Optimality and Domination in Repeated Games with Bounded Players

Lance Fortnow*
Duke Whang†
Department of Computer Science
The University of Chicago
1100 East 58th Street
Chicago, Illinois 60637

Abstract

We examine questions of optimality and domination in repeated stage games where one or both players may draw their strategies only from (perhaps different) computationally bounded sets. We also consider optimality and domination when bounded convergence rates of the infinite payoff. We develop a notion of a "grace period" to handle the problem of vengeful strategies.

1 Introduction

Consider the Prisoner's Dilemma game: Two prisoners have to decide independently of each other whether to cooperate as friends and refuse to talk to police, or whether to defect from their friendship, admit guilt and rat on their fellow prisoner. If both players cooperate then they can both walk free. If one defects while the other cooperates the first goes free with a small reward while the other gets sent to jail. If both defect they both get sent to jail but with small rewards each. We describe a payoff matrix for this game in Section 2.1.

In this game, defection is a dominant strategy: Whatever the second prisoner decides to do, the best strategy for the first player is to defect. Unfortunately this leads to two rational players both defecting instead of the clearly advantageous scenario where they both cooperate.

Now look at the game where Prisoner's Dilemma is repeated k times and the payoff is the average of the payoffs of the individual game. Always defecting is no longer a dominant strategy though one can show that it is the only strategy that will be played by rational players.

In our paper we will be concerned with Prisoner's Dilemma played an infinite number of times where, for example, the payoff is the limit (actually lim inf) of the average of the first k rounds. In this game, not only is always defecting not a dominant strategy, but it is possible for two rational players to always cooperate.

We will be interested in issues of optimality and domination. Roughly a strategy for Player I is optimal if it is the best possible strategy against a specific strategy of Player II. A strategy is dominant if it is the best strategy for Player I no matter what strategy Player II uses.

In particular we will consider games where the sets of possible strategies for Players I and II are limited in a computational sense. Previous work has looked at limiting the players to strategies implemented by finite automata of size less than a given bound. See Kalai [Kal90] for a survey on "Bounded Rationality" for repeated games.

However, finite automata are a very weak form of computation. In the real world human players are often armed with powerful computers to help them optimize their strategies in business and financial games. We would like to look at computational models that better capture what games computers can play. Computer scientists generally use polynomial-time computation by a Turing machine as a good rough model of tractable computation.

Thus we will study four sets of strategies: the set of all possible strategies, the set of all strategies realized by arbitrary Turing machines, the set of all strategies realized by polynomial-time Turing machines and the

*Email: fortnow@cs.uchicago.edu. Partially supported by NSF grant CCR 92-53582.
†Email: whang@cs.uchicago.edu. Partially supported by NSF grant CCR 92-53582 and a NDSEG graduate fellowship, grant DAAH04-93-G-0273.

set of all strategies realized by finite automata. We will also look at behavioral (probabilistic) versions of these strategies. These strategies are formally defined in Section 2.5.

Gilboa and Samet [GS89] first looked at the question of domination in the bounded rationality model. They show that when Player II is limited to strategies realized by strongly connected finite automata (there is a path from any state to any other state) there exists a recursive dominant strategy.

Gilboa and Samet need the strong connected condition to protect against "vengeful" strategies, i.e., strategies of Player II that may penalize Player I forever simply because Player I chose a certain action early in the game. We do not wish to limit ourselves in this way. In particular there may not be such a simple characterization of nonvengeful polynomial-time Turing machines.

In order to reproduce the Gilboa-Samet result for arbitrary finite automata we need to weaken our notion of domination. We define a notion of *eventually dominant strategy* where a strategy of Player I is only required to be dominant among all other strategies of Player I that agree with Player I's actions for some initial finite number of rounds. These initial rounds form a sort of grace period that allows Player I to learn enough about Player II to start playing in a dominant manner. Complete definitions of eventually optimal and dominant can be found in Section 2.4.

We then extend the result of Gilboa and Samet to show that there exists an eventually dominant recursive strategy against strategies realized by any finite automaton (Theorem 3.2). Gilboa and Samet's result follows from ours since every eventually optimal strategy is optimal against strongly connected automata.

A polynomial-time strategy could achieve dominance in this model by using a sneaky trick where it saves up its time in previous rounds for use in future rounds. It achieves dominance but not for a very large number of rounds.

Thus we would like to also look at bounding the number of rounds. While it may seem counterintuitive to bound the number of rounds of an infinite game, we can achieve this effect by bounding the rate of convergence of the lim inf in the payoff function. We formally describe this approach in Section 2.6.

Gilboa and Samet's result and our extension require an exponential number of rounds. We show that this bound is tight (Theorem 3.3).

However, if we look at the Matching Pennies game (as described in Section 2.1), the situation differs. Since the best payoff for Player I is the same no matter what Player II plays, Player II is unable to punish (or reward) Player I. This punishing factor plays an important role in the proof of the exponential lower bound. We show that there exists a dominant polynomial-time strategy against any finite automaton in the Matching Pennies game which uses only a polynomial number of rounds (Theorem 3.4). However, there exists a strategy implemented by a probabilistic finite automaton for Player II for which there is no optimal strategy for Player I (Theorem 3.5).

Finally we consider the case where Player II is now limited to strategies realized by polynomial-time machines. For Prisoner's Dilemma, we show a very strong negative result: There exists such a strategy for Player II for which for all ϵ, $0 \leq \epsilon < 1$, there is no eventually ϵ-optimal recursive strategy for Player I (Theorem 3.7). This result extends work of Knoblauch [Kno94].

For Matching Pennies we show that there does exist a dominant recursive strategy against polynomial-time strategies (Theorem 3.8).

We also show similar results for discounted games where the payoffs in future rounds are multiplied by some discount factor over the current rounds. In particular, we study optimal and dominant behavior as the discount factor approaches one.

In Section 5 we discuss some further directions that we would like to see explored.

2 Definitions and Preliminaries

We will use the standard definitions of finite automata and Turing machines. See Hopcroft and Ullman [HU79] for formal definitions. Garey and Johnson [GJ79] give a good introduction to the notion of polynomial-time algorithms and which problems may or may not have such solutions.

We use Z to represent the integers and Z^+ to represent the positive integers.

2.1 Stage games

Let us consider two person games with players denoted by I and II. A *stage game* G consists of two finite and nonempty *action sets* S_{I} and S_{II} and two *payoff functions* $u_{\mathrm{I}}, u_{\mathrm{II}} : S \rightarrow Z$ where $S = S_{\mathrm{I}} \times S_{\mathrm{II}}$.

We will consider two particular games in this paper. The first is known as Prisoner's Dilemma and has the following payoff matrix:

	C	D
c	$(3,3)$	$(0,4)$
d	$(4,0)$	$(1,1)$

Throughout this paper we will use the convention of lower case letters for Player I's actions and upper case letters for Player II's actions. The first number of a matrix entry is the utility of Player I where the second number is the utility of Player II. For example, if Player I plays c and Player II plays D then Player I receives zero utils (units of utility) and Player II receives four.

Note that if Player II defects (D) then the maximum score achievable by Player I is one but if Player II cooperates (C) then Player I could have achieved a score of four. These payoffs allow Player II to reward (or punish) Player I based on earlier play. In fact all of our results about Prisoner's Dilemma will hold for games where if $S_1 = \{a_1, a_2\}$ and $S_2 = \{A_1, A_2\}$ then

$$\max(u_{\mathrm{I}}(a_1, A_1), u_{\mathrm{I}}(a_2, A_1)) \neq$$

$$\max(u_{\mathrm{I}}(a_1, A_2), u_{\mathrm{I}}(a_2, A_2)).$$

The other game we consider is Matching Pennies which has the following payoff matrix:

	H	T
h	$(1,-1)$	$(-1,1)$
t	$(-1,1)$	$(1,-1)$

In this game whether Player II plays heads (H) or tails (T) Player I could achieve the score of one. If Player I is somehow able to predict Player II's moves then Player II would have no way to reward or punish earlier behavior of Player I. All of our results about Matching Pennies will hold for nontrivial games where if $S_1 = \{a_1, a_2\}$ and $S_2 = \{A_1, A_2\}$ then

$$\max(u_{\mathrm{I}}(a_1, A_1), u_{\mathrm{I}}(a_2, A_1)) =$$

$$\max(u_{\mathrm{I}}(a_1, A_2), u_{\mathrm{I}}(a_2, A_2)).$$

In Matching Pennies, we say that Player I *wins* in a single round if Player I plays the same action as Player II and Player I *loses* if Player I and Player II play different actions.

2.2 Repeated stage games

In this paper we will be interested in games consisting of a stage game repeated infinitely often.

A *history* of player $i \in \{\mathrm{I}, \mathrm{II}\}$ of the first k rounds is a k-tuple of actions from S_i. We denote the empty history by Λ. The play of the game depends on the *strategy* $\sigma_i : H_{\mathrm{I}} \times H_{\mathrm{II}} \to S_i$ of each player where H_i is the of all finite histories of player i.

Fixing σ_{I} and σ_{II}, we can inductively define for each player i the action $a_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})$ at round k and the history $h_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})$ after k rounds. We will drop the dependencies on σ_{I} and σ_{II} when they can be inferred from context.

For each i, let $h_i^0 = \Lambda$.

For each i and $k \geq 0$, let $a_i^{k+1} = \sigma_i(h_{\mathrm{I}}^k, h_{\mathrm{II}}^k)$.

For each i and $k \geq 0$, let h_i^{k+1} be the $k+1$ tuple whose first k coordinates are h_i^k and whose $k+1$st coordinate is a_i^{k+1}.

A *repeated stage game* consists of a stage game G, and two sets of strategies, Σ_{I}^G and Σ_{II}^G which are sets of strategies from S_{I} and S_{II} respectively. Each player i will choose one strategy σ_i from Σ_i^G.

We will consider two models for the *payoff functions* π_{I} and π_{II} for these games. In the *limit of the means* games denoted G^∞ we define $\pi_i^{G^\infty}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) =$

$$\liminf_{k \to \infty} \frac{1}{k} \sum_{j=1}^{k} u_i(a_{\mathrm{I}}^j(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}), a_{\mathrm{II}}^j(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})). \qquad (1)$$

In the *discounted* game with discount δ ($0 < \delta < 1$) denoted G^δ we define $\pi_i^{G^\delta}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) =$

$$(1-\delta) \sum_{k=1}^{\infty} \delta^{k-1} u_i(a_{\mathrm{I}}^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}), a_{\mathrm{II}}^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})). \qquad (2)$$

In general we will be interested in the discounted games as δ approaches one.

2.3 Optimal and Dominant Strategies

Fix a strategy σ_{II} in Σ_{II}^G.

We will define optimality separately for the limit of the means games G^∞ and the discounted games G^δ.

For G^∞: A strategy σ_{I} is *optimal* if for every strategy $\sigma_{\mathrm{I}}' \in \Sigma_{\mathrm{I}}^G$

$$\pi_{\mathrm{I}}^{G^\infty}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) - \pi_{\mathrm{I}}^{G^\infty}(\sigma_{\mathrm{I}}', \sigma_{\mathrm{II}}) \geq 0. \qquad (3)$$

For G^δ we are interested in optimal behavior as δ approaches one: A strategy σ_{I} is *optimal* if for every strategy $\sigma_{\mathrm{I}}' \in \Sigma_{\mathrm{I}}^G$

$$\liminf_{\delta \to 1^-} (\pi_{\mathrm{I}}^{G^\delta}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) - \pi_{\mathrm{I}}^{G^\delta}(\sigma_{\mathrm{I}}', \sigma_{\mathrm{II}})) \geq 0. \qquad (4)$$

We define a strategy to be *ϵ-optimal* by replacing 0 by $-\epsilon$ in Equations (3) and (4).

A strategy σ_{I} is *dominant* if for every choice of strategy σ_{II} from Σ_{II}^G, the strategy σ_{I} is optimal. Likewise we can define *ϵ-dominant* strategies.

2.4 Handling Vengeful Strategies

For a robust set of strategies Σ_{II}^G there may not exist any dominant strategies σ_{I} because of the existence of "vengeful" strategies. Loosely, a vengeful strategy σ_{II} may decide to punish Player I infinitely often based on some action Player I made early in the game.

Gilboa and Samet [GS89] handle vengeful strategies by removing these strategies from the set of strategies under consideration by studying only strongly connected finite automata for Σ_{II}^G.

We will use a different approach. Instead of restricting the types of strategies, we will use a modified definition of domination called *eventually dominant.* This definition allows Player I a "grace period" where he can learn about σ_{II}. We then only require that σ_{I} be optimal among all the strategies in σ_{I} that perform the same during that grace period.

Formally two strategies σ_{I} and σ_{I}' in Σ_{I}^G are *consistent after k rounds* if $h_{\mathrm{I}}^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) = h_{\mathrm{I}}^k(\sigma_{\mathrm{I}}', \sigma_{\mathrm{II}})$. Note that all pairs of strategies in Σ_{I}^G are consistent after zero rounds.

A strategy σ_{I} is *optimal after k rounds* if σ_{I} is optimal where we restrict the choices of σ_{I}' to those strategies in Σ_{I}^G that are consistent with σ_{I} after k rounds.

A strategy σ_{I} is *eventually optimal* if there exists some k such that σ_{I} is optimal after k rounds.

We can similarly define *ϵ-optimal after k rounds* and *eventually ϵ-optimal.*

A strategy σ_{I} is *eventually dominant* if for every choice of strategy σ_{II} from Σ_{II}^G, the strategy σ_{I} is eventually optimal. Likewise we can define *eventually ϵ-dominant* strategies.

2.5 Bounded Strategies

Usually game theorists allow Σ_{I}^G and Σ_{II}^G to be any strategy available to a rational player. Thus we will let the *rational strategies* be the set of all possible strategies σ_i mapping $H_{\mathrm{I}} \times H_{\mathrm{II}}$ to S_i.

In this paper we would like to look at smaller sets of strategies induced by computation devices of possibly limited resources. First we look at the *regular strategies*, the set of strategies realizable by finite automata. See Gilboa and Samet [GS89] for a formal definition. However we will *not* require the automata to be (strongly) connected. For an example of a finite automaton that implements a regular strategy for Prisoner's Dilemma see Figure 1.

In this paper we would also like to look at strategies implemented by Turing machines. We call the *recursive strategies* those strategy functions computed by some Turing machine that halts on all inputs.

Finally, we define the class of *polynomial-time strategies* as the set of all strategy functions computed by a polynomial-time Turing machine. Since the length of a history is the number of the current round, the set of polynomial-time strategies is the set of strategies realized by some Turing machine that uses $p(r)$ time between rounds r and $r+1$ for some polynomial p.

Note that the set of rational strategies strictly contains the set of recursive strategies which strictly contains the set of polynomial-time strategies which strictly contains the set of regular strategies. There is an uncountable number of rational strategies but only a countable number of recursive, polynomial-time and regular strategies.

In some cases, we would like to allow behavioral (probabilistic) strategies. Behavioral versions of the machines above allow the machines to move according to a distribution over states instead of to a single state during each step of computation. In keeping with the spirit of computation, we require rational probabilities for these distributions. For an example of a probabilistic finite automaton that implements a behavioral regular strategy see Figure 3.

2.6 Bounding the number of rounds

One interpretation of the limit of the means payoff says that the average value of each finite initial segment of the game should approach the final payoff of the game. Since the computation time of the game will concern us in this paper, we would like to require a reasonable rate of convergence.

Fix a game G^∞. We define the average payoff function

$$\mu_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) = \frac{1}{k} \sum_{j=1}^{k} u_i(a_{\mathrm{I}}^j(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}), a_{\mathrm{II}}^j(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})).$$

By Equation (1) we have

$$\pi_i^G(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) = \liminf_{k \to \infty} \mu_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}).$$

From the definition of lim inf we have that for all $\alpha > 0$ there exists some round t such that for all rounds $k \geq t$,

$$\mu_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) \geq \pi_i^G(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) - \alpha.$$

To bound the number of rounds, we will require that t be a function of α and on the "size" of the strategy for σ_{II}.

Assign a size function $s : \Sigma_{\mathrm{II}}^G \to Z^+$. For example for a regular strategy σ_{II}, let $s(\sigma_{\mathrm{II}})$ be the number of states of the smallest finite automaton that realizes σ_{II}. For behavioral strategies, the size should bound the description of the probabilities as well as the number of states.

Fix a strategy σ_{II} from Σ_{II}^G. A strategy σ_{I} *converges in $t(m)$ rounds* if for $\alpha > 0$ and $k \geq t(s(\sigma_{\mathrm{II}})/\alpha)$,

$$\mu_i^k(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) - \pi_i^G(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) \geq -\alpha. \tag{5}$$

A behavioral strategy σ_{I} converges in $t(m)$ rounds if the expected value of the left hand side of Equation (5) is at least $-\alpha$.

For discounted games, we will use a different approach: We can use the value of δ to bound the number of rounds. As δ approaches 1, Player I can use more rounds for exploration without affecting the total payoff by too much. We can formalize this notion by bounding the growth of the lim inf in Equation (4).

For a game G^{δ} and a strategy $\sigma_{\mathrm{II}} \in \Sigma_{\mathrm{II}}^{G}$, we say that a strategy σ_{I} *converges after* $t(m)$ *rounds* if for all $\alpha > 0$ and for all $\delta \geq 2^{-1/t(s(\sigma_{\mathrm{II}})/\alpha)}$,

$$\pi_{\mathrm{I}}^{G^{\delta}}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) - \pi_{\mathrm{I}}^{G^{\delta}}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) \geq -\alpha. \qquad (6)$$

Once again, a behavioral strategy σ_{I} converges in $t(m)$ rounds if the expected value of the left hand side of Equation (5) is at least $-\alpha$.

3 Main Results

In this section we describe the main results of our paper. All of these results hold for both the limit of the means and discounted payoffs. Proof sketches for these results can be found in Section 4.

The first theorem is a reformulation of a theorem by Gilboa-Samet [GS89]. They proved the existence of a single algorithm which will become dominant against the set of strategies realized by strongly connected finite automata. (A finite automaton is strongly connected if there is a directed path from any state to any other state.)

Theorem 3.1 (Gilboa-Samet) *For any game G, there is a dominant recursive strategy σ_{I} for the class of rational strategies against strategies realized by strongly connected finite automata.*

The next theorem extends this result and states that there is an eventually dominant recursive strategy against all strategies realized by finite automata whether strongly connected or not. Note that we need to use "eventually dominant" because of vengeful automata, but for strongly connected finite automata the notions of optimality and of eventual optimality coincide.

Theorem 3.2 *For any game G, there is a recursive strategy σ_{I} which is eventually dominant for the class of rational strategies against the class of strategies realized by finite automata.*

In the proofs of Theorems 3.1 and 3.2 the Turing machine computing the strategy uses exponential time per round and converges in an exponential number of rounds. One can use a delaying tactic to create a machine that runs in polynomial time by spending an exponential number of rounds just playing simply and then using the fact that polynomial time in an exponential history yields exponential time computation.

However, what if we require the strategy in Theorem 3.2 to converge in a polynomial number of rounds? The answer depends on the type of game.

For Prisoner's Dilemma, we prove a lower bound for the number of rounds needed for *any* strategy to become eventually dominant over regular strategies. This lower bound is exponential in the number of states of the minimal automaton that implements the regular strategy.

Theorem 3.3 *Consider Prisoner's Dilemma and fix any ϵ, $0 \leq \epsilon < 1$ and any n. If σ_{I} is any strategy for Player* I *there exists some rational strategy σ_{II} of Player* II *implemented by a finite automaton of n states such that any strategy σ_{I} ϵ-optimal against σ_{II} will require an exponential number of rounds to converge.*

However, for the Matching Pennies game, the situation changes dramatically:

Theorem 3.4 *For the Matching Pennies game, there exists a polynomial-time strategy that dominates all finite automata and converges in a polynomial number of rounds.*

However, against probabilistic finite automata, domination and even optimality cannot be achieved:

Theorem 3.5 *There exists a behavioral regular strategy for which there is no optimal rational strategy even for Matching Pennies.*

For Prisoner's Dilemma, Vicki Knoblauch [Kno94] creates two recursive strategies $\sigma_{\mathrm{II}}^{(1)}$ and $\sigma_{\mathrm{II}}^{(2)}$ where there is no optimal rational strategy for $\sigma_{\mathrm{II}}^{(1)}$ and there is an optimal rational strategy but no optimal recursive strategy for $\sigma_{\mathrm{II}}^{(2)}$.

Now we consider the situation if σ_{II} can be implemented by a polynomial time algorithm. For Prisoner's Dilemma, we get very strong negative results that extend the techniques and results of Knoblauch.

Theorem 3.6 *For Prisoner's Dilemma, there is a polynomial-time strategy σ_{II} for which there is no eventually optimal rational strategy σ_{I}.*

For any $\epsilon > 0$ and any strategy σ_{II}, there is clearly some rational ϵ-optimal strategy σ_{I}. However, we show that in general σ_{I} will not be recursive.

Theorem 3.7 *For Prisoner's Dilemma, there is some polynomial-time strategy σ_{II} such that there is an optimal rational strategy but for all ϵ, $0 \leq \epsilon < 1$, there is no eventually ϵ-optimal recursive strategy for the class of rational strategies.*

However, for Matching Pennies, the situation changes dramatically.

Theorem 3.8 *For Matching Pennies games, there is a recursive strategy σ_{I} which is dominant for the class of rational strategies against the class of polynomial time strategies.*

In general for Matching Pennies, if $\mathcal{C}$ is any class of uniformly recursive strategies then $\mathcal{C}$ can be dominated by a recursive strategy.

4 Proofs of Main Theorems

Sketch of Proof of Theorem 3.2: We modify the proof of Gilboa and Samet of Theorem 3.1 [GS89] by first attempting to move the automaton into a strongly connected component that has no paths leading out of that component. The rest of the Gilboa-Samet proof will carry through. □

Sketch of Proof of Theorem 3.3: We create a regular strategy that only cooperates once Player I plays a specific sequence. Any strategy will require an exponential number of rounds to discover this sequence.

For every string $w \in \{c, d\}^*$ we create a strategy σ_{II}^{w} that works as follows: Player II will play "D" until Player I plays the sequence w. Player II will then play "C" while Player I plays "D". When Player I plays "C", Player II will start all over again.

We can implement strategy σ_{II}^{w} with an automaton with $|w| + 1$ states. We give an example of an automaton that implements $\sigma_{\mathrm{II}}^{cdccd}$ in Figure 1.

Clearly the optimal strategy for Player I against σ_{II}^{w} is to play w and then always play "d" achieving a limit of the means payoff of four. It is not hard to show that for any strategy for Player I and for every n, there is some w, $|w| = n - 1$ such that Player I will not play the sequence w in the first 2^{n-1} moves. Thus there exists an automaton with $|w| + 1 = n$ states where Player I's strategy can achieve a payoff of at most one in each of the first 2^{n-1} rounds. Therefore Player I will require an exponential number of rounds to converge if Player I ϵ-dominates finite automata for $\epsilon < 3$. □.

Proof of Theorem 3.4: Rivest and Schapire [RS93] extending techniques of Angluin [Ang87] present probabilistic polynomial-time algorithms for inferring finite automata in the setting with no reset but with a teacher that provides counterexamples. In our setting, we will not have access to such a teacher but the techniques will still be applicable because any loss will produce a counterexample.

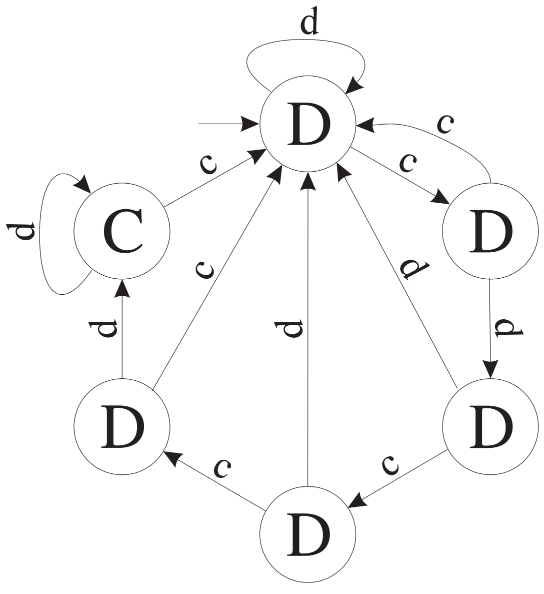

Figure 1: An automaton that implements $\sigma_{\mathrm{II}}^{cdccd}$.

We do not need to infer the entire automaton, but only enough to find a winning cycle. We use this advantage to build a nonbehavioral (deterministic) polynomial-round polynomial-time strategy to dominate regular strategies playing Matching Pennies, extending further the techniques of Rivest and Schapire and Angluin.

We use $a \cdot b$ to denote the concatenation of a and b and ϵ to denote the empty string. For now, let us assume that Player I knows n, an upper bound on the number of states of the minimal automaton that implements σ_{II}. We will show later how to get rid of this assumption.

We describe the strategy σ_{I} by the algorithm in Figure 2. The strategy creates a "homing sequence" w based loosely on the homing sequences of Rivest and Schapire. Player I will play w and use Player II's response r to help guess what state i Player II is currently in. If Player I guesses incorrectly then this helps refine the homing sequence. Otherwise Player I is able to add one new move to the winning sequence c_i starting at state i. Once the winning sequence is long enough than Player I can discover a winning cycle in the automaton.

We need to show that given that Player II uses a strategy implemented by a finite automaton using at most n states:

1. The algorithm will reach the ANALYZE phase within only a polynomial in n number of computation steps and rounds.

2. The algorithm will never output "ERROR".

The first requirement follows from Lemma 4.1.

START: $w \leftarrow h$; $k \leftarrow 0$

NEW w: $r_1, \ldots, r_n, c_1, \ldots, c_n \leftarrow \epsilon$

PLAY: Play the whole sequence w. Let r be the response string of Player II to w.

  If $r = r_i$ for some $i \leq k$ then Goto EXTEND

  If $k \geq n$ then ERROR

  $r_{k+1} \leftarrow r$; $k, i \leftarrow k+1$

EXTEND: Play c_i (If $c_i = \epsilon$ do nothing.)

  If a loss occurs during play of c_i then $w \leftarrow w \cdot c_i$; Goto NEW w

  If $|c_i| \geq n^2 + 3n$ then Goto ANALYZE

  Play "h"

    If win results then $c_i \leftarrow c_i \cdot h$

    If loss results then $c_i \leftarrow c_i \cdot t$

    GOTO PLAY

ANALYZE: Analyze c_i to find a minimal length cycle x that has been repeated for the last n^2 of the moves in c_i. Repeat x indefinitely. If a loss ever results then ERROR.

Figure 2: Polynomial-time strategy to dominate regular strategies playing Matching Pennies. The string c_i is the "winning continuation" after r_i is received as a response to w, the homing sequence which the algorithm generates.

Lemma 4.1 *The algorithm extends w at most $\binom{n}{2} = O(n^2)$ times. Also, each time w is extended, it is extended by a string of length at most $n^2 + 3n$.*

Proof: w is extended when the finite automaton starts on two different states q_a and q_b and produces the same response r but different responses on the extended word $w \cdot c_i$. The new $w = w \cdot c_i$ now distinguishes q_a from q_b. The number of pairs of states (q_a, q_b) for which $q_a \neq q_b$ is at most $\binom{n}{2}$. Each time w is extended, it is extended by c_i whose length is at most $n^2 + 3n$. □ (Lemma 4.1)

If the first ERROR occurs that means that there exists more than n responses to a fixed w. This cannot occur if σ_{II} is implemented by a finite automaton of at most n states.

Now suppose that algorithm has reached the ANALYZE phase. Observe that for every state of the finite automaton M implementing σ_{II}, there is a unique winning action for Player I. If the algorithm wins for more than $3n$ rounds consecutively, it must have repeated a "cycle" of states: At most n rounds to enter a cycle, and then at most $2n$ rounds to play a sequence twice.

Suppose we have a finite automaton M' that has the same response to c_i as M but on c_i is in a cycle z where z is not multiple iterations of x. Since x and z do not have any factor words in common, the string $x^{|z|} \neq z^{|x|}$. However since $|x^{|z|}| = |z^{|x|}| \leq n^2$ this is a contradiction, because Player I would have lost on one of the two strings, producing a loss within n^2 rounds.

Therefore Player II must be using an automaton that is in a cycle consisting of repetitions of x. Thus Player I repeatedly playing x will always win.

Now suppose Player I does not know the value of n. Player I initially assumes $n = 2$ and uses the algorithm in Figure 2. If an ERROR occurs then Player I starts all over again doubling his assumption about the value of n. Eventually, either no more errors occur in which case Player I always wins or Player I's assumption about n will dominate the number of states of the automaton used by Player II. □ (Theorem 3.4)

Proof of Theorem 3.5:

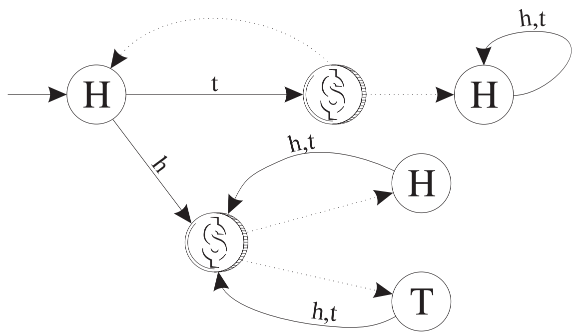

Figure 3: A probabilistic automaton for which no optimal strategy exists.

Look at the behavioral regular strategy σ_{II} implemented by the probabilistic finite automaton in Figure 3. When the automaton enters a coin state, it chooses a dotted path independently with probability one-half.

For any k, there exists a (regular) strategy σ_{I} for Player I that achieves an expected payoff of $1-2^{-k}$ by playing "t" for the first k moves and then always playing "h". However, an expected payoff of one is impossible: a strategy of always playing tails will yield a payoff of -1 and if Player I ever plays heads there is always a small probability that σ_{II} will be in the bottom part of the automaton giving Player I a payoff of zero. □

Sketch of Proof of Theorem 3.6: Consider the much simpler game:

	0	1
a	$(0,0)$	$(1,0)$
b	$(0,0)$	$(1,0)$

We design a "reward" function with two properties: first, that the longer σ_{I} "waits," the larger the reward; and second, that the reward is not paid while σ_{I} waits. Note as a consequence, if σ_{I} waits forever, he is paid nothing. σ_{II} will play as to allow σ_{I} to achieve (finite stage) means that converge to this reward.

Suppose the history of Player I at round n, h_{I}^n, has the form

$$a^{n_0}b^{m_0}a^{n_1}b^{m_1}a^{n_2}b^{m_2}\ldots a^{n_k}b^{m_k}$$

for some $k \geq 0$, and some sequence $n_0, m_0, \ldots n_k, m_k$ where all $n_i > 0$ and all $m_i > 0$ except for possibly n_0 and m_k.

Define $V(h_{\mathrm{I}}^n)$ as follows:

$$V(h_{\mathrm{I}}^n) = \sum_{i=0}^{k} \frac{n_i}{n_i+1} 2^{-(i+1)}$$

The polynomial time strategy σ_{II} will play 0 on the first round. On round $n+1$, the algorithm looks at the previous action of σ_{I}, a_{I}^n.

If $a_{\mathrm{I}}^n = b$ and $\mu_{\mathrm{I}}^n(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}}) < V(h_{\mathrm{I}}^n)$, then σ_{II} plays 1, otherwise σ_{II} plays 0. Player I waits and increases its future payoffs during the rounds in which he plays a, and he is paid his reward when he plays b.

Lemma 4.2 *For any history h_{I}^n and for any σ_{I} which is consistent with h_{I}^n, there is a σ_{I}' also consistent with h_{I}^n such that*

$$\pi^{G^\infty}(\sigma_{\mathrm{I}}', \sigma_{\mathrm{II}}) > \pi^{G^\infty}(\sigma_{\mathrm{I}}, \sigma_{\mathrm{II}})$$

Proof omitted.

The lemma shows us that no σ_{I} can be eventually optimal. The reasoning will also apply to the Prisoner's Dilemma game. □ (Theorem 3.6)

Sketch of Proof of Theorem 3.7: The rough idea is to have Player II in phase $m = \langle i,j,k\rangle$ "punish" Player I if the ith partial recursive function on the history up to round $2^{\langle i,j\rangle}$ computes in exactly k steps the action taken by Player I in round $2^{\langle i,j\rangle}+1$. Thus any recursive strategy will be "punished" infinitely often.

Formally, the strategy σ_{II} plays in phases, where the mth phase consists of 2^m rounds. Let $m = \langle i,j,k\rangle$. The strategy plays "D" for the whole phase if, given the move history up to round $2^{\langle i,j\rangle}$, the ith partial recursive function would have played the action made by Player I in round $2^{\langle i,j\rangle}+1$ in exactly k steps. Otherwise, the strategy will play "C" for the whole phase. Observe the following two lemmas:

Lemma 4.3 *Every recursive strategy σ_{I} will achieve a payoff of at most* 2.5.

Proof: Suppose strategy σ_{I} is implemented by the ith recursive function. Let $t(i,j)$ be the number of steps used by this function to compute the action in round $2^{\langle i,j\rangle}+1$. By the construction of σ_{II}, Player II will play "D" during phase $\langle i,j,t(i,j)\rangle$. Player I could only achieve a payoff of one during this phase. At the end of this phase, Player I could have at best an average payoff of 2.5. Since this will happen for every j, the lim inf of the average payoffs will be at most 2.5. □.

Lemma 4.4 *There is a rational strategy for Player I that achieves a payoff of four against σ_{II}.*

Proof: The strategy σ_{I} will play "c" in the mth round if m is of the form $2^{\langle i,j\rangle}+1$ and the ith partial recursive function given the current history would have halted with output "d". Otherwise, σ_{I} will play "d". Note that against σ_{I}, Player II will always play "C". Thus Player I will achieve a utility of four in every round that is not one greater than a power of two and thus in the limit Player I will have a payoff of four.

Hence, by the two lemmas, if $\epsilon < 1.5$, then no recursive strategy can become ϵ-optimal. □ (Theorem 3.7)

Sketch of Proof of Theorem 3.8: Let $\sigma_1, \sigma_2, \ldots$ be an enumeration of the polynomial-time strategies for Player II. Player I will assume that Player II is using strategy σ_1 until Player I loses. Player I will then assume that Player II is using strategy σ_2, etc. Eventually, Player I's assumptions will be correct and Player I will win from that point on. □

5 Open Questions

Theorem 3.5 only scratches the surface on our knowledge of optimality and domination when Player II uses a behavioral regular strategy. All of the following questions remain open:

- Does there exist a behavioral regular strategy for which there is no eventually optimal rational strategy?
- For some $\epsilon > 0$ does there exist a behavioral regular strategy for which there is no eventually ϵ-optimal recursive or polynomial-time strategy?
- In the other extreme, does there exist a polynomial-time or recursive strategy that eventually dominates all behavioral regular strategies.

We are also interested in questions about optimality and domination in other types of games such as infinite games that are not repeated stage games perhaps with imperfect information. Also, is there a way to formalize computational issues for finite games?

Finally, we would like to see how other game theoretic issues are affected by polynomial-time strategies. For example, what kinds of equilibria might be achieved by players using polynomial-time strategies in various different games?

Acknowledgments

We would like to thank Itzhak Gilboa, Ehud Kalai and In-Koo Cho for a game theory perspective and some very helpful discussions particularly on the definitions.

Robert Schapire has been incredibly helpful in pointing us to very useful literature in computational learning.

We would like to thank several of our fellow students and faculty at Chicago for various discussions on these topics. In particular, we would like to thank Stuart Kurtz for his help with Theorems 3.6 and 3.7.

References

[Ang87] Dana Angluin. Learning regular sets from queries and counterexample. *Information and Compuation*, 75:87–106, 1987.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and intractability. A Guide to the theory of NP-completeness.* W. H. Freeman and Company, New York, 1979.

[GS89] I. Gilboa and D. Samet. Bounded versus unbounded rationality: The tyranny of the weak. *Games and Economic Behavior*, 1(3):213–221, 1989.

[HU79] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, Mass., 1979.

[Kal90] E. Kalai. Bounded rationality and strategic complexity in repeated games. In T. Ichiishi, A. Neyman, and Y. Tauman, editors, *Game Theory and Applications*, pages 131–157. Academic Press, San Diego, 1990.

[Kno94] V. Knoblauch. Computable strategies for repeated prisoner's dilemma. *Games and Economic Behavior*, 1994. To appear.

[RS93] R. Rivest and R. Schapire. Inference of finite automata using homing sequences. *Information and Compuation*, 103(2):299–347, 1993.