ON THE INFERENCE OF STRATEGIES

by

Charles Swart[1]
Department of Computer Science
Oregon State University
Corvallis, Oregon, 97331

and

Dana Richards
Department of Computer Science
University of Virginia
Charlottesville, Virginia, 22903

ABSTRACT

We investigate the use of automata theory to model strategies for nonzero-sum two-person games such as the Prisoner's Dilemma. We are particularly interested in infinite tournaments of such games. In the case of finite-state strategies (such as the well-known strategies ALL D or TIT FOR TAT) we use graph traversal techniques to show the existence of a (non-terminating) procedure for detecting our opponent's strategy and developing an "optimal" defense. We also investigate counter machine and Turing machine strategies. We show that the optimal defense to a counter machine strategy need not be finite-state, thus disproving a previous conjecture. We show that determining an optimal defense to an arbitrary Turing machine strategy is undecidable.

1. INTRODUCTION

Consider the following nonzero-sum two-person game, called the *Prisoner's Dilemma*, given by the payoff matrix in Figure 1. (In this description we follow closely the presentation given by Hofstadter [5].) In the payoff matrix with (i, j) entry $(P(i, j), Q(i, j))$, $P(i, j)$ represents the payoff to player X when X chooses move i and Y chooses move j. Similarly, $Q(i, j)$ represents the payoff to player Y.

Payoff		Player Y C	D
Player X	C	(3,3)	(0,5)
	D	(5,0)	(1,1)

Figure 1

[1] Current address: Floating Point Systems, Box 23489, Portland, Oregon 97223

The origin of the term Prisoner's Dilemma comes from the following hypothetical situation: Suppose you and an accomplice attempt to commit a crime and are caught. The prosecuting attorney offers each of you, independently, the following "deal." If neither of you confess (i.e., you both *cooperate* with each other), you will both be convicted and will each serve two years in prison. However, if either of you confesses *(defects* against your partner) and the other doesn't, the one who confesses will be released and the other will serve five years. If you both confess you will each serve four years. This situation can be described by the matrix of Figure 2 where $(-x, -y)$ means that you get x years in jail and your accomplice gets y years. If you add five to each element in Figure 2 you get the original matrix of Figure 1, referred to as the canonical Prisoner's Dilemma payoff matrix.

The dilemma of this game arises from the following analysis. If Y chooses C (to cooperate) the best choice for X is D (to defect), for X then will win 5 instead of 3. If Y chooses D then X still does best by choosing D, winning 1 instead of 0. Consequently, the best choice for X is to defect regardless of what Y does. The same analysis for player Y indicates that Y should also defect. Consequently, both players receive 1 unit when they both could have received 3 units by mutual cooperation.

The Prisoner's Dilemma has been used by Axelrod and Hamilton [2] and by Hofstadter [5] to study the evolution of cooperation. Imagine that X and Y play an infinite sequence, or *tournament*, of games, each using the past history of the other player to determine its next move. The general condition for a Prisoner's Dilemma is a payoff matrix of the form given in Figure 3, where R is the reward for mutual cooperation, T is the temptation to defect, P is the punishment for mutual defection, and S is the sucker's payoff. The interesting case occurs when 1) $T > R > P > S$ and 2) $(T+S)/2 < R$. The first condition gives the dilemma. The second guarantees that if the two players get locked into out-of-phase alternations (i.e., on one move X defects and Y cooperates, on the next move X cooperates and Y defects, etc.) then both players will do worse than with mutual cooperation.

Payoff		Accomplice *C*	Accomplice *D*
You	*C*	(-2,-2)	(-5,0)
	D	(0,-5)	(-4,-4)

Figure 2

Payoff		Player Y	
		C	D
Player X	C	(R, R)	(S, T)
	D	(T, S)	(P, P)

Figure 3

We will examine various strategies for playing such a tournament using standard computing devices, specifically, Turing Machines, counter machines and finite-state automata. In this model a player will choose a move, be informed of the other player's move, then perform some computation to choose its next move.

A finite-state strategy for player X can be represented by a directed graph similar to the well-known state diagram. Each vertex of the graph is labeled with the choice of player X. Each directed edge is labeled with a choice of player Y. So an edge labeled y_1 joining vertices x_1 to x_2 represents the fact that if player X is in the state indicated by the vertex labeled x_1 then it will choose move x_1 and if player Y makes a corresponding move of y_1 then player X next chooses move x_2.

Various strategies for playing tournaments of the Prisoner's Dilemma were examined by Axelrod and subsequently by Hofstadter. Some of the simplest (and most interesting) of them are finite-state. We give a few strategies taken from Hofstadter [5] along with their corresponding automata.

ALL D: Defect every time. (Figure 4.) ALL C is defined similarly.

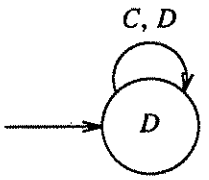

Figure 4

MRS: (Massive retaliatory strike.) Cooperate until Y defects, then defect every time. (Figure 5.)

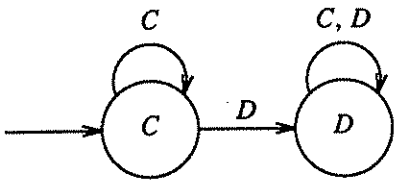

Figure 5

TIT FOR TAT: Cooperate on the first move, then mimic *Y*'s last move. (Figure 6.)

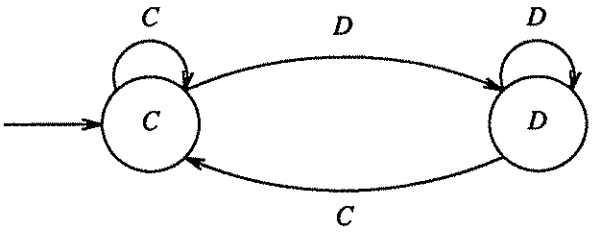

Figure 6

In computer tournaments TIT FOR TAT was found to be an extremely successful strategy. See Axelrod [3].

We also introduce another interesting strategy which seems intuitively reasonable.

TOT TIT FOR TAT: Cooperate if *Y* has cooperated at least as often as it has defected.

Note that this is not a good strategy in practice. An optimal defense against this strategy is to alternately cooperate and defect. We are interested in this strategy because it is not finite-state, but it can be modeled with a counter machine which uses the counter to keep track of the difference between the number of times *Y* cooperated and the number of times it defected.

In this paper we apply the techniques of Richards and Swart [7] for graph traversal and identification to attempt to solve the following problems:

1) Determine *Y*'s strategy, by playing a tournament as *X*,

2) Given *Y*'s strategy, produce an optimal strategy (or *defense)* for *X*.

We assume that each player is using a deterministic strategy. We assume that each strategy can be modeled by a transducer, specifically either a finite-state machine (FSM), a counter machine (CM), or a Turing machine (TM). The input to the transducer represents the *Y* move and the output represents the subsequent *X* move.

2. TERMINOLOGY

The following notation, similar to Trakhtenbrot and Barzdin' [9] is helpful. Each player's strategy produces a sequence of moves which may depend on its particular opponent Let the moves of player X be given by

$$X \text{ vs } Y = x_1, x_2, \cdots$$

and let the moves of player Y be given by

$$Y \text{ vs } X = y_1, y_2, \cdots.$$

Then for $i \geq 2$, x_i depends on $y_1, y_2, \cdots, y_{i-1}$, and y_i depends on $x_1, x_2, \cdots, x_{i-1}$. Suppose X and Y have each chosen their strategies. We define the *payoff sequence* $P(X \text{ vs } Y) = P_1(X \text{ vs } Y), P_2(X \text{ vs } Y), \cdots$, where $P_i(X \text{ vs } Y)$ is $P(x_i, y_i)$, i.e., the payoff to player X after the ith move in the tournament.

For convenience, we often identify players with their strategies.

Next we must determine what we mean by a good strategy for X playing in an infinite tournament with Y. Simply adding the total payoffs to X is inadequate since this sum is usually infinite. We suggest the following definition. Consider two strategies X_1 and X_2 against a given strategy for player Y. Then strategy X_1 *dominates* X_2 with respect to Y, $X_1 \geq_Y X_2$, if and only if there exists an $n_0 \geq 1$ such that for all $n \geq n_0$, $\sum_{i=1}^{n} P_i(X_1 \text{ vs } Y) \geq \sum_{i=1}^{n} P_i(X_2 \text{ vs } Y)$. This means that after some point, the total payoff to X_1 is never less than the total payoff to X_2. The reason for not demanding that each partial sum dominate is to allow initial sacrifices which eventually provide a larger payoff. A strategy X is *strongly optimal* with respect to Y if for every strategy X', $X \geq_Y X'$. As we shall show below, a strongly optimal strategy need not exist.

It is sometimes possible to prove another form of optimality. The (''worst-case average'') *value* of a strategy $V(X \text{ vs } Y) = \liminf_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} P_i(X \text{ vs } Y)$. Since each term in the partial sum is bounded above and below by values in the payoff matrix the averages of the partial sums are always bounded. If we did not take the infimum the limit may fail to exist; the average sums do not diverge but may oscillate between two finite values. We say that X is *value optimal* against Y if the value of X is at least as large as the value of any other strategy X' against Y.

In the following, when there is no chance of confusion, we shall refer to value optimal strategies simply as optimal.

3. STRATEGY INFERENCE

Suppose player X tries to determine player Y's strategy by playing a tournament against Y. This problem lies in the area of *inductive inference*, for which the fundamental reference is Gold [4]. A good survey of the field may be found in Angluin and Smith [1]. Strategy identification is very similar to the "black box identification" found in section five of Gold [4] and falls within the general model of identification suggested by section six of the same paper.

As is the case with graph traversal in Richards and Swart [7] there are some inherent limitations to any attempt at strategy identification.

1) *Cardinality*: There are uncountably many strategies. Furthermore, as we shall show, reasonable schemes for enumerating strategies embed non-computable functions, so no procedure can identify all strategies.

2) *Termination*: For any finite sequence of moves by Y there are infinitely many different continuations possible, even when Y's strategy is known to be finite-state. Therefore X can never know it has successfully determined Y's strategy.

3) *Reachability*: In general, Y's strategy may be represented by a directed (possibly infinite) graph with edges labelled by the opponent's moves. Unless this graph is strongly-connected it may be that no sequence of moves by X could explore all the states of Y's strategy. and therefore its identification would be impossible. We therefore consider it sufficient to identify one strongly-connected component of the strategy. This is essentially what Gold refers to as "weak learnability."

A very general approach for inferring a given strategy is the following. First we develop a procedure for generating a list of all strategies. (For example, for finite-state strategies it is possible to generate simpler strategies before more complex ones, according to the number of states used to describe the strategy.) These strategies are produced as needed below. We initialize by selecting the first strategy on the list as a candidate strategy S_c. We then repeat the following pair of steps forever.

Choose Alternate Strategy. Find the next strategy on the list which is consistent with the results of the tournament as played so far and which is distinguishable from S_c. Call this strategy S_a, the *alternate* strategy.

Eliminate One Strategy. Generate a finite distinguishing sequence of plays which will differentiate S_c from S_a. Run this sequence against the actual strategy to eliminate either S_c or S_a. Update S_c as necessary.

Certain conditions must hold for this procedure to be valid.

1. Possible strategies must be enumerable. This is true for strategies given by finite-state machines, counter machines and Turing machines.

2. A distinguishing sequence for non-equivalent machines must be found. If two machines are known to be non-equivalent, such a finite sequence can always be found in an "off-line" manner. For example, the algorithm could generate all sequences in lexicographical order and test them by simulating the effect of each sequence on the two machines.

3. The equivalence of two given strategies must be decidable. For games with only two possible moves we can immediately reduce this problem to the standard machine equivalence problem by considering one move to correspond to an accepting state, the other to a non-accepting state. The equivalence problem for finite-state machines is well known to be decidable. Valiant and Paterson [10] have shown that the equivalence problem for (deterministic) counter automata is decidable. The equivalence problem for Turing machines is well known to be undecidable.

This analysis gives us the following result.

Theorem 1. A (non-terminating) procedure exists for identifying a finite-state strategy or a counter machine strategy.

4. OPTIMAL DEFENSES

In this section we suppose the strategy of Y has been identified. What defense should X choose to play with Y? We have positive results when Y uses a finite-state strategy, and, not surprisingly, negative results when Y uses a Turing machine strategy. We have resolved a previous conjecture concerning counter machine strategies.

As we mentioned above, observe that a strongly optimal defense might not exist. For example, consider the following finite-state strategy for Y playing a Prisoner's Dilemma given in Figure 7. Y cooperates

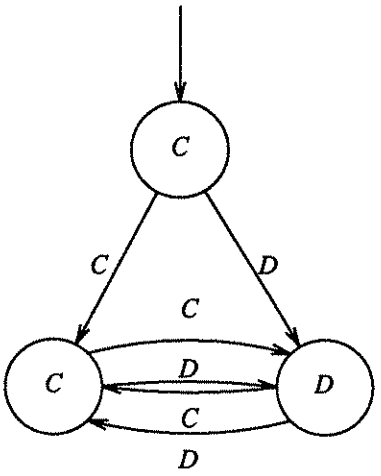

Figure 7.

on the first move and then alternates between cooperation and defection. The first choice of X determines Y's second move. Specifically, Y's second move repeats X's first move. After the first move Y's moves are independent of X's choices. X has only two reasonable strategies in the sense that any alternate strategy is dominated by one of these. They are:

X_1: Cooperate on move one then defect forever.

X_2: Defect forever.

The payoff sequences of the two strategies are:

$$P(X_1 \text{ vs } Y) = R,T,P,T,P, \cdots$$

$$P(X_2 \text{ vs } Y) = T,P,T,P,T, \cdots .$$

The conditions of a Prisoner's Dilemma give us $T > R > P$. Since $R > P$, X_1 dominates X_2 after every even move, and since $T > R$, X_2 dominates X_1 after every odd move. Therefore a strongly optimal defense does not exist.

However, we are able to show, using a pumping argument, that a value optimal defense exists against a finite-state strategy.

Theorem 2. A value optimal defense against an FSM strategy exists and is finite-state. Furthermore, an algorithm exists for determining an optimal defense if a description of the FSM strategy is given.

Proof: Suppose Y uses a finite-state strategy. Form the graph corresponding to Y's strategy. Give each edge in the graph weight equal to the payoff to X if that edge is chosen. Find a simple cycle which is reachable from the start state and which has the largest average payoff to X, i.e., find a cycle whose payoff per edge is

maximal. Note that there may be several such cycles. Let the average payoff of such a cycle be p. Define the strategy $\bar{X}$ to be one which takes Y to such a cycle and then stays in this cycle. Clearly the value of this strategy is p, and since the sequence of states generated is ultimately periodic, the strategy is finite-state. Next, let X' be any strategy against Y. Consider the set of states into which X' takes Y. At least one state, q, must occur infinitely often. So the behavior of the finite-state machine for Y consists of an initial sequence of moves ending in state q followed by an infinite sequence of (not necessarily distinct) cycles taking q to q. Since each of these cycles has an average payoff at most p, the value of this strategy $V(X' \text{ vs } Y)$ is at most p. Therefore, since $V(\bar{X} \text{ vs } Y) \geq V(X' \text{ vs } Y)$, $\bar{X}$ is value optimal. □

Next we consider counter machine strategies. Notice that an optimal defense against the CM strategy TOT TIT FOR TAT is $CDCDCD \cdots$, which is finite-state. Observations such as this led to the conjecture [8] that a value optimal defense against a CM strategy exists and is finite-state. We now give a counter-example to that conjecture along with a theorem that shows that nearly optimal finite-state defenses exist against CM strategies.

Recall that a counter machine is a pushdown automaton which has only one stack symbol, with the exception of a bottom-of-stack marker. (See Hopcroft and Ullman [6].) The machine can push the symbol on the stack (increment the counter), pop the symbol from the stack (decrement a non-zero counter) and check to see if the top stack symbol is the bottom-of-stack marker (if the count is zero). The CM either reads a symbol before each counter operation, or it makes an "ε-move", i.e., it changes state and performs an operation without reading any input.

We can use a generalization of the graph for finite-state strategies to represent CM strategies. Each vertex corresponds to the CM's states. A labelled vertex corresponds to the player's move in the game, as before. An unlabelled vertex corresponds to a state for an ε-move. Note that the input, the opponent's last move, does not change at an unlabelled vertex since the player has not made a move at this point. Our examples do not use ε-moves. An edge joining two vertices is labeled by a triple. The first component is the opponent's move, just as with a finite-state strategy. The second component is a test for zero value on the counter (0 zero, –0 not zero). The final component is the action to be taken on the counter (+ add one to the counter, – subtract one from the counter, 0 no action).

Using this notation, consider the CM strategy for the Prisoner's Dilemma for player Y given by Figure 8. All unspecified edges take Y to an ALL D state.

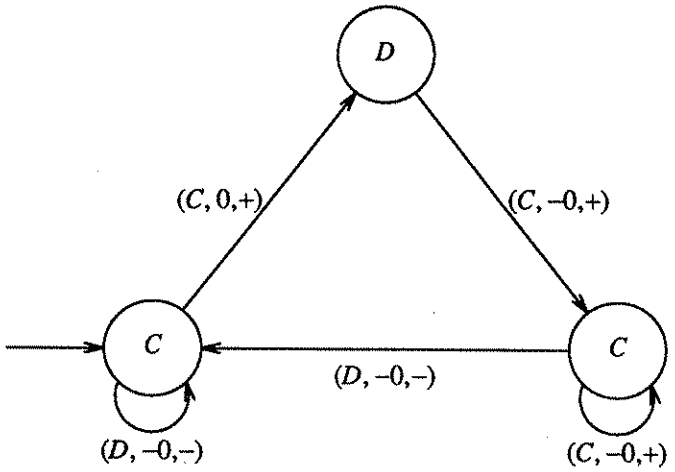

Figure 8

A simple interpretation of Y's strategy can be given. Whenever the counter is zero Y cooperates once and then defects once, incrementing the counter both times. If X has cooperated on both these moves then Y continues to cooperate; at this point Y expects X to cooperate and it increments the counter as long as X does. When X defects Y continues to cooperate but decrements the counter. Y expects X to continue to defect until the counter is zero. When the counter again becomes zero Y begins this action all over again.

We make the following observations about the possible strategies for X playing against this particular Y. Intuitively, X must periodically make a small initial sacrifice (by cooperating when it knows Y will defect) whenever the counter changes from 0 to 1. Formally we assert:

(1) If X cooperates n times and then defects n times it receives a payoff $R+S+(n-2)R+nT$. In this sequence of $2n$ moves X receives an average payoff of $\frac{R+T}{2} - \frac{R-S}{2n}$. Similarly the defense $C^2D^2C^3D^3C^4D^4\cdots$ has a value of $\frac{R+T}{2}$. (Note that X only achieves the value of its defense in the limit since it must make infinitely many (but widely spaced) one-move sacrifices.)

(2) We say a defense for X is *canonical* if it is of the form $C^{i_1}D^{i_1}C^{i_2}D^{i_2}C^{i_3}D^{i_3}\cdots$, where each $i_j \geq 2$. X can choose to adopt ALL C at any point when Y expects it to cooperate, and X will have a defense with value R. Any other deviation from the canonical defense will cause Y to play ALL D at some point; hence X will have a defense with a value of only P.

(3) Consequently any defense for X which is not canonical has value less than $\frac{R+T}{2}$ and is therefore not optimal. Conversely, any canonical defense in which $\limsup_{j\to\infty} i_j = \infty$ is optimal.

(4) For any finite-state defense there is a maximum value n such that $C^n D^n$ is part of the canonical defense, and therefore its value is at most $\frac{R+T}{2} - \frac{R-S}{2n}$, and it is not optimal.

Less obvious is the fact that any optimal defense to Y's strategy is not a CM strategy. The intuition behind this result is that X must generate longer and longer sequences and also must know when Y has a zero counter; a single counter cannot do this simultaneous bookkeeping.

Lemma 1. A CM with constant input has a fixed bound (depending only on its number of states) on the number of steps from one zero counter to the next zero counter.

Proof: Define a configuration to be state and counter value pair. It is easy to show that there exists such a bound on the number of steps, such that after that many steps either the CM will loop through a set of configurations or the CM will loop with the counter growing without bound. In either case there is a contradiction. □

Theorem 3. There exists a CM strategy for which no optimal defense is a CM strategy.

Proof: We show that the strategy in Figure 8 is such a strategy. Assume that X has an optimal CM strategy against Y. As discussed above X must play along with Y and so must produce, for arbitrary p, a sequence of more than p C's before switching to a sequence of more than p D's and then switching back to a C, and throughout the input is constant (after the second C). The CM needs some internal cue to switch its output since the input is constant. First, notice that for p large enough the CM must produce a zero counter during the C's since, otherwise, by using a pumping argument, it follows that the CM's behavior must loop and it will not change to a sequence of D's. Similarly during the sequence of D's a zero counter must occur. Second, by the same reasoning used in the proof of Lemma 1, there is a fixed bound on the number of moves between the last time the counter is zero in a sequence of D's and the next time that X chooses a C. This follows because with constant input there is a bound on the number of distinct configurations that the CM can attain before reaching the configuration in which it next chooses to cooperate. Hence the CM will produce a series of zero counters while the input is kept constant. At least one of these zero counters occurs on a C choice and one occurs a bounded distance from the end of the D choices.

There are two cases. If for all p it produces a number of zero counters that is less than or equal to the number of states then, for large enough p, we can find a pair of consecutive zero counter configurations separated by a number of steps larger than any fixed bound. This contradicts Lemma 1. Therefore, for some

p, it produces more zero counters than states. In this case some state must repeat in the zero counter configuration. Therefore, the CM must loop. Its looping behavior may be all C's, all D's, or a combination of the two, but in all cases it does not play a canonical defense. □

Recall that X can get as close as it wishes to the optimal value of this game by simply choosing a large enough n and playing the finite-state strategy $C^nD^nC^nD^n\cdots$. This situation generalizes to the following theorem.

Theorem 4. Suppose player Y uses a CM strategy and suppose X has a value optimal defense against Y with value v. Let $\varepsilon>0$ be given. Then X has a finite-state strategy against Y with value at least $v-\varepsilon$.

Proof: Suppose X plays its value optimal defense against Y. Then Y's moves can be described by a sequence of configuration pairs (s_i, h_i), where s_i is the state of Y and h_i is the height of the counter. There are two cases to consider.

a) $h_i=0$ infinitely often (i.e., the counter is zero infinitely often). Since the CM has only finitely many states there must be some state, say s_i such that the pair $(s_i, 0)$ occurs infinitely often. Therefore, the sequence of Y's moves can be partitioned into a finite initial segment and infinitely many segments, each of finite length and each starting with $(s_i, 0)$ and ending just before the next $(s_i, 0)$. See Figure 9. We claim that at least one of these segments has an average payoff per move of at least $v-\varepsilon$. If every segment had average payoff strictly less that $v-\varepsilon$ then by adding all the payoffs of all the segments we would get a value for X's optimal strategy which is bounded above by $v-\varepsilon$, contrary to hypothesis. Suppose that the kth segment has value at

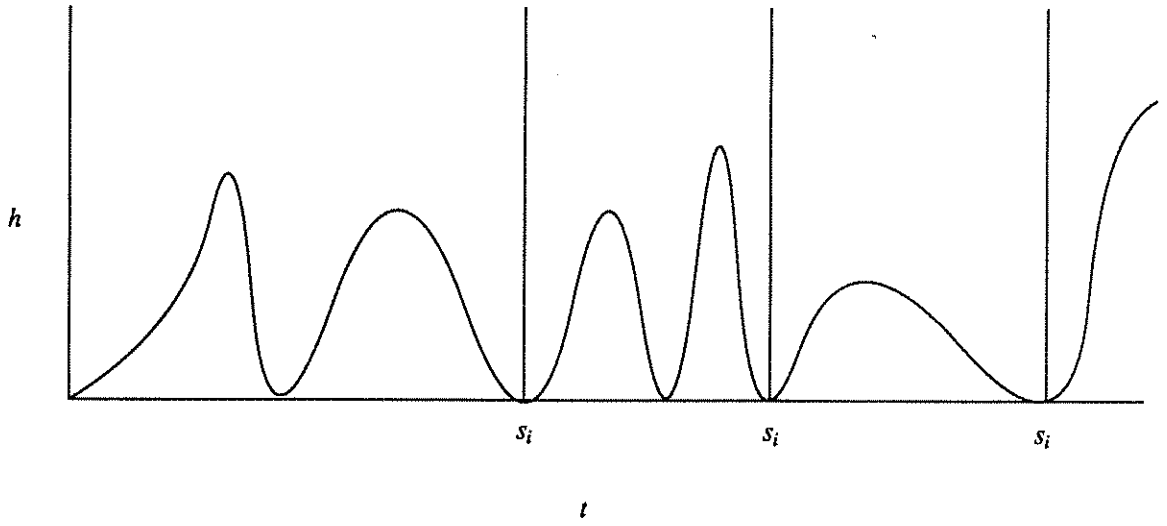

Figure 9

least $v-\varepsilon$. Then by choosing a strategy for X which is the same as the value optimal one for the first $k-1$ segments and then repeatedly chooses the moves which produced segment k, we get a defense which has value at least $v-\varepsilon$. Since this strategy is ultimately periodic, it is finite-state.

b) $h_i=0$ finitely often. A similar argument works. There must be some state s_i and some infinite subsequence of configurations which begins somewhere after the last time the counter is 0, (s_i, g_0), (s_i, g_1), $\cdots$ such that $g_j \le g_{j+1}$ for each j. See Figure 10. Again, the sequence of Y's moves can be partitioned into a finite initial segment and infinitely many segments, each of finite length, starting with (s_i, g_0), (s_i, g_1), $\cdots$. As before, at least one of these segments, say the kth, has value at least $v-\varepsilon$. Choose a strategy for X which is the same as the value optimal one for the first $k-1$ segments and then repeatedly chooses the moves which produced segment k. Notice that segment k started with the counter at value g_{k-1} and ended with value g_k and that at no time did the counter become zero while Y was playing in this segment. When the moves which produced segment k are repeated, the new segment $k+1$ starts with the counter at value g_k and ends with the counter at value $2g_k-g_{k-1}$. Since $g_{k-1} \le g_k$, the initial counter value at the start of segment $k+1$ is at least as large as the initial counter at the start of segment k, so all moves in the new segment have counter values at least as large as corresponding moves in the kth segment. Hence the counter is never zero in segment $k+1$. So the behavior of Y in playing segment $k+1$ is identical to its behavior in segment k. In particular the average payoff per move is the same. Continuing this sequence of moves gives us an ultimately periodic strategy of value at least $v-\varepsilon$. □

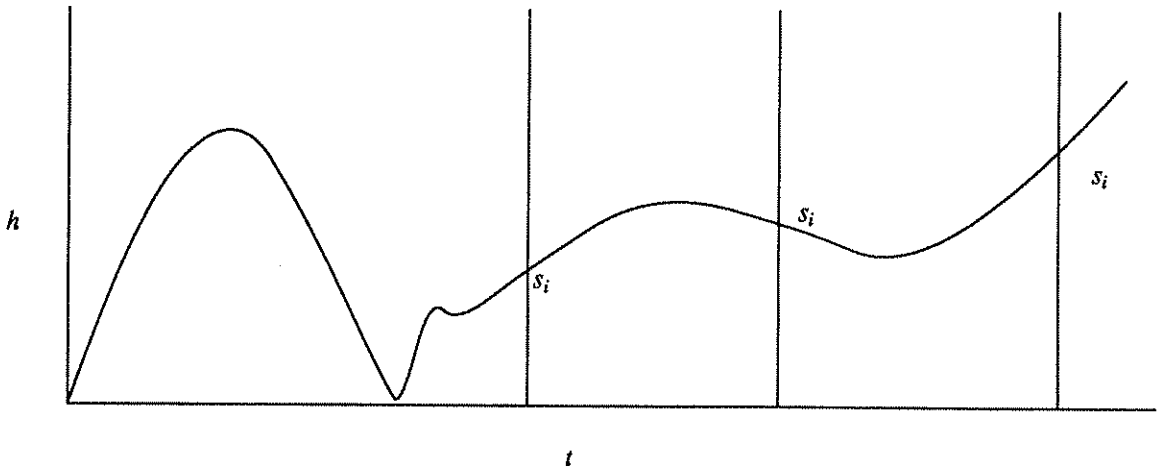

Figure 10

Finally, we consider general Turing machine strategies. We extend the technique used for FSM strategies to show that a general TM strategy need not have a value optimal defense, and then show that even if a value optimal or strongly optimal defense exists finding it is undecidable.

Theorem 5. There is a TM strategy which has neither a strongly optimal nor a value optimal defense.

Proof: Consider the following strategy for Y which after an initial section follows a fixed loop. Initially Y defects and continues to defect as long as X cooperates. Once X defects Y switches to its fixed loop. The loop consists of n C's followed by one D, where n is the number of initial cooperations X made.

If X never defects the value of its defense is just S. The other defenses can be put into equivalence classes according to n. Clearly each equivalence class is dominated by the defense $C^n D^\infty$. The value of each of these defenses is $\frac{n}{n+1} T + \frac{1}{n+1} P$. Therefore X can choose from any one of an infinite sequence of defenses with increasing value, but no single value optimal or strongly optimal defense exists. □

Our final results in this area show that it hopeless to work with TM strategies, as might be expected.

Theorem 6. There is no algorithm which, given a description of a TM strategy, produces a value (or strongly) optimal defense, even if it is known that such a defense exists.

Proof: We show that the existence of such a strategy would imply the solution to the blank tape halting problem. Consider the following Turing machine strategy for player Y, which uses the description of an arbitrary Turing machine H as additional input and plays a Prisoner's Dilemma tournament. Y cooperates on the first play. If X cooperates Y plays TIT FOR TAT. If X defects Y uses a scratch tape to simulate H when started on a blank tape. Y defects once for each step H makes. If H halts then Y cooperates in all future plays. Again, X has two reasonable strategies, completely determined by its first move. If X chooses to cooperate on its first move, Y will play TIT FOR TAT. The optimal defense to this strategy is to cooperate forever, (ALL C), so if X chooses this its payoff sequence is $R, R, R, \cdots$ If, however, X defects on its first move, then Y's future moves are independent of X's, so the best strategy for X is ALL D. This strategy has payoff $T, P, P, \cdots$ if H never halts since then Y will play ALL D forever. The same strategy has payoff $P, P, \cdots P, T, T, T, \cdots$ with n P's if H halts after n moves, since after n defections, Y will cooperate forever. So the value of ALL C is R and the value of ALL D is either P if H never halts or T if H halts. Recall that $P < R < T$ so that ALL D is optimal if H halts and ALL C is optimal if H doesn't halt. □

Corollary 1. It is undecidable whether an optimal defense exists for a given TM strategy.

Proof: Augment the construction in the preceding proof by not having Y play ALL C when H halts. Instead have Y play the strategy in the proof of Theorem 5 when H halts. So if H does not halt there is an optimal defense for X with value R. If H does halt then there is no optimal defense, but instead a sequence of strategies with values approaching T. □

5. COMBINING IDENTIFICATION WITH OPTIMAL DEFENSE

In Richards and Swart [7] we combined graph identification with graph traversal. A similar situation occurs in the case of game strategies. When methods exist for identifying a strategy and computing an optimal defense, then in some sense it is possible to do both simultaneously. The basic idea is straightforward. Modify the identification procedure to alternate between identification and defense playing. Every time the identification procedure finishes one pass through the steps *Choose Alternate Strategy* and *Eliminate One Strategy* (i.e., it has either reconfirmed the old candidate strategy or selected a new candidate strategy) it computes the optimal defense against the candidate strategy and spends some time playing that strategy. Since the procedure eventually identifies the correct strategy, it eventually plays the correct defense. The major difficulty with this approach is assuring that more moves are spent playing than testing. For instance, it is conceivable that each new test takes more moves than all the previous moves made up to that test. In some cases one can prevent this by predicting the maximum length of the next distinguishing sequence and playing long enough in advance. By careful prediction one can assure that any desired fraction (less than one) of the time is spent playing the defense and that in the limit this fraction can be made to go to one.

In the case of finite-state strategies, this goal is achievable. Suppose M_c, the automaton for the candidate strategy S_c has n states and that M_a, the automaton for the next consistent, distinguishable alternate strategy has m states. Before playing the optimal defense against strategy S_a, for each pair (i, j) we compute a sequence which distinguishes M_c starting in state i from M_a starting in state j. Whenever we finish playing we know that one of these sequences will distinguish S_c from S_a. Therefore, the longest of these nm sequences gives us a uniform upper bound on our next test sequence. We can use this bound to determine how long to play our defense before resuming testing.

6. REFERENCES

1. D. Angluin and C. H. Smith, Inductive Inference: Theory and Methods, *Computing Surveys*, **15**(3), September 1983, pp. 237-269.

2. R. Axelrod and W. D. Hamilton, The Evolution of Cooperation, *Science*, **211**, 27 March 1981, pp. 1390-1396.

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

4. E. M. Gold, Language Identification in the Limit, *Information and Control*, **10**, 1967, pp. 447-474.

5. D. R. Hofstadter, Metamagical Themas, *Scientific American*, May, 1983, pp. 18-26.

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

7. D. Richards and C. Swart, Universal Traversal Sequences, Graph Traversal and Graph Identification, in *Combinatorics on Words*, L. J. Cummings (ed.), Academic Press, Toronto, 1983, 387-405.

8. C. Swart and D. Richards, Identification of Strategies for Two-Person Games, *Congressus Numerantium*, **45**, 1984, pp. 193-205.

9. B. A. Trakhtenbrot and Y. M. Barzdin', *Finite Automata Behavior and Synthesis*, North-Holland, Amsterdam, 1973.

10. L. G. Valiant and M. S. Paterson, Deterministic One-Counter Automata, *Journal of Computer and System Sciences*, **10**, 1975, pp. 340-350.