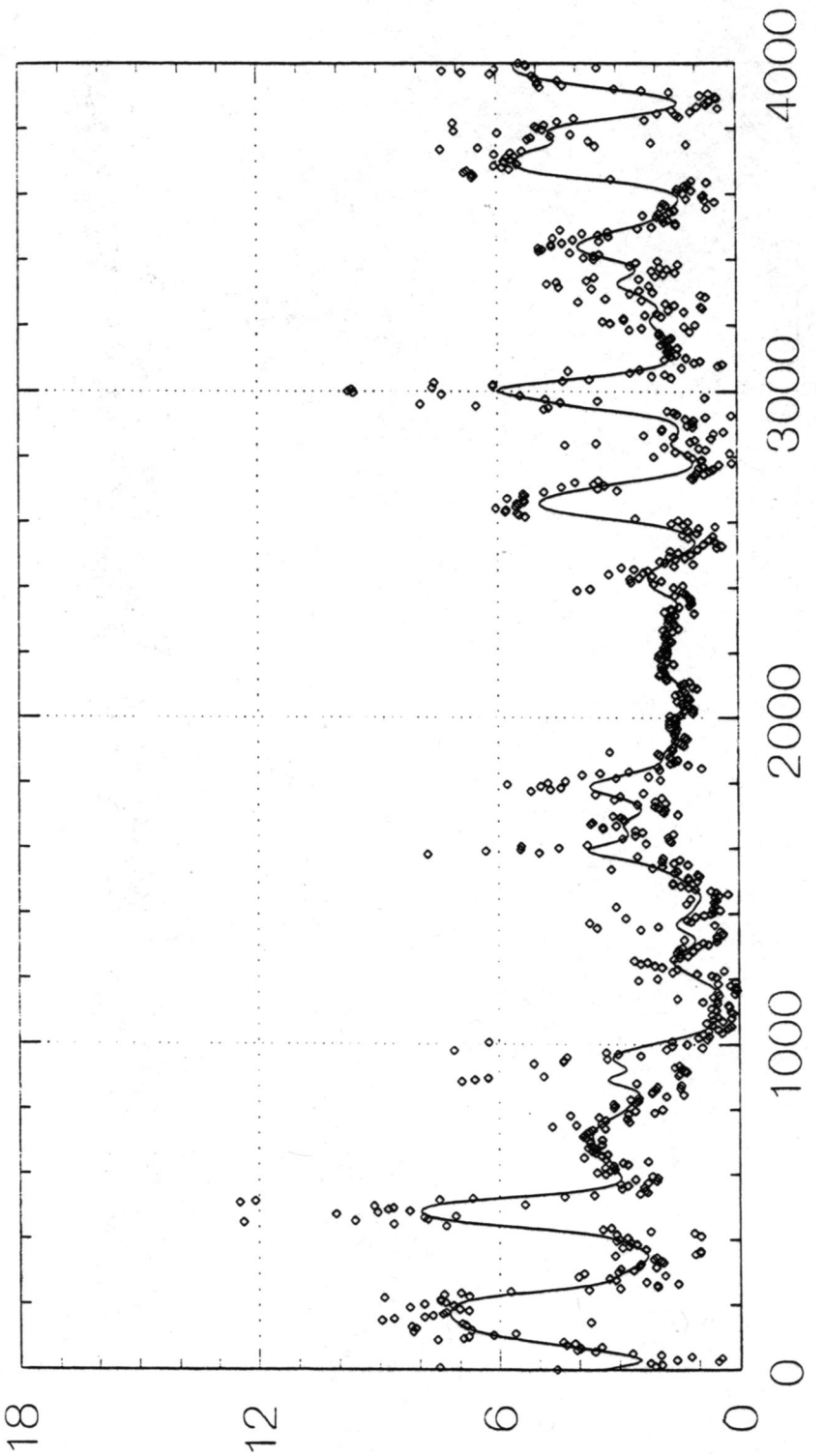

Figure 8. **Run 2, Seed 2**

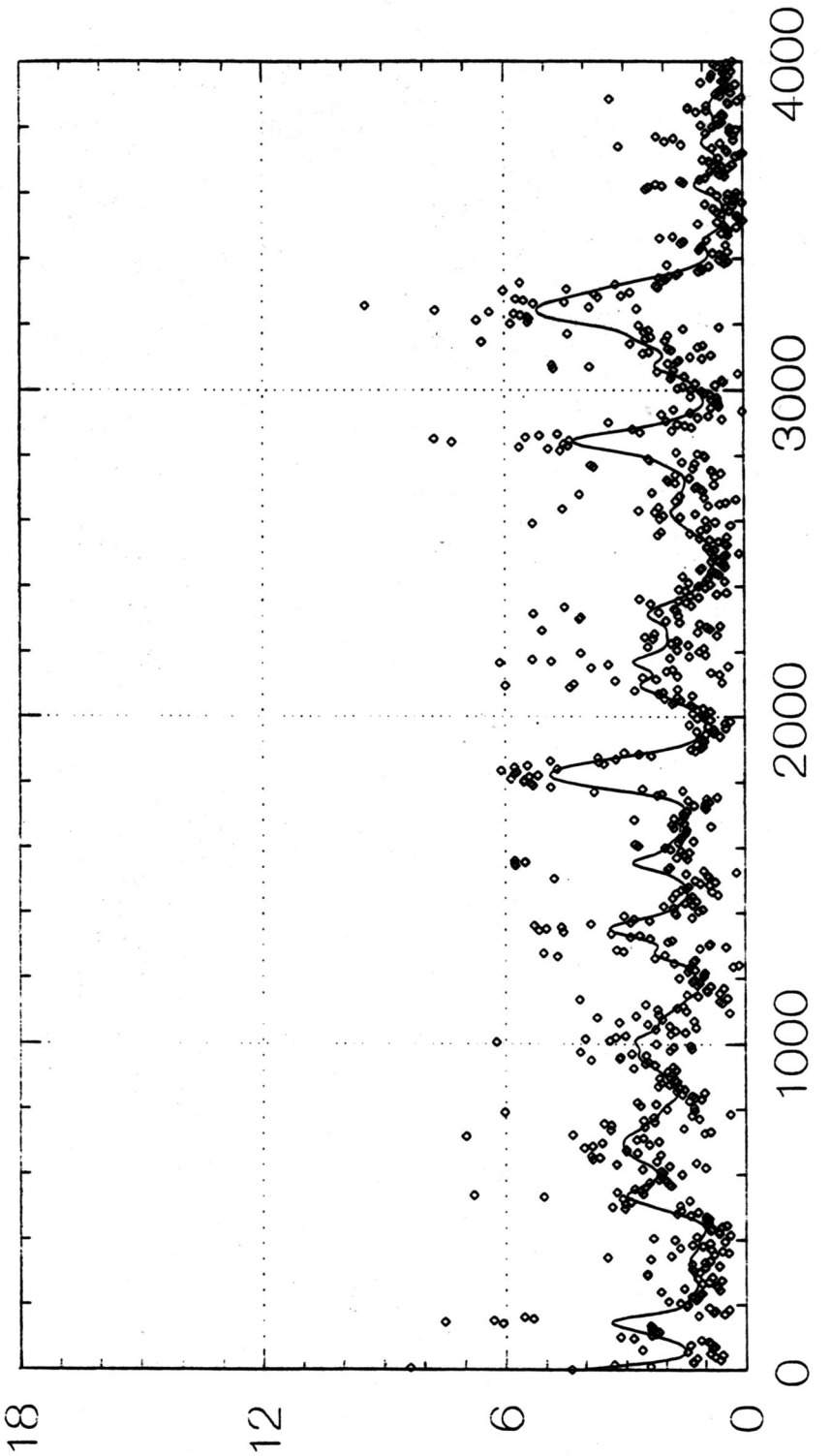

Figure 9. **Run 2, Seed 3**

Table 2.

	Generations (thousands)				
	0–1	1^+-2	2^+-3	3^+-4	1^+-4
Mean absolute deviation from quota, median rule (s.e.)					
run 1	3.861	2.173	2.614	2.962	2.583
	(0.198)	(0.101)	(0.132)	(0.126)	(0.076)
run 2	2.807	1.911	2.375	2.064	2.116
	(0.108)	(0.095)	(0.117)	(0.119)	(0.068)

Table 3.

	Generations (thousands)				
	0–1	1^+-2	2^+-3	3^+-4	1^+-4
Average payoff as coalition member, median rule (s.e.)					
Agent A					
run 1	9.566	7.212	8.147	8.773	8.004
	(0.264)	(0.129)	(0.193)	(0.189)	(0.011)
run 2	8.561	7.243	7.527	7.379	7.383
	(0.133)	(0.136)	(0.153)	(0.156)	(0.095)
Agent B					
run 1	12.806	11.802	12.304	11.815	11.974
	(0.311)	(0.187)	(0.239)	(0.278)	(0.142)
run 2	11.567	11.732	11.425	11.439	11.532
	(0.225)	(0.195)	(0.216)	(0.223)	(0.124)
Agent C					
run 1	17.293	19.516	18.192	17.435	18.533
	(0.260)	(0.159)	(0.190)	(0.230)	(0.121)
run 2	17.171	17.736	19.101	18.938	18.591
	(0.228)	(0.180)	(0.202)	(0.175)	(0.112)

we shall be concerned with the case in which $x(t)$ represents the average payoff won as a member of a coalition agreement by an agent's *best rule*, that is, the agent's rule that wins overall the greatest average payoff. First, let us define $x(t)$ to be *settled prima facie* if both of the following conditions obtain: (1) the m: = 5 immediate predecessors of $x(t)$, with mean $x(m,t)$, have a standard deviation no greater than d: = 0.5, and (2) $x(t)$ deviates no more than d from $x(m,t)$. Second, if $x(t)$ is settled prima facie, then we define to be *settled retrospectively* any of its m immediate predecessors that deviates no more than d from $x(m,t)$. Then, we say that $x(t)$ is *settled at t* if it is settled prima facie or retrospectively at t. Finally, we define the *system* of agents to be *settled at t* if, for each of the three agents, the average payoff won by that agent's best rule when it succeeds in forming a coalition agreement is settled at t.

In our discussion of equilibrium behavior (as defined by settledness), rather than discuss details of just the two runs mentioned earlier, we report results in terms of weighted averages across all thirty-six simulations. Specifically, these averages are reported for those generations ($n = 3,158$, out of a total of $801 \times 12 = 9,612$ generation reports) during which, as defined above, the systems were settled.

Coalition Payoffs during Equilibrium

During the periods in which the system was settled, the best rules of artificial agents *A*, *B*, and *C*, respectively, averaged in their coalition agreements a payoff of 8.6, 11.9, and 18.5 points. (For these same periods, the payoffs won by each of the agent's median rules averaged only about a half-point less: 8.1, 11.4, and 18.1, respectively.) Thus, *A*'s best rule averages 2.6 points more than the quota in these simulations. In comparison, human subjects in Kahan and Rapoport's experiments average 4.8, 12.7, and 17.7 (transformed—see note 15) to players *A*, *B*, and *C*, respectively. Presumably, *A* is disadvantaged by the proclivity of human subjects, unlike the artificial agents, to form the (*BC*) coalition.[16] Players *B* and *C* approximate their quotas in both the human and artificial data.

Overall, the mean absolute deviation from the quota solution is roughly 2.5 points in the artificial data, rivaling that of 2.0 points in the human data. Moreover, in the artificial data, the distance from the quota decreases in a strictly monotonic progression as the number of agents whose best rule is settled increases in unit steps from zero to three: the more settled the system, the more it approaches the game-theoretic solution. In addition, fully 94.2 percent of the coalition agreements formed by artificial agents, but only 68.7 percent of those formed by the humans in Kahan and Rapoport's laboratory study [22], lie closer to the quota solution than to an even split of the coalition's value. In this respect, the artificial agents, perhaps because they are not encumbered by equity considerations or the cognitive and social prominence of even splits, are closer than humans to the game-theoretic solution.

Opportunity Costs and Best Responses

But do these rules approximate optimal responses to the environment? The quota solution can be implemented by a class of equilibrium strategies (stationary, subgame-perfect, Nash), and thus is consistent with rational play of the bargaining situation represented as a noncooperative game in extensive (tree) form [27, 33, 34]. We have demonstrated above that the cooperative agreements reached in these games lie roughly in the neighborhood of the cooperative game-theoretic solution, but now investigate further whether the rules embody strategies that, at least approximately, constitute best responses to the rules they are playing in the tournament.

If so, then the rules incur no opportunity cost. We operationalize opportunity cost in this situation as follows: Consider any game in the tournament in which a rule gets the floor in the negotiations. At this stage in the bargaining, the rule can make an offer to either of the other two agents. We compute the rule's *opportunity cost* for this game as the maximum payoff the rule could win by making an offer that would be accepted

immediately, minus the actual payoff achieved by the rule in this game's outcome.[17]

For example, consider the bargaining sequence displayed in Figure 3 for the game in which *B* is the first initiator. *B* achieves a payoff of twelve points in this game's outcome. Yet the strategies shown in Figure 2 reveal that *A* (respectively, *C*) would accept an offer from *B* in which *B* wins sixteen points (respectively, thirteen). The maximum of these opportunities for *B* is sixteen. Thus, in agreeing to an offer of twelve points, *B* incurs an opportunity cost of 16–12 = 4. Clearly, *B*'s strategy is not a best response to the strategies of the other agents. At a Nash equilibrium, each agent's opportunity cost would be zero.

Table 4 presents the mean opportunity cost (and standard error of the mean) incurred in the two runs for each of the three agents' median rules, averaged across games, seeds, and 1,000-generation segments. The results clearly indicate that these rules are failing to exploit opportunities present in their opponents' strategies: most significantly, the median rules of agents *B* and *C* are incurring average opportunity costs of about 5.5 and 7 points, respectively, in the last 3,000 generations. These agents' median rules do not approximate best response strategies, and little learning in this regard is evident in these data.

Focusing on the opportunity costs accrued during equilibrium play tells a similar story. The opportunity costs incurred in all thirty-six simulations during the settled periods by the best rule of agents *A*, *B*, and *C* averaged approximately zero, three, and four points, respectively. Thus, *A* employs a strategy that is a best response to the strategies being played by the other agents, but *B* and *C* do not.

Thus, although these cognitively simple, adaptive agents coevolve in these runs to produce coalition agreements that approximate the cooperative game-theoretic solution, they do not discover best responses to their opponents' strategies. We wonder whether the human subjects in the Kahan and Rapoport study formulated strategies that more closely approximated a Nash equilibrium.

Conclusion

THESE RESULTS ARE ENCOURAGING, BUT NOT UNALLOYEDLY SO. In a cognitively deprived, information-poor environment (e.g., no memory of previous play, lack of full knowledge of the game's conditions), our agents have been able to evolve behavior that accords well with what has been prescribed by game theory. The agents more or less find the quota solution. They do so with rules we can demonstrate to be nonoptimal: even the best rules during equilibrium play have significant opportunity costs. Nonetheless, good play has been achieved and rules that in fact do well against a broad set of competing rules have been discovered. That this has been done in the current context, when a normative solution is known, augurs well for contexts in which no unique solution is known. We may hope that computational approaches, GP in particular, will yield insights into games for which game-theoretic solution theory provides no clear answer.

The good, though nonoptimal, performance of our artificial agents should be evaluated in the context of the large search space they face. Let us calculate: Agent *A*

Table 4.

	Generations (thousands)				
	0–1	1^+-2	2^+-3	3^+-4	1^+-4
Average opportunity cost, median rule (s.e.)					
Agent A					
run 1	2.378	1.688	1.796	2.035	1.981
	(0.276)	(0.183)	(0.235)	(0.215)	(0.130)
run 2	1.349	1.522	0.638	0.448	0.869
	(0.225)	(0.236)	(0.222)	(0.193)	(0.134)
Agent B					
run 1	5.347	5.146	5.685	5.410	5.577
	(0.256)	(0.231)	(0.226)	(0.203)	(0.130)
run 2	6.684	6.150	5.335	5.190	5.559
	(0.209)	(0.188)	(0.195)	(0.207)	(0.117)
Agent C					
run 1	6.804	6.090	7.493	5.921	6.846
	(0.220)	(0.177)	(0.185)	(0.194)	(0.114)
run 2	7.911	7.757	7.112	6.594	7.155
	(0.200)	(0.181)	(0.216)	(0.191)	(0.119)

can receive 19 different offers from agent *B* (0–18), and 25 different offers by *C* for a total of 44 different possible inputs. *A* then has 44 + 1 = 45 different responses that are possible. *A* can respond with any of *A*'s possible offers, or *A* can end the game by accepting. Thus, *A* must choose among 44 × 45 = 1,980 different response policies.

Even so, *A*'s problem is much harder, since the rules and responses for *B* and *C* are relevant. *B*'s corresponding numbers are: (on input) 19 + 31 = 50, (on output) 51, and (overall) 50 × 51 = 2,550. And, *C*'s corresponding numbers are: (on input) 25 + 31 = 56, (on output) 57, and (overall) 56 × 57 = 3,192. Thus, the system of players must search over a space of 1,980 × 2,550 × 3,192 = 1.6×10^{10} combinations of response policies.

There are very many more possible combinations of rules, since:

1. Every player is forced (by the design of the experiment) to have a complete decision policy. That is, every player must be able at all times to respond to every possible valid input condition for that player.
2. Every such (complete) decision policy for playing this game is possible syntactically. That is, every response policy may be represented rather directly in the syntax of the agents' rule language.
3. Any given complete decision policy can be represented by a large number of syntactically distinct, but behaviorally equivalent, rules.

Conservatively, there will be more than 100 logically (behaviorally) equivalent representations per complete decision policy. Thus, the search space is at least on the order of 10^{12}. Note further that:

4. Our (artificial) players are coevolving; they are not playing against constant opponents.
5. In a run of 4,000 generations with 3 players and 50 rules per player per generation, the maximum number of rules created in the simulation is 6×10^5.

Given this, we can but stand in awe of these simple agents' performance in achieving quite reasonable results.

Much, of course, remains to be done. Longer runs will likely yield better results and further insight. We are conducting them. Better architectures and more complex problems need to be investigated. Still, the results to date can only lend credence to the belief that automatically discovered models, decision policies, and strategies may someday be routinely in our service.

NOTES

1. See, in particular, chapter 10 of Koza [24], in which he investigates the symbolic regression problem. A functional relationship, $y = f(x)$, known to the experimenter, but not to the genetic program, is used to generate data, i.e., $<x, y>$ pairs, which are used by the genetic program to evolve a function that fits the data. Koza's work here does not include studies of reactions to noisy data.

2. Any tragedy is from the point of view of the players. If the players are firms setting prices in a market, then from the point of view of social welfare, failure (mutual defection) may be the preferred outcome.

3. Throughout this paper we assume basic familiarity with genetic algorithms and genetic programming. Excellent introductions to genetic algorithms (e.g., [15, 19]) and genetic programming (e.g., [24, 25]) are readily available.

4. Actually, the responder might return the same proposal. The mechanism treats such an echo like any other proposal, recording an agreement other than the default outcome if and only if the response is accepted.

5. The restriction to six offers per game is not very limiting. In our simulations, almost all agreements are reached within three offers. Initially, we limited games to ten offers, but discovered that any game that went beyond five offers continued until the mechanism ended the game by default. Therefore, we lowered the limit to six to speed up the simulations.

6. This section summarizes the details of our system. In [13], we present additional details of our system and discuss the differences between our system and the original genetic programming system developed by Rice and Koza [24].

7. Actually, we limit the complexity of the strategies to a maximum tree depth of eight. See [13] for details.

8. In the second example of Figure 2, we omit the second argument of NOT, which is ignored anyway, to improve readability.

9. Population size also is controlled by a parameter.

10. For all runs reported in this paper, mutation probability = 0.5. The twelve-cell design is based on the full Cartesian product (2×2×3) of the following parameter sets: crossover fractions = {0.2, 0.5}, leaf crossover probabilities = {0.2, 0.5}, and leaf mutation probabilities = {0.2, 0.5, 0.8}. See [13] for details.

11. In the first (respectively, second) run, crossover fraction = CDS crossover probability = 0.5 (resp., 0.2), and CDS mutation probability = 0.2 (resp., 0.8).

12. The characteristic function of their game IV is: $v'(AB) = 66$, $v'(AC) = 86$, $v'(BC) = 106$, else $v'(S) = 0$. To compare payoffs achieved by human subjects in game IV to those obtained by our artificial agents, we use the following transformation for positive-valued coalitions: $v(S) = 0.3[v'(S)–6]$ if $v'(S) > 0$; hence $v(AB) = 18$, $v(BC) = 24$, $v(BC) = 30$.

13. We have restricted these agents to employ strategies that do not select actions pro-

babilistically (i.e., pure strategies). In contrast, for a game in which negotiations are governed by rules that differ from those used in our games, Selten [34, ch. 11] has characterized probabilistic strategies for negotiating three-person quota games that, at equilibrium, both implement the quota solution and generate precise coalition probabilities such that, indeed, *BC* is the most likely coalition.

14. The function plotted in Figures 3 through 8 is fitted with the SYSTAT software by distance-weighted least squares, with tension parameter: = 0.0001, a considerably less stiff tension value than SYSTAT's default value, $1/n$ (= 0.00125, given $n = 801$).

15. Given these results, we were confident that the system in Figure 6 would recover from the large shock experienced after generation 3,900 once we extended this run beyond generation 4,000. It did.

16. During the periods in which the system was settled, taking into account the results of all twelve cells in the design, the best rules of the artificial agents formed coalitions with the following frequencies: $p(AB) = 0.359$, $p(AC) = 0.360$, $p(BC) = 0.162$.

17. By this measure, an agent's opportunity cost in a particular game can be negative: the agent may accept an offer of a greater payoff than it could achieve by making an offer to another agent that would be accepted.

REFERENCES

1. Abreu, D., and Rubinstein, A. The structure of Nash equilibrium in repeated games with finite automata. *Econometrica, 56* (1988), 1259–1281.

2. Albers, W., and Laing, J.D. Prominence, competition, learning, and the generation of offers in computer-aided experimental spatial games. In R. Selten (ed.), *Game Equilibrium Models, III: Strategic Bargaining*. Berlin: Springer-Verlag, 1991, pp. 141–185.

3. Andreoni, J., and Miller, J.H. Auctions with adaptive artificially intelligent agents. University of Wisconsin, Social Systems Research Institute Working Paper 9032, 1990.

4. Angeline, Peter J. Genetic programming and emergent intelligence. In K.E. Kinnear, Jr. (ed.), *Advances in Genetic Programming*. Cambridge, MA: MIT Press, 1994, pp. 75–97.

5. Aumann, R. Perspectives on bounded rationality. Notes of lectures presented at the Workshop on Bounded Rationality, Department of Economics, Stanford University, July 25, 1989.

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

7. Axelrod, R. The evolution of strategies in the iterated prisoner's dilemma. In L. Davis (ed.), *Genetic Algorithms and Simulated Annealing*. Los Altos, CA: Morgan Kaufmann, 1987, pp. 32–41.

8. Camerer, C. Behavioral game theory. In R.M. Hogarth (ed.), *Insights in Decision Making: A Tribute to Hillel J. Einhorn*. Chicago: University of Chicago Press, 1990, pp. 311–336.

9. Cornell, B., and Roll, R. Strategies for pairwise competitions in markets and organizations. *Bell Journal of Eonomics, 12* (1981), 210–216.

10. Dawes, R.M. *Rational Choice in an Uncertain World*. San Diego: Harcourt Brace Jovanovich, 1988.

11. Dworman, G. Games computers play: simulating characteristic function game playing agents with classifier systems. The Wharton School, University of Pennsylvania, draft, 1994.

12. Dworman, G.; Kimbrough, S.O.; and Laing, J.D. On automated discovery of models using genetic programming in game-theoretic contexts. In J.F. Nunamaker, Jr., and R. Sprague, Jr. (eds.), *Proceedings of the 28th Hawaii International Conference on System Sciences, Decision Support and Knowledge-Based Systems*. Los Alamitos, CA: IEEE Press, 1995, pp. 428–438.

13. Dworman, G.; Kimbrough, S.O.; and Laing, J.D. Implementation of a genetic programming system in a game-theoretic context. The Wharton School, University of Pennsylvania, Department of Operations and Information Management, working paper 95–01–02, 1995.

14. Dworman, G., and Schoenberg, E. Games computers play (part 2): applying a bucket brigade to a classifier system implementation of characteristic function games. The Wharton School, University of Pennsylvania, draft, 1993.

15. Goldberg, D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning*.

Reading, MA: Addison-Wesley, 1989.

16. Hammerstein, P., and Reichert, S.E. Payoffs and strategies in territorial contests: ESS analyses of two ecotypes of the spider Agelenopsis Aperta. *Evolutionary Ecology, 2* (1988), 115–138.

17. Harsanyi, J.C., and Selten, R. *A General Theory of Equilibrium Selection in Games*. Cambridge, MA: MIT Press, 1988.

18. Ho, T.-H. Finite automata play repeated prisoner's dilemma with information processing costs. University of Pennsylvania, The Wharton School, Department of Operations and Information Management, working paper, 1991.

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

20. Holland, J.H. Using classifier systems to study adaptive nonlinear networks. In D. Stein (ed.), *Lectures in the Sciences of Complexity*. Reading, MA: Addison-Wesley Longman, 1989, pp. 463–499.

21. Holland, J.H.; Holyoak,K.J.; Nisbett, R.E.; and Thagard, P.R. *Induction: Processes of Inference, Learning, and Discovery*. Cambridge, MA: MIT Press, 1986.

22. Kahan, J.P., and Rapoport, A. Test of the bargaining set and kernel models in three-person games. In A. Rapoport (ed.), *Game Theory as a Theory of Conflict Resolution*. Dordrecht: D. Reidel, 1974.

23. Karjalainen, R. *Using Genetic Algorithms to Find Technical Trading Rules*. Ph.D. dissertation, University of Pennsylvania, The Wharton School, 1994.

24. Koza, J.R. *Genetic Programming*. Cambridge, MA: MIT Press, 1992.

25. Koza, J.R. *Genetic Programming II*. Cambridge, MA: MIT Press, 1994.

26. Kreps, D.; Milgrom, P; Roberts, J.; and Wilson, R. Rational cooperation in the finitely-repeated prisoner's dilemma. *Journal of Economic Theory, 27* (1982), 245–252.

27. Laing, J.D. A noncooperative model of bargaining in simple spatial games. In R. Selten (ed.), *Game Equilibrium Models, III: Strategic Bargaining*. Berlin: Springer-Verlag, 1991, pp. 80–117.

28. Marimon, R.; McGrattan, E; and Sargent, T.J. Money as a medium of exchange in an economy with artifically intelligent agents. *Journal of Economic Dynamics and Control, 14* (1990), 329–373.

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

30. Miller, J.H. The coevolution of automata in the repeated prisoner's dilemma. Santa Fe Institute, Economics Research Program Working Paper 89–003, 1989.

31. Nash, J. Non-cooperative games. *Annals of Mathematics, 54* (1951), 286–295.

32. Selten, R. Reexamination of the perfectness concept for equilibrium points in extensive games. *International Journal of Game Theory, 4* (1975), 25–55.

33. Selten, R. A noncooperative model of characteristic-function bargaining. In V. Boehm and H. Nachthanp (eds.), *Essays in Game Theory and Mathematical Economics in Honor of Oskar Morgenstern*. Wien-Zürich: Wissenschaftsverlag Bibliographisches Institut Mannheim, 1981, pp. 313–351.

34. Selten, R. *Models of Strategic Rationality*. Dordrecht: Kluwer, 1988.

35. Shubik, M. *Game Theory in the Social Sciences: Concepts and Solutions*. Cambridge, MA: MIT Press, 1982.

36. Tversky, A., and Kahneman, D. Judgement under uncertainty: heuristics and biases. *Science, 185* (1974), 1124–1131.