Synthesis of Strategies from Interaction Traces

Tsz-Chiu Au
Dept. of Computer Science
University of Maryland
College Park, MD 20742, USA
chiu@cs.umd.edu

Sarit Kraus
Dept. of Computer Science
Bar-Ilan University
Ramat Gan, 52900, Israel
sarit@cs.biu.ac.il

Dana Nau
Dept. of Computer Science
University of Maryland
College Park, MD 20742, USA
nau@cs.umd.edu

ABSTRACT

We describe how to take a set of interaction traces produced by different pairs of players in a two-player repeated game, and combine them into a *composite strategy*. We provide an algorithm that, in polynomial time, can generate the best such composite strategy. We describe how to incorporate the composite strategy into an existing agent, as an enhancement of the agent's original strategy.

We provide experimental results using interaction traces from 126 agents (most of them written by students as class projects) for the Iterated Prisoner's Dilemma, Iterated Chicken Game, and Iterated Battle of the Sexes. We compared each agent with the enhanced version of that agent produced by our algorithm. The enhancements improved the agents' scores by about 5% in the IPD, 11% in the ICG, and 26% in the IBS, and improved their rank by about 12% in the IPD, 38% in the ICG, and 33% in the IBS.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Performance, Economics, Experimentation, Theory

Keywords

Agents, interaction, learning, multi-agent systems, prisoner's dilemma, repeated games

1. INTRODUCTION

To create new and better agents in multi-agent environments, we may want to examine the strategies of several existing agents, in order to combine their best skills. One problem is that in general, we won't know what those strategies are. Instead, we'll only have observations of the agents' interactions with other agents. The question is how to synthesize, from these observations, a new strategy that performs as well as possible.

In this paper we present techniques for taking *interaction traces* (i.e., records of observed interactions) from many different pairs of agents in a 2-player iterated game, and synthesizing from these traces a new strategy called a *composite strategy* (see Figure 1). We also show how an existing agent can enhance its performance

Cite as: Synthesis of Strategies from Interaction Traces, Tsz-Chiu Au, Sarit Kraus and Dana Nau, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

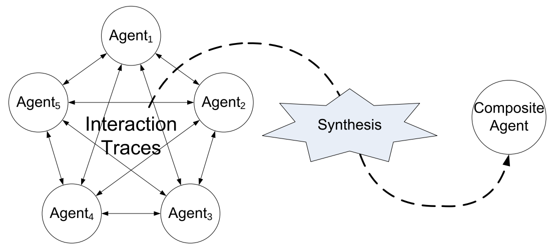


Figure 1: A composite strategy is synthesized from records of the interactions among many existing agents. An agent using this strategy can potentially outperform the agents from whom the interaction traces were obtained.

by combining its own strategy with the composite strategy. Our contributions include the following:

- We give a formal definition of a composite strategy, and present necessary and sufficient conditions under which a set of interaction traces from different agents can be combined together to form a composite strategy.
- We provide a polynomial-time algorithm for synthesizing, from a given set of interaction traces $\mathcal{T}$, a composite strategy that is *optimal*, in the sense that it performs at least as well as any other composite strategy that can be formed from $\mathcal{T}$.
- We provide a way to enhance an existing agent's performance by augmenting its strategy with a composite strategy.
- We provide experimental results demonstrating our algorithm's performance in the Iterated Prisoner's Dilemma (IPD), Iterated Chicken Game (ICG), and Iterated Battle of the Sexes (IBS), using interaction traces from 126 agents (117 written by students as class projects, and 9 standard agents from the published literature). For each agent, we compared its performance with the performance of our enhanced version of that agent. On the average, the percentage improvements in score were about 5% in the IPD, 10% in the ICG, and 25% in the IBS; and the percentage improvements in rank were about 11% in the IPD, 38% in the ICG, and 33% in the IBS.

2. BASIC DEFINITIONS

Consider a two-player finite repeated game, such as the IPD. At any time point t, two agents ϕ_A and ϕ_B can choose actions a and b from finite sets of actions A and B, respectively. Neither agent learns what the other's action is until after choosing its own action. Throughout this paper, we will look at games from ϕ_A's point of view; i.e., ϕ_A is "our agent" and ϕ_B is the opponent.

We call the pair (a, b) an *interaction* between ϕ_A and ϕ_B. We assume interactions occur at a finite number of discrete time points $t_1, t_2, \ldots, t_n$, where n is the total number of interactions.[1] An *interaction trace* between ϕ_A and ϕ_B is a sequence of interactions $\tau = \langle (a_1, b_1), (a_2, b_2), \ldots (a_n, b_n) \rangle$, where a_i and b_i are the actions of ϕ_A and ϕ_B at time t_i.[2] The histories of ϕ_A's and ϕ_B's actions are $\tau^A = \langle a_1, a_2, \ldots, a_n \rangle$ and $\tau^B = \langle b_1, b_2, \ldots, b_n \rangle$, respectively. The *history* up to time t_k is the *k'th prefix* of τ, i.e., it is the subsequence $\mathsf{prefix}_k(\tau) = \langle (a_1, b_1), (a_2, b_2), \ldots (a_k, b_k) \rangle$. τ_k^A and τ_k^B are $\langle a_1, a_2, \ldots, a_k \rangle$ and $\langle b_1, b_2, \ldots, b_k \rangle$, respectively.

A *pure strategy* is a function ϕ from histories into actions, that returns the next action to perform. A *mixed strategy* is a pair $\bar{\phi} = (\Phi, \Delta)$, where Φ is a nonempty set of pure strategies and Δ is a probability distribution over Φ. To make it easy to distinguish between pure and mixed strategies (or equivalently, between deterministic and probabilistic agents), we'll normally write the latter with a line over it (e.g., $\bar{\phi}$ rather than ϕ).

An agent uses a *strategy* to determine what actions it should take in every time point. A *deterministic* agent is one whose strategy is a pure strategy, and a *probabilistic* agent is one whose strategy is a mixed strategy. We often will use the same symbol (e.g., ϕ_A or ϕ_B) to denote both the agent and its strategy.

When two deterministic agents ϕ_A and ϕ_B interact, only one possible interaction trace can be generated, namely the interaction trace $\mathsf{trace}(\phi_A, \phi_B) = \tau_n$, where τ_i is defined recursively as follows: $\tau_0 = \langle \rangle$, and $\tau_{j+1} = \tau_j \circ \langle (\phi_A(\tau_j), \phi_B(\tau_j)) \rangle$ for $i = 1, \ldots, n$. We assume ϕ_A's performance is measured by a *utility function* $U_A(\mathsf{trace}(\phi_A, \phi_B)) \geq 0$ that gives the reward for ϕ_A when its opponent is ϕ_B.

Consider an agent $\bar{\phi}_B$ that uses a mixed strategy (Φ_B, Δ_B). We will call the pure strategies in Φ_B the *possible* strategies for $\bar{\phi}_B$. In each game, $\bar{\phi}_B$ will choose one strategy $\phi \in \Phi_B$ according to the probability distribution Δ_B, and ϕ_A will not know which strategy was chosen. We call the chosen strategy $\bar{\phi}_B$'s *actual* strategy in this game. The overall performance of ϕ_A in its interactions with $\bar{\phi}_B$ is ϕ_A's *expected utility*, which is

$$E(\phi_A, \bar{\phi}_B) = \sum \{ \Delta_B(\phi) \times U_A(\mathsf{trace}(\phi_A, \phi)) : \phi \in \Phi_B \}.$$

3. SYNTHESIS OF STRATEGIES FROM INTERACTION TRACES

Section 3.1 presents an algorithm that can reconstruct the strategy of a single agent ϕ_A, given a collection of interaction traces generated by ϕ_A. Section 3.2 shows how to take a collection of traces that were generated by more than one agent, and construct a new strategy called a *composite* strategy, that combines parts of the strategies of all of those agents. Section 3.3 gives an algorithm to find the optimal one of these composite strategies.

3.1 Reconstructing an Agent's Strategy

Suppose we play a pure strategy ϕ_A against a mixed strategy $\bar{\phi}_B = (\Phi, \Delta)$, where $\Phi = \{\phi_1, \phi_2, \phi_3\}$. Then the set of all possible interaction traces is

$$\mathcal{T} = \{ \mathsf{trace}(\phi_A, \phi_j) : j \in \{1, 2, 3\} \}.$$

[1]It is easy to modify our definitions and algorithms to handle games in which the number of interactions can vary. But to simply our discussion, we'll assume the number of interactions is constant.

[2]This formalism can easily model environments where the actions are interleaved rather than occurring simultaneously, by specifying that at odd time points ϕ_A's action must always be a "null" action, and similarly for ϕ_B at even time points.

```
Agent CA($\mathcal{T}$)    /* a composite agent synthesized from $\mathcal{T}$ */
1. $i := 1$ ; $\mathcal{T}_i := \mathcal{T}$
2. Loop until the end of the game
3.   If $\mathcal{T}_i = \emptyset$, then exit with failure because $\mathcal{T}$ is insufficient
4.   If $\mathcal{T}_i \neq \emptyset$, then
5.     $A_i := \{a : (\exists\, \tau \in \mathcal{T}_i)\ a$ is the $i$'th action of $\tau^A\}$
6.     If $|A_i| \neq 1$, then exit with failure because $\mathcal{T}$ is incompatible
7.     If $|A_i| = 1$, then let $a_i$ be the action in $A_i$
8.     Output $a_i$ and receive the other agent's action $b_i$
9.     $\mathcal{T}'_i := \mathcal{T}_i$
10.    For each $\tau \in \mathcal{T}_i$,
11.      If the $i$'th interaction in $\tau$ isn't $(a_i, b_i)$, remove $\tau$ from $\mathcal{T}'_i$
12.    $\mathcal{T}_{i+1} := \mathcal{T}'_i$ ; $i := i + 1$
```

Figure 2: The pseudocode of a composite strategy.

$\mathcal{T}$ contains enough information for us to construct a new strategy $\phi_{\mathcal{T}}$ whose interaction traces with $\bar{\phi}_B$ will be exactly the same as ϕ_A's. Figure 2 gives the pseudocode for an agent CA($\mathcal{T}$) that can generate and play the strategy $\phi_{\mathcal{T}}$. We will call CA($\mathcal{T}$) a *composite agent*, and $\phi_{\mathcal{T}}$ a *composite strategy*. The strategy $\phi_{\mathcal{T}}$ generated by CA is *partial*, i.e., it is only defined over some of the possible histories. However, as we will see in Theorem 1, $\phi_{\mathcal{T}}$ is $\bar{\phi}_B$*-total*, i.e., it is defined over the set of all histories that can are possible when playing it against $\bar{\phi}_B$. In other words, in any history τ_k of interactions between $\phi_{\mathcal{T}}$ and $\bar{\phi}_B$, $\phi_{\mathcal{T}}$ will always be able to determine what its next action a_{k+1} should be.

To see how the CA agent works, suppose that the three interaction traces in $\mathcal{T}$ are

$$\begin{aligned} \tau_1 &= \langle (C, C), (C, D), (D, C) \rangle, \\ \tau_2 &= \langle (C, C), (C, C), (C, C) \rangle, \\ \tau_3 &= \langle (C, D), (C, D), (D, C) \rangle. \end{aligned}$$

Next, suppose we play CA($\mathcal{T}$) against $\bar{\phi}_B$. Since ϕ_A's first action is C in all three traces, we have $A_i = \{C\}$ at Line 5 of CA, so CA($\mathcal{T}$) returns $a_1 = C$ as its first action. Suppose $\bar{\phi}_B$'s first action is C. Then $\bar{\phi}_B$ cannot be using the strategy that produced τ_3, so Line 11 of the agent removes τ_3 from $\mathcal{T}'_1$, leaving τ_1 and τ_2 in $\mathcal{T}_2$. Hence in the next interaction with $\bar{\phi}_B$, CA($\mathcal{T}$) will choose $a_2 = C$. Suppose $\bar{\phi}_B$'s second action is $b_2 = D$. Then τ_2 is removed from $\mathcal{T}'_2$, and CA($\mathcal{T}$)'s next action is D. Hence that the composite agent CA($\mathcal{T}$) ended up "replaying" the interaction trace τ_1.

The following theorem provides a condition under which CA($\mathcal{T}$) will always generate the same action that ϕ_A would generate against $\bar{\phi}_B$:

THEOREM 1. *Let ϕ_A and $\bar{\phi}_B = (\Phi_B, \Delta_B)$ be pure and mixed strategies, respectively. If $\mathcal{T} = \{\mathsf{trace}(\phi_A, \phi) : \phi \in \Phi_B\}$, then $\mathsf{trace}(CA(\mathcal{T}), \phi) = \mathsf{trace}(\phi_A, \phi)$ for every $\phi \in \Phi_B$.*

Sketch of Proof. Due to limited space, we only sketch the proof here. Without loss of generality, let ϕ be the actual strategy of $\bar{\phi}_B$, and let $\tau = \mathsf{trace}(\phi_A, \phi)$ be the interaction trace between ϕ_A and ϕ. Suppose there exists i such that (a_i, b_i) at Line 8 of Figure 2 is not the i'th interaction (a'_i, b'_i) in τ, but, for $1 \leq j < i$, (a_j, b_j) is the j'th interaction in τ. First, all interaction traces in $\mathcal{T}_i$ have the same prefix up to the $(i-1)$'th iteration (any traces without this prefix were removed at Line 11 on a previous iteration) and ϕ_A is a deterministic agent. Therefore all traces in $\mathcal{T}_i$ (including τ) have the same i'th action a'_i, and the composite agent will certainly output a'_i at the iteration i. Thus, $a_i = a'_i$. Second, ϕ will certainly output b'_i at the i'th interaction, given the action sequence $a_1, a_2, \ldots, a_{i-1}$ as its inputs; that is, $b_i = b'_i$. Hence, a contradiction occurs. Therefore (a_i, b_i) at Line 8 of Figure 2 is always the

i'th interaction in τ, and $\mathsf{trace}(\mathrm{CA}(\mathcal{T}), \phi) = \tau$. □

3.2 Constructing New Strategies

The previous section showed how to reconstruct a strategy of a single agent ϕ_A against an agent $\bar{\phi}_B$, given ϕ_A's interaction traces with $\bar{\phi}_B$. The same approach can be used to construct *new* strategies from a collection of traces generated by more than one agent, provided that two conditions, called *compatibility* and *sufficiency*, are satisfied.

To illustrate the notion of compatibility, consider two well-known strategies for the IPD: Tit-For-Tat (TFT) and Tit-For-Two-Tats (TFTT). In the usual setting, the optimal strategy against TFT is to cooperate in every iteration, and the optimal strategy against TFTT is to defect and cooperate alternatively. For an IPD game of five iterations, these can be represented by the following two interaction traces:

$$\begin{aligned}\tau_1 &= \langle (C,C),(C,C),(C,C),(C,C),(C,C)\rangle,\\ \tau_2 &= \langle (D,C),(C,C),(D,C),(C,C),(D,C)\rangle.\end{aligned}$$

However, no pure strategy can generate both of these interaction traces, because in the first interaction, no agent can choose both Cooperate and Defect simultaneously. Thus, there is no single agent that can act optimally against both TFT and TFTT, and we say that τ_1 and τ_2 are *incompatible* with each other. If we run $\mathrm{CA}(\{\tau_1, \tau_2\})$, it will return an error message at Line 6 of Figure 2.

Now consider an agent that defects in the first iteration and then cooperates for the rest of a game. When this agent plays against TFT, the interaction trace is $\tau_3 = \langle (D,C),(C,D),(C,C),(C,C),(C,C)\rangle$. Although this strategy is not optimal against TFT, it is *compatible* with τ_2: both τ_2 and τ_3 produce D for a_1 and C for a_2; and the opponent's response at the end of the 2nd interaction gives us enough information to decide which of τ_2 and τ_3 to use thereafter. If the opponent's second action b_2 is C, then we discard τ_3 and continue with τ_2, and if b_2 is D, we discard τ_2 and continue with τ_3. This is exactly what $\mathrm{CA}(\{\tau_2, \tau_3\})$ does when it plays against a mixed strategy that includes TFT and TFTT.

We now formalize the concepts introduced in the above example, to provide necessary and sufficient condition on $\mathcal{T}$ for the synthesis of a composite strategy.

DEFINITION 1. *The* longest common prefix *of two action sequences $\alpha = \langle a_1, a_2, \ldots, a_n\rangle$ and $\alpha' = \langle a'_1, a'_2, \ldots, a'_n\rangle$ is the longest action sequence* $\mathsf{lcp}(\alpha, \alpha')$ *that's a prefix of both α and α'.*

We now define a condition called *compatibility*, that (as we'll see in Theorem 2) is necessary for a set of interaction traces $\mathcal{T}$ to be used successfully by CA. The interaction traces do not all need to be generated by the same agent.

DEFINITION 2. *Two interaction traces τ_1 and τ_2 are* compatible *if either (1) $\tau_1^A = \tau_2^A$, or (2) $|\mathsf{lcp}(\tau_1^A, \tau_2^A)| > |\mathsf{lcp}(\tau_1^B, \tau_2^B)|$. Otherwise, τ_1 and τ_2 are* incompatible.

DEFINITION 3. *A set $\mathcal{T}$ of interaction traces is* compatible *if and only if there is no incompatible pair of interaction traces in $\mathcal{T}$.*

Even if a set $\mathcal{T}$ of interaction traces is compatible, $\mathrm{CA}(\mathcal{T})$ will not always be able to use them against $\bar{\phi}_B$ unless there is at least one interaction trace for each of $\bar{\phi}_B$'s possible strategies. The following definition formalizes this notion.

DEFINITION 4. *Let $\bar{\phi}_B = (\Phi_B, \Delta_B)$ be a mixed strategy. A set $\mathcal{T}$ of interaction traces is* $\bar{\phi}_B$-sufficient *if and only if for every strategy $\phi \in \Phi_B$, $\mathcal{T}$ contains an interaction trace $\tau = \langle (a_1,b_1),\ldots,(a_n,b_n)\rangle$ such that the action sequence $\langle b_1,\ldots,b_n\rangle$ can be generated by ϕ given $\langle a_1,\ldots,a_n\rangle$ as its inputs.*

The following theorem shows that compatibility and $\bar{\phi}_B$-sufficiency are necessary and sufficient to guarantee that $\mathrm{CA}(\mathcal{T})$ will always be able to play an entire game against $\bar{\phi}_B$.

THEOREM 2. *The composite agent $\mathrm{CA}(\mathcal{T})$ will never exit with failure when it plays against an opponent $\bar{\phi}_B = (\Phi_B, \Delta_B)$, if and only if $\mathcal{T}$ is compatible and $\bar{\phi}_B$-sufficient.*

Sketch of Proof. The "only if" part is trivial. For the "if" part, suppose $\mathcal{T}$ is compatible and $\bar{\phi}_B$-sufficient. Without loss of generality, let ϕ be the strategy that $\bar{\phi}_B$ chooses to use at the start of the game. By induction on i, we prove that $\mathrm{CA}(\mathcal{T})$ does not exit with failure at the i'th iteration. More precisely, we prove (1) $|A_i| = 1$ at Line 6, and (2) there exist $\tau \in \mathcal{T}_i$ such that τ can be generated by ϕ (i.e., $\mathcal{T}_i \neq \emptyset$ at Line 3), for $1 \le i \le n$. First, let us consider $i = 1$. For any two interaction traces $\tau_1, \tau_2 \in \mathcal{T}$, the first actions in τ_1^A and τ_2^A must be the same since $\mathcal{T}$ is compatible. Therefore, $|A_1| = 1$. Since $\mathcal{T}$ is $\bar{\phi}_B$-sufficient and ϕ_B is nonempty, it follows that there is at least one interaction trace $\tau \in \mathcal{T}$ that can be generated by ϕ. Thus, $\mathcal{T}_1 \neq \emptyset$. Now consider $i = k+1$. Suppose $|A_k| = 1$ and there exist $\tau \in \mathcal{T}_k$ that can be generated by ϕ. First, since all traces in $\mathcal{T}_k$ have the same prefix up to the $(k-1)$'th iteration and ϕ is a pure strategy, ϕ will return a unique b_k at Line 8, which is the k'th action of τ (otherwise τ is not generated by ϕ). In addition, by Definition 2, for any two interaction traces $\tau_1, \tau_2 \in \mathcal{T}_k$, the first k actions in τ_1^A and τ_2^A must be the same. Thus, the k'th action of τ^A must be a_k, the action generated at Line 8. Therefore, (a_k, b_k) is the i'th interaction of τ, and τ will not be removed from $\mathcal{T}'_k$ at Line 11. Hence $\tau \in \mathcal{T}_{k+1}$ since $\mathcal{T}_{k+1} = \mathcal{T}'_k$ at Line 12. Second, $|A_{k+1}| \ge 1$ because $|\mathcal{T}_{k+1}| \ge 1$. Third, at the start of each iteration $k+1$, all interaction traces in $\mathcal{T}_{k+1}$ have the same prefix up to the k'th iteration (any traces without this prefix were removed at Line 11 on a previous iteration). By Definition 2, for any two interaction traces $\tau_1, \tau_2 \in \mathcal{T}_{k+1}$, the first $k+1$ actions in τ_1^A and τ_2^A must be the same; and consequently $|A_{k+1}| \le 1$. Therefore, $|A_{k+1}| = 1$. By induction, $|A_i| = 1$ and $\mathcal{T}_i \neq \emptyset$, for $1 \le i \le n$. □

3.3 Finding the Best Composite Strategy

We now consider the problem of finding a strategy against a mixed strategy. It is not difficult to show by reduction from 3SAT that the problem of finding an *optimal* strategy—a strategy with the highest expected utility against a mixed strategy—is **NP**-hard. Therefore, instead of finding the optimal strategy, we find the best composite strategy that can be synthesized from a given set of interaction traces, and then experimentally show that the best composite strategy can perform pretty well in practice.

Let $\bar{\phi}_B = (\Phi_B, \Delta_B)$ be a mixed strategy. Suppose we are given a set $\mathcal{T}_B$ of interaction traces that were played against $\bar{\phi}_B$; and suppose that for each interaction trace $\tau \in \mathcal{T}_B$, we know the utility $U_A(\tau)$ for the agent that played against $\bar{\phi}_B$. Let

$$\mathbb{T} = \{\mathcal{T} \subseteq \mathcal{T}_B : \mathcal{T} \text{ is compatible and } \bar{\phi}_B\text{-sufficient}\};$$

and for each $\mathcal{T} \in \mathbb{T}$, let $\phi_{\mathcal{T}}$ be the composite strategy constructed from $\mathcal{T}$. Then the **optimal composite strategy problem** is the problem of finding a composite agent $\phi_{\mathcal{T}^*}$ such that $\mathcal{T}^* \in \mathbb{T}$ and $E(\phi_{\mathcal{T}^*}, \bar{\phi}_B) \ge E(\phi_{\mathcal{T}}, \bar{\phi}_B)$ for every $\mathcal{T} \in \mathbb{T}$. We say that $\phi_{\mathcal{T}^*}$ is $(\mathcal{T}_B, \bar{\phi}_B)$*-optimal.*

Here is another formulation of the optimal composite strategy problem that's equivalent to the above formulation. Suppose we are given sets of interaction traces $\mathcal{T}_1, \ldots, \mathcal{T}_m$ from games against several different agents $\phi_1, \ldots, \phi_m$, and for each $\tau_i \in \mathcal{T}_i$ we are given

```
Procedure CIT($k, \{\mathcal{T}_i\}_{i=1,\ldots,m}, \{p_i\}_{i=1,\ldots,m}$)
1. If $k > n$, then        /* $n$ is the maximum number of interactions */
2.   For $i = 1$ to $m$, choose any one trace, namely $\tau_i$, in $\mathcal{T}_i$
3.   Return $(\{\tau_i\}_{i=1..m}, \sum_{i=1..m}\{p_i \times U(\tau_i)\})$
4. Else
5.   $A_i := \{a_k : \langle(a_j, b_j)\rangle_{j=1..n} \in \mathcal{T}_i\}$, for $i := 1..m$
6.   $B_i := \{b_k : \langle(a_j, b_j)\rangle_{j=1..n} \in \mathcal{T}_i\}$, for $i := 1..m$
7.   $A' := A_1 \cap A_2 \cap \ldots \cap A_m$; $B' := B_1 \cup B_2 \cup \ldots \cup B_m$
8.   If $A' = \emptyset$, then return $(\emptyset, -1)$    /* incompatibility detected */
9.   If $|B_i| \neq 1$ for some $i$, then exit with failure.
10.  For $i := 1$ to $m$
11.    Partition $\mathcal{T}_i$ into $\mathcal{T}_i^{ab}$ for each pair $(a, b) \in A' \times B'$ such that
12.      $\mathcal{T}_i^{ab} := \{\tau \in \mathcal{T}_i$ : the k'th interaction in $\tau$ is $(a, b)\}$
13.  For each $(a, b) \in A' \times B'$
14.    $\mathbb{T}^{ab} := \{\mathcal{T}_i^{ab} : 1 \leq i \leq m, \mathcal{T}_i^{ab} \neq \emptyset\}$
15.    $\mathcal{P}^{ab} := \{p_i : 1 \leq i \leq m, \mathcal{T}_i^{ab} \in \mathcal{T}^{ab}\}$
16.    $(\mathcal{T}_*^{ab}, U_*^{ab}) :=$ CIT$(k+1, \mathbb{T}^{ab}, \mathcal{P}^{ab})$    /* call CIT itself */
17.  For each $a \in A'$
18.    If $U_*^{ab} \geq 0$ for all $b \in B'$, then
19.      $\hat{\mathcal{T}}_a := \bigcup_{b \in B'}\{\mathcal{T}_*^{ab}\}$; $\hat{U}_a := \sum_{b \in B'} U_*^{ab}$
20.    Else     /* i.e., $\mathcal{T}_*^{ab}$ is not a solution for some $b \in B'$ */
21.      $\hat{\mathcal{T}}_a := \emptyset$; $\hat{U}_a := -1$      /* i.e., no solution */
22. $a^{max} := \arg\max_{a \in A'}\{\hat{U}_a\}$; Return $(\hat{\mathcal{T}}_{a^{max}}, \hat{U}_{a^{max}})$
```

Figure 3: The pseudocode of the CIT algorithm.

the utility $U_A(\tau_i)$ for the agent that played against ϕ_i. Furthermore, suppose we are given numbers $p_1, \ldots, p_m$ such that p_i is the probability that we'll need to play against ϕ_i. Now, consider the problem of finding a strategy with an optimal expected utility against these agents. This is equivalent to an optimal composite strategy problem in which $\mathcal{T}_B = \{\mathcal{T}_1 \cup \ldots \cup \mathcal{T}_m\}$ and $\bar{\phi}_B = (\{\phi_1, \ldots, \phi_m\}, \Delta)$, where $\Delta(\phi_i) = p_i$ for each i.

As an example, suppose we want to play the IPD against two agents ϕ_1 = TFT and ϕ_2 = TFTT, who will be our opponents with probabilities $p_1 = 0.7$ and $p_2 = 0.3$, respectively. Suppose we are given $\mathcal{T}_1 = \{\tau_1, \tau_2, \tau_3\}$, where

$$\begin{aligned} \tau_1 &= \langle(C, C), (C, C), (D, C)\rangle; & U_A(\tau_1) &= 11.0; \\ \tau_2 &= \langle(D, C), (C, D), (C, C)\rangle; & U_A(\tau_2) &= 8.0; \\ \tau_3 &= \langle(D, C), (D, D), (D, D)\rangle; & U_A(\tau_3) &= 7.0; \end{aligned}$$

and $\mathcal{T}_2 = \{\tau_1', \tau_2', \tau_3'\}$, where

$$\begin{aligned} \tau_1' &= \langle(C, C), (C, C), (C, C)\rangle; & U_A(\tau_1') &= 9.0; \\ \tau_2' &= \langle(D, C), (C, C), (D, C)\rangle; & U_A(\tau_2') &= 13.0; \\ \tau_3' &= \langle(D, C), (D, C), (D, D)\rangle; & U_A(\tau_3') &= 11.0. \end{aligned}$$

We can map this into the optimal composite agent problem by letting $\bar{\phi}_B = (\Phi_B, \Delta_B)$ and $\mathcal{T}_B = \mathcal{T}_1 \cup \mathcal{T}_2$, where $\Phi_B = \{\phi_1, \phi_2\}$, $\Delta_B(\phi_1) = 0.7$, and $\Delta_B(\phi_2) = 0.3$.

There are nine subsets of $\mathcal{T}$ that contain one trace for each of ϕ_1 and ϕ_2 (and hence are $\bar{\phi}_B$-sufficient), namely $\{\tau_j, \tau_k'\}$ for $j = 1, 2, 3$ and $k = 1, 2, 3$. Only two of these nine sets are compatible: $\{\tau_2, \tau_2'\}$ and $\{\tau_3, \tau_3'\}$. Of the two sets, $\{\tau_2, \tau_2'\}$ is the $(\mathcal{T}_B, \bar{\phi}_B)$-optimal one, because $E(\{\tau_2, \tau_2'\}, \bar{\phi}_B) = 9.5 > 8.2 = E(\{\tau_3, \tau_3'\}, \bar{\phi}_B)$. Hence, our $(\mathcal{T}_B, \bar{\phi}_B)$-optimal strategy is to choose D and C in the first two iterations, and then choose C (or D) in the third iteration if the opponent chooses D (or C) in the second iteration, respectively.

Figure 3 shows an algorithm, CIT, that can be used to solve the above problem. CIT is a recursive algorithm that works by analyzing interaction traces played against a set of agents $\{\phi_1, \ldots, \phi_m\}$. Its inputs include, for each ϕ_i, a set of interaction traces $\mathcal{T}_i$ and a probability p_i that we'll have ϕ_i as our opponent; and a number k that represents how many moves deep we have gone in our analysis of the interaction traces.

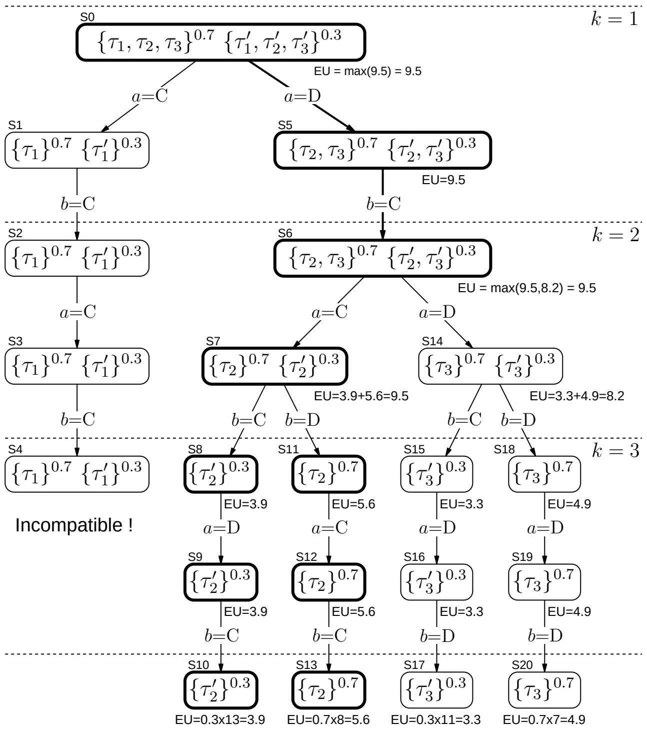


Figure 4: The search space of the CIT algorithm.

Example. We'll now illustrate CIT's operation on the same example that we discussed earlier. In Figure 4, the node S0 represents the initial call to CIT. The two sets of traces are $\mathcal{T}_1$ and $\mathcal{T}_2$ described earlier, and the superscripts on these traces are the probabilities $p_1 = 0.7$ and $p_2 = 0.3$ described earlier.

Each path from S0 to the bottom of the tree is one of the interaction traces; and the value $k = 1$ at S0 indicates that we're currently looking at the first interaction in each trace.

At the k'th interaction for each k, we need to consider both our possible moves and the opponent's possible moves. Although these moves are made simultaneously, we can separate the computation into two sequential choices: for each value of k, the higher layer of nodes corresponds to *our* possible moves, and the lower layer of nodes corresponds to the opponent's possible moves. In the higher layer, the expected utility of each node can be computed by taking the maximum of the expected utilities of the child nodes; hence we call these nodes *max nodes*. In the lower layer, the expected utility of each node can be computed by adding the expected utilities of the child nodes; hence we call these nodes *sum nodes*.[3]

In our example, the max node at $k = 1$ is S0, and the sum nodes at $k = 1$ are S1 and S5. The edges between the max node and the sum nodes correspond to our actions that can lead from the max node to the sum nodes. For instance, if our action is C at S0, the next sum node is S1; otherwise, the next sum node is S5. The edges between the sum nodes at $k = 1$ and the max nodes at $k = 2$ corresponds to the actions that can be chosen by the opponent. At $k = 1$, the opponent can only choose C, but at S7 at $k = 2$, the opponent can choose either C or D, thus leading to two different max nodes S8 and S11. A terminal node corresponds to the set of interaction traces that are consistent with the actions on the path

[3]Mathematically, what we're computing here is a weighted average; but each number has already been multiplied by the appropriate weight, so we just need to add the numbers together.

from S0 to the terminal node. The expected utility of a terminal node is the sum of the probability of the set of interaction traces (denoted by the superscripts in Figure 4) times the utility of any interaction trace in the set. For instance, at S10, the expected utility is $\sum_{i=1..m}\{p_i \times U_A(\tau_i)\} = 0.3 \times 13 = 3.9$. Notice that all interaction traces in a set of interaction trace of a terminal node are the same; thus the algorithm chooses any τ_i from $\mathcal{T}_i$ at Line 2 since they all have the same utility.

The CIT algorithm basically does a depth-first search on a tree as shown in Figure 4, and propagates the expected utilities of the terminal nodes to S0 together with the compatible set of interaction traces that gives the expected utility. The expected utility of a max node is the maximum of the expected utilities of its child nodes, whereas the expected utility of a sum node is the sum of the expected utilities of its child nodes. Notice that at each max node the CIT algorithm will check the compatibility of the set of interaction traces of the node at Line 8. For example, at S4 the algorithm discovers that τ_1 and τ_1' are incompatible. Then a failure signal (the expected utility -1) is passed from S4 to the ancestor nodes. The max nodes and the sum nodes would then ignore the solution with a failure signal, thus eliminate the incompatibility. This is one of the key difference between the CIT algorithm and other search algorithms for game trees or MDPs—the CIT algorithm directly operates with interaction traces and checks the incompatibility among them as the search process proceeds.

Running time. To achieve efficient performance in CIT we have used some indexing schemes that we will not describe here, due to lack of space. CIT's running time is $O(nM)$, where n is the number of iterations and $M = \sum_{i=1}^{m} |\mathcal{T}_i|$. In practice, the CIT algorithm is very efficient—it can find the optimal solution from a database of $500,000$ interaction traces in less than 15 minutes on a computer with a 3.8GHz Pentium 4 CPU and 2GB RAM.

Discussion. At first glance, CIT's search space (see Figure 4) looks somewhat like a game tree; but there are several important differences. First, our purpose is to compute an agent's entire strategy offline, rather than having the agent do an online game-tree search at every move of a game. Second, we are not searching an entire game tree, but are just searching through a set of interaction traces; hence the number of nodes in our tree is no greater than the total number of interactions in the interaction traces. Third, game-tree search always assumes, either explicitly or tacitly, a model of the opponent's behavior.[4] Rather than assuming any particular model of the opponent, we instead proceed from observations of agents' actual interactions.

3.4 Using a Base Strategy

Recall that CA($\mathcal{T}$)'s strategy is partial. In particular, although CA($\mathcal{T}$) will never exit with failure when playing against $\bar{\phi}_B$ when $\mathcal{T}$ is compatible and $\bar{\phi}_B$-sufficient, it might do so if we play it against another opponent $\bar{\phi}_C = (\Phi_C, \Delta_C)$ for which $\mathcal{T}$ is not $\bar{\phi}_C$-sufficient. However, in iterated games such as the IPD, there is enough overlap in the strategies of different agents that even if $\bar{\phi}_C$ was not used to construct any of the traces in $\mathcal{T}$, $\mathcal{T}$ may still be $\bar{\phi}_C$-sufficient.

[4]For example, minimax game-tree search assumes that the opponent will always use its dominant strategy. In perfect-information games such as chess, this opponent model has worked so well that it is taken more-or-less for granted. But in an imperfect-information variant of chess, it has been shown experimentally [13] that this model is not the best one—instead, it is better to use a model that assumes the opponent has very little idea of what its dominant strategy might be.

```
Agent MCA($\mathcal{T}_B, \Delta, \phi_{base}$)      /* a modified composite agent */
1. $\mathcal{T}^* :=$ CIT($0, \mathcal{T}_B, \Delta$)       /* $\mathcal{T}^*$ is $(\mathcal{T}_B, \bar{\phi}_B)$-optimal */
2. $\phi_A :=$ CA($\mathcal{T}^*$)              /* $\phi_A$ is a composite agent */
3. Loop until the end of the game
4.    Get an action $a$ from $\phi_A$
5.    If $\phi_A$ fails, then get an action $a$ from $\phi_{base}$ and $\phi_A := \phi_{base}$
6.    Output $a$ and receive the other agent's action $b$
7.    Give $b$ to $\phi_A$ as its input
```

Figure 5: Algorithm for a composite agent with a base strategy.

Even if $\mathcal{T}$ is not $\bar{\phi}_C$-sufficient, CA($\mathcal{T}$) may still be able to play against $\bar{\phi}_C$ most of the time without exiting with failure. To handle cases where CA($\mathcal{T}$) does exit with failure, we can modify the CA algorithm so that instead of exiting with failure, it instead uses the output produced by a "base strategy" ϕ_{base}, which may be TFT or any other strategy. We call this modified algorithm the *modified composite agent* (MCA), and its pseudo-code is shown in Figure 5. A modified composite agent may be viewed in either of two ways:

- As a composite strategy that can use ϕ_{base} when the composite strategy is insufficient.
- As an enhanced version of ϕ_{base}, in which we modify ϕ_{base}'s moves in cases where the set of traces $\mathcal{T}_B$ might yield a better move. From this viewpoint, $\mathcal{T}_B$ is a case base that is processed by the CIT algorithm so that cases can be selected quickly when they are appropriate.

4. EXPERIMENTAL EVALUATION

We evaluated our technique in three well-known games: the Iterated Prisoner's Dilemma (IPD), the Iterated Chicken Game (ICG), and the Iterated Battle of the Sexes (IBS). The IPD and ICG are the iterated versions of the Prisoner's Dilemma [1] and the Game of Chicken [5], whose payoff matrices are:

Prisoner's Dilemma		Player B	
		Cooperate (C)	Defect (D)
Player A	Cooperate (C)	$(3,3)$	$(0,5)$
	Defect (D)	$(5,0)$	$(1,1)$

Chicken Game		Player B	
		Cooperate (C)	Defect (D)
Player A	Cooperate (C)	$(4,4)$	$(3,5)$
	Defect (D)	$(5,3)$	$(0,0)$

The IBS is the iterated version of the Battle of the Sexes [11], whose payoff matrix is shown below. To allow arbitrary agents to play against each other without having to take on different roles (hence different strategies), we needed to reformulate the IBS to make the roles of Husband and Wife interchangeable. This was quite easy to do, as shown in the following matrix: for the Wife we renamed Football to C and Opera to D; and for the Husband we renamed Football to D and Opera to C.

Battle of the Sexes		Husband	
		D (was Football)	C (was Opera)
Wife	C (was Football)	$(1,2)$	$(0,0)$
	D (was Opera)	$(0,0)$	$(2,1)$

4.1 Experimental Design

To obtain a large collection of agents for the games, we asked the students in several advanced-level AI classes to contribute agents. We did not tell the students the exact number of iterations in each game, but did tell them that it would be at least 50 (in all of our experiments we used 200 iterations). The students contributed 43 IPD agents, 37 ICG agents, and 37 IBS agents. For each game, we also contributed 9 more agents that used the following well-known strategies: ALLC, ALLD, GRIM, NEG, PAVLOV, RAND, STFT, TFT, and TFTT.[5] This gave us a total of 52 IPD agents, 46 ICG agents, and 46 IBS agents. In the rest of this section, we'll call these the *original* agents, to distinguish them from the composite agents generated by our algorithms.

For each agent ϕ, we wanted to find out how much improvement we could get by replacing ϕ with a modified composite agent whose base strategy is ϕ. We investigated this by doing a 5-fold cross-validation experiment that worked as follows.

For each of the three games (IPD, ICG, and IBS), we took our original set of agents for the game, and randomly and evenly partitioned it into five subsets $\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5$. Then we repeated the following steps five times, once for each Φ_i:

1. For the test set Φ_{test}, we chose Φ_i. For the training set Φ_{train}, we chose $\bigcup_{j \neq i} \Phi_j$.

2. We ran a 200-iteration tournament (the details are described below) among the agents in Φ_{train} (with one modification, also described below), and let $\mathcal{T}_{train}$ be the set of all interaction traces recorded in this tournament.

3. For each agent $\phi \in \Phi_{train}$, we played 100 tournaments, each 200 iterations long, involving ϕ and all of the agents in Φ_{test}. We calculated the agent's *average score*, which is equal to $\frac{1}{100 \times |\Phi_{test}|} \sum_{1 \leq l \leq 100} \sum_{\phi_k \in \Phi_{test}} \{$ the score of ϕ when it plays against ϕ_k in the l'th tournament$\}$.

4. Likewise, for each agent $\phi \in \Phi_{train}$, we played 100 tournaments, each 200 iterations long, involving a modified composite agent $\phi' = \text{MCA}(\mathcal{T}_{train}, \Delta, \phi)$ and all of the agents in Φ_{test}, where Δ is a uniform probability distribution over Φ_{train}. Apart from the average score of ϕ', we also recorded the frequency with which ϕ' used its base strategy ϕ.

All of our tournaments were similar to Axelrod's IPD tournaments [1] and the 2005 IPD tournament [8]. Each participant played against every participant including itself (thus a tournament among n agents consists of n^2 iterated games).

One problem in Step 2 is that MCA's input needs to come from deterministic agents, but many of our students' agents were probabilistic (see Table 1). We handled this problem as follows. Each probabilistic agent $\bar{\phi}$ used a random number generator for which one can set a starting seed. By using ten different starting seeds, we created ten determinized versions of $\bar{\phi}$; and we generated interaction traces using the determinized agents rather than $\bar{\phi}$.

Note that we used the determinized agents *only during training*. The modified composite agents generated from the training data are quite capable of being played against probabilistic agents, so we played them against the probabilistic agents in our tests.

4.2 Experimental Results

Table 2 tells how many traces, on average, were collected for each type of game, and how many traces were in the composite strategies generated by CIT.

In each experiment, we calculated 4 average scores for each agent (one for each test sets that did not contain the agent) and 4 average scores for each MCA agent. We repeat the above experiment 100 times using different partitions and random seeds.

[5]These are often used as standard strategies; e.g., they were used as standard entries in the 2005 IPD tournament [8].

Table 1: Among the original agents, how many of each type.

	IPD	ICG	IBS
Deterministic agents	34	22	17
Probabilistic agents	18	24	29

Table 2: Average number of interaction traces collected during the training sessions, before and after removing duplicate traces, and average number of interaction traces in the composite strategies generated by the CIT algorithm.

	IPD	ICG	IBS
Before removing duplicates	29466.0	44117.4	60470.5
After removing duplicates	7452.0	25391.5	29700.8
Composite strategies	171.2	209.6	245.6

Figure 6 shows the agents' *overall average scores*, each of which is an average of 400 average scores. Since each average score is an average of $100 \times |\Phi_{test}|$ scores, each data point in Figure 6 is computed from the scores of $40000 \times n$ games, where n is the average number of agents in the test sets. Hence in the IPD, each data point is computed from the scores of approximately 415769.2 games, and in both the ICG and the IBS each data point is computed from the scores of approximately 367826.1 games. The data file of the experiments can be downloaded at http://www.cs.umd.edu/~chiu/papers/Au08synthesis_data.tar.gz.

Figure 6 includes three graphs, for the IPD, ICG, and IBS, respectively. In each graph, the x axis shows the agents and the y axis shows their scores. The lower line shows the performance of each original agent ϕ (the agents are sorted in order of decreasing overall performance). The upper line shows the average performance of the corresponding MCA agents.

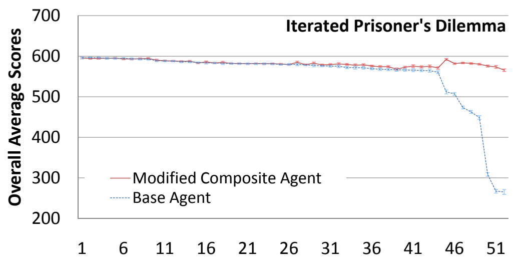
From the graphs, we note the following. In all three games, the average scores of the MCA agents were as good or better than the corresponding base agent. In the IPD, the differences in performance were usually small; and we believe this is because most IPD agents have similar behaviors and therefore leave little room for improvement. In the ICG and IBS, the differences were usually quite large. Finally, in all three games, the MCA agents performed well even when their base agents performed poorly. For example, in the IPD and the IBS, when we incorporated our composite strategy into the weakest of the existing strategies, it more than doubled that strategy's score.

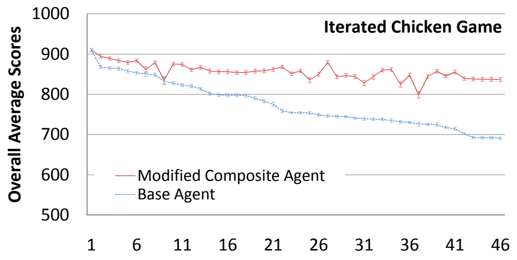
Figure 7 shows the increase in rank of an agent after incorporating the composite strategy into it, while all other agents did not incorporate the composite strategy. We can see that the ranks of most agents increased after the modification. We conducted sign tests to see whether the overall average scores of MCAagents are greater than that of the original agents. The p-values of one-sided sign tests are less than 0.00001 in all games. Therefore, the MCA agents are significantly better at the 99.9% level.

Table 3 shows the average improvement that each MCA agents provided relative to the corresponding original agent. On the average, the percentage improvements in score were about 5% in the IPD, 11% in the ICG, and 26% in the IBS; and the percentage improvements in rank were about 12% in the IPD, 38% in the ICG, and 33% in the IBS.

Table 4 shows how frequently the MCA agents invoked their base agents. We can make the following observations.

- In more than 84% of the IPD games, the MCA agents did not need to use their base strategies at all. In other words, the MCA agents' composite strategies worked successfully throughout those games, even though the games were played with oppo-





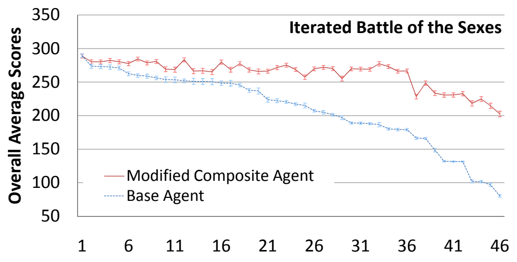


Figure 6: Overall average scores of the base agents and the MCA agents. The agents are displayed on the x axis in order of decreasing score of the base agent. The error bars denote the 95% confidence intervals of the overall average scores.

nents other than the ones used to build the composite strategies. One possible the reason for this high reusability of interaction traces is that the IPD is a well known game and thus most of the original strategies are similar to certain well-known strategies such as Tit-for-Tat.

- In the ICG and IBS, the MCA agents invoked their base strategies a little more than half of the time. We think the reason for this is that there was more diversity among the original strategies, perhaps because these games are not as well-known as IPD. But even though the MCA agents used their composite strategies less frequently than in the IPD, the composite strategies provided much more improvement, relative to the base strategy, than in the IPD. In other words, the places where MCA was used its composite strategy provided a much bigger enhancement to the base strategy's performance.

5. RELATED WORK

Reinforcement learning. One similarity between our approach and reinforcement learning is that our technique improves an agent's performance by incorporating previous experiences. But reinforcement learning agents usually use on-line learning (e.g., Q-learning agents learn and improve their policy during acting), while our composite agent uses off-line learning to produce a composite strategy that can then be incorporated into an agent.

A more important difference is that our technique does not require us to know the set of all possible states of the other agents, as opposed to most existing work on POMDPs or learning automata, in which the set of all possible states must be known beforehand. In open environments such as IPD tournaments in which the opponents' strategies are not known, it would be difficult, if not impossible, to identify all possible states of the opponent's strategy. Moreover, the Markovian assumption intrinsic to a policy or automaton does not always hold because an agent's decision can depend on the entire history in a game. Our paper demonstrates that it is possible to perform well in certain open environments such as the IPD without knowing the set of the opponents' internal states. To the best of our knowledge, contemporary reinforcement learning techniques are, unlike our techniques, not yet efficient enough to compete with strategies such as TFT and Pavlov in the IPD. In future, we would like to run a much wider variety of experiments to see whether our technique can be applicable in other open environments.

Modeling other agents. One approach for playing against a given opponent is to develop an *opponent model* that predicts the opponent's likely behavior [3, 4, 7]. This model is then used to aid in developing strategies against that opponent. In contrast, we focus on generating strategies directly from observations, without constructing an explicit opponent model.

Case-based reasoning. Our technique has some similarities to Derivational Analogy [15] and the reuse of macro actions/operators [9], in which records of a problem-solver's decisions or actions are used to solve new problems. Most work on derivational analogy and macro actions focuses on problems in which there is no need to interact with the environment during problem solving. But there are some works on using derivational analogy or macro actions in interactive environments [2, 12, 14]. In these domains, it would be beneficial not to discard the observation sequences generated by the environments, since the observation sequences capture important information that can be used to determine whether an agent can utilize two different action sequences in

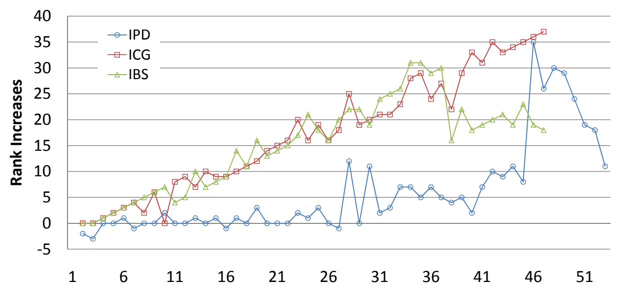


Figure 7: Increase in rank of each enhanced agent, relative to the corresponding base agent. The x axis is as in Figure 6.

Table 3: Average increases in score and rank of the MCA agents, relative to the corresponding original agents.

	IPD	ICG	IBS
Average MCA agent score	582.69	856.37	262.47
Average original agent score	553.62	774.38	208.73
Average difference in score	29.08	81.99	53.74
Average % difference in score	5.3%	10.6%	25.7%
Average increase in rank	6.0	17.4	15.2
Average % increase in rank	11.5%	37.8%	33.0%

Table 4: Average frequency of invocation of base strategies.

	IPD	ICG	IBS
Average percentage of games	15.4%	52.7%	54.4%

the same problem (see Definition 2). Our work pushes this idea further by showing how to construct a strategy from interaction traces.

Case-based reasoning techniques have been used to improve existing reinforcement learning techniques [6, 16]. But so far case-based reasoning played a supporting role only in these work. In contrast, our technique generates fully-functional agents out of previous problem solving experiences, without the help of existing reinforcement learning techniques.

The winner of the 2005 IPD tournament is based on some sort of case-based reasoning technique [10]. Similarly, the winning strategy of our ICG tournament is a combination of three different ways to deal with three different type of opponents. The success of these strategies compels us to believe that one way to dominate monolithic strategies such as Tit-for-Tat is to combine the winning strategies for different type of opponents together.

One problem with the case-based reasoning strategies mentioned in the above paragraph is that the ways they combine strategies together are quite ad hoc, and only manage to consider a handful of possible opponents' strategies. Our work shows a systematic way to combine strategies that can be scaled up to handle a large number of different opponent's strategies.

6. SUMMARY AND FUTURE WORK

The idea that an agent can improve its performance by observing interactions among other agents is not new. What is new in this paper is how to do it without the knowledge of the set of opponents' internal states. In open environments such as IPD tournaments, we know little about the opponents' strategy, let alone the current state of the opponents' strategy. Hence methods that do not require us to know the opponents' states can be quite valuable in such domains.

To avoid using the notion of states or belief states as in policies or automata, our approach directly selects and combines the records of the other agents' interactions to form a partial strategy called a composite strategy, which maximizes the expected utility with respect to an estimated probability distribution of opponents that can possibly generate those interactions, without knowing the states or other details of the strategies of these opponents. The composite strategy can be used to decide how an agent should respond to typical behaviors of opponents in an environment. Out of a set of 126 agents in three iterated games (the IPD, ICG, and IBS), our technique significantly improved the agent's rank (compared to the other agents) after incorporating the partial strategy.

In future, we would like to address the following issues:

- In iterated games such as the IPD, ICG, and IBS, players frequently encounter the game situations that have been seen before; hence combining the interaction traces was sufficient to provide large increases in an agent's performance. By itself, this would not be suitable for large-scale non-iterated games such as chess, where games among experts usually lead to situations that no player has ever seen before. We would like to develop a way to combine our technique with game-tree search, to see how much that extends the scope of the technique.
- A significant limitation of our technique is that it requires an estimate of the probability with which we'll encounter each of opponents we have observed. In the future, we would like to study how to estimate the probability from a database of interaction traces and modify our techniques to alleviate this requirement.
- We've done preliminary tests on some nondeterministic partially-observable planning domains (e.g., transportation domains with unknown traffic patterns). Our approach worked quite well in our tests, but we need to do larger cross-validated experiments before reporting the results. In future, we intend to generalize our method for domains that are more complicated than two-player repeated games.
- Our current formalization cannot handle problems whose horizon is infinite. We would like to see how to extend our algorithm to deal with interaction trace of potentially infinite length.

7. ACKNOWLEDGMENTS

This work was supported in part by AFOSR grants FA95500510298, FA95500610405, and FA95500610295, DARPA's Transfer Learning and Integrated Learning programs, and NSF grants IIS0412812 and IIS0705587. Sarit Kraus is also affiliated with UMIACS. The opinions in this paper are those of the authors and do not necessarily reflect the opinions of the funders.

8. REFERENCES

[1] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[2] S. Bhansali and M. T. Harandi. Synthesis of UNIX programs using derivational analogy. *Machine Learning*, 10:7–55, 1993.

[3] D. Billings, N. Burch, A. Davidson, R. Holte, and J. Schaeffer. Approximating game-theoretic optimal strategies for full-scale poker. In *IJCAI*, pages 661–668, 2003.

[4] J. Denzinger and J. Hamdan. Improving modeling of other agents using tentative stereotypers and compactification of observations. In *Proceedings of the International Conference on Intelligent Agent Technology*, pages 106–112, 2004.

[5] M. Deutsch. *The Resolution of Conflict: Constructive and Destructive Processes*. Yale University Press, 1973.

[6] C. Drummond. Accelerating reinforcement learning by composing solutions of automatically identified subtasks. *JAIR*, 16:59–104, 2002.

[7] D. Egnor. Iocaine powder explained. *ICGA Journal*, 23(1):33–35, 2000.

[8] G. Kendall, X. Yao, and S. Y. Chong. *The Iterated Prisoner's Dilemma: 20 Years On*. World Scientific, 2007.

[9] R. E. Korf. Macro-operators: A weak method for learning. *Artif. Intel.*, 26(1):35–77, 1985.

[10] J. Li. How to design a strategy to win an IPD tournament. In G. Kendall, X. Yao, and S. Y. Chong, editors, *The Iterated Prisoner's Dilemma: 20 Years On*, pages 89–104. World Scientific, 2007.

[11] R. D. Luce and H. Raiffa. *Games and Decisions: Introduction and Critical Survey*. Wiley, 1957.

[12] A. McGovern and R. S. Sutton. Macro-actions in reinforcement learning: An empirical analysis. Technical Report 98-70, University of Massachusetts, Amherst, 1998.

[13] A. Parker, D. Nau, and V. Subrahmanian. Overconfidence or paranoia? search in imperfect-information games. In *AAAI*, July 2006.

[14] D. Precup, R. S. Sutton, and S. Singh. Theoretical results on reinforcement learning with temporally abstract options. In *ECML*, pages 382–393, 1998.

[15] M. M. Veloso and J. G. Carbonell. Derivational analogy in prodigy: Automating case acquisition, storage, and utilization. *Machine Learning*, 10(3):249–278, 1993.

[16] D. Zeng and K. Sycara. Using case-based reasoning as a reinforcement learning framework for optimization with changing criteria. In *ICTAI*, 1995.