5

Paranoia versus Overconfidence in Imperfect-Information Games

AUSTIN PARKER, DANA NAU, AND V.S. SUBRAHMANIAN

Only the paranoid survive.
–Andrew Grove, Intel CEO

Play with supreme confidence, or else you'll lose.
–Joe Paterno, college football coach

1 Introduction

In minimax game-tree search, the *min* part of the minimax backup rule derives from what we will call the *paranoid assumption*: the assumption that the opponent will always choose a move that minimizes our payoff and maximizes his/her payoff (or our estimate of the payoff, if we cut off the search before reaching the end of the game). A potential criticism of this assumption is that the opponent may not have the ability to decide accurately what move this is. But in several decades of experience with game-tree search in chess, checkers, and other zero-sum perfect-information games, the paranoid assumption has worked so well that such criticisms are generally ignored.

In game-tree search algorithms for imperfect-information games, the backup rules are more complicated. Many of them (see Section 6) involve computing a weighted average over the opponent's possible moves (or a Monte Carlo sample of them), where each move's weight is an estimate of the probability that this is the opponent's best possible move. Although such backup rules do not take a *min* at the opponent's move, they still tacitly encode the paranoid assumption, by assuming that the opponent will choose optimally from the set of moves he/she is actually capable of making.

Intuitively, one might expect the paranoid assumption to be less reliable in imperfect-information games than in perfect-information games; for without perfect information, it may be more difficult for the opponent to judge which move is best. The purpose of this paper is to examine whether it is better to err on the side of paranoia or on the side of overconfidence. Our contributions are as follows:

1. **Expected utility.** We provide a recursive formula for the expected utility of a move in an imperfect-information game, that explicitly includes the opponent's strategy σ. We prove the formula's correctness.

2. **Information-set search.** We describe a game-tree search algorithm called *information-set search* that implements the above formula. We show analytically

that with an accurate opponent model, information-set search produces optimal results.

3. **Approximation algorithm.** Information-set search is, of course, intractable for any game of interest as the decision problem in an imperfect-information game is complete in double exponential time [Reif 1984]. To address this intractability problem, we provide a modified version of information-set search that computes an approximation of a move's expected utility by combining Monte Carlo sampling of the belief state with a limited-depth search and a static evaluation function.

4. **Paranoia and overconfidence.** We present two special cases of the expected-utility formula (and hence of the algorithm) that derive from two different opponent models: the *paranoid* model, which assumes the opponent will always make his/her best possible move, and the *overconfident* model, which assumes the opponent will make moves at random.

5. **Experimental results.** We provide experimental evaluations of information-set search in several different imperfect-information games. These include imperfect-information versions of P-games [Pearl 1981; Nau 1982a; Pearl 1984], N-games [Nau 1982a], and kalah [Murray 1952]; and an imperfect-information version of chess called kriegspiel [Li 1994; Li 1995]. Our main experimental results are:

    - Information-set search outperformed HS, the best of our algorithms for kriegspiel in [Parker, Nau, and Subrahmanian 2005].
    - In all of the games, the overconfident opponent model outperformed the paranoid model. The difference in performance became more marked when we decreased the amount of information available to each player.

This work was influenced by Judea Pearl's invention of P-games [Pearl 1981; Pearl 1984], and his suggestion of investigating backup rules other than minimax [Pearl 1984]. We also are grateful for his encouragement of the second author's early work on game-tree search (e.g., [Nau 1982a; Nau 1983]).

2 Basics

Our definitions and notation are based on [Osborne and Rubinstein 1994]. We consider games having the following characteristics: two players, finitely many moves and states, determinism, turn taking, zero-sum utilities, imperfect information expressed via *information sets* (explained in Section 2.1), and *perfect recall* (explained in Section 2.3). We will let G be any such game, and a_1 and a_2 be the two players. Our techniques are generalizable to stochastic multi-player non-zero-sum games,[1] but that is left for future work.

[1]Nondeterministic initial states, outcomes, and observations can be modeled by introducing an additional player a_0 who makes a nondeterministic move at the start of the game and after each of the other players' moves. To avoid affecting the other players' payoffs, a_0's payoff in terminal states is always 0.

At each state s, let $a(s)$ be the player to move at s, with $a(s) = \emptyset$ if the game is over in s. Let $M(s)$ be the set of available moves at s, and $m(s)$ be the state produced by making move m in state s. A *history* is a sequence of moves $h = \langle m_1, m_2, \ldots, m_j \rangle$. We let $s(h)$ be the state produced by history h, and when clear from context, will abuse notation and use h to represent $s(h)$ (e.g., $m(h) = m(s(h))$). Histories in which the game has ended are called *terminal*. We let H be the set of all possible histories for game G.

2.1 Information Sets

Intuitively, an information set is a set of histories that are indistinguishable to a player a_i, in the sense that each history h provides a_i with the same sequence of observations. For example, suppose a_1 knows the entire sequence of moves that have been played so far, except for a_2's last move. If there are two possibilities for a_2's last move, then a_1's information set includes two histories, one for each of the two moves.

In formalizing the above notion, we will not bother to give a full formal definition of an "observation." The only properties we need for an observation are the following:[2]

- We assume that each player a_i's sequence of observations is a function $O_i(h)$ of the current history h. The rationale is that if a_1 and a_2 play some game a second time, and if they both make the same moves that they made the first time, then they should be able to observe the same things that they observed the first time.

- We assume that when two histories h, h' produce the same sequence of observations, they also produce the same set of available moves, i.e., if $O_i(h) = O_i(h')$, then $M(s(h)) = M(s(h'))$. The rationale for this is that if the current history is h, a_i's observations won't tell a_i whether the history is h or h', so a_i may attempt to make a move m that is applicable in $s(h')$ but not in $s(h)$. If a_i does so, then m will produce *some* kind of outcome, even if the outcome is just an announcement that a_i must try a different move. Consequently, we can easily make m applicable in $s(h)$, by defining a new state $m(s(h))$ in which this outcome occurs.

- We assume that terminal histories with distinct utilities always provide distinct observations, i.e., for terminal histories $h, h' \in T$, if $U_i(h) \neq U_i(h')$ then $O_i(h) \neq O_i(h')$.

We define a_i's information set for h to be the set of all histories that give a_i the same observations that h gives, i.e., $[h]_i = \{h' \in H : O_i(h') = O_i(h)\}$. The set of all possible information sets for a_i is $\mathcal{I}_i = \{[h]_i : h \in H\}$. It is easy to show that $\mathcal{I}_i$ is a partition of H.

Figure 1 shows an example game tree illustrating the correspondence between information sets and histories. In that game, player a_1 makes the first move, which is hidden to player a_2. Thus player a_2 knows that the history is either $\langle L \rangle$ or $\langle R \rangle$, which is denoted by putting a dotted box around the nodes for those histories.

[2] Some game-theory textbooks define information sets without even using the notion of an "observation." They simply let a player's information sets be the equivalence classes of a partition over the set of possible histories.

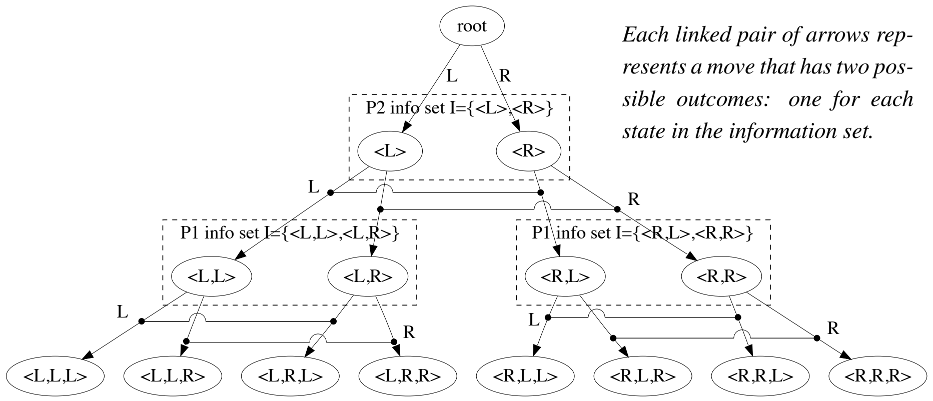

root
L
R
P2 info set I={<L>,<R>}
<L>
<R>
L
R
P1 info set I={<L,L>,<L,R>}
P1 info set I={<R,L>,<R,R>}
<L,L>
<L,R>
<R,L>
<R,R>
L
R
L
R
<L,L,L>
<L,L,R>
<L,R,L>
<L,R,R>
<R,L,L>
<R,L,R>
<R,R,L>
<R,R,R>
Each linked pair of arrows represents a move that has two possible outcomes: one for each state in the information set.

Figure 1. A game tree for a two-player imperfect-information game between two players P1 and P2 who move in alternation. The players may move either left (L) or right (R), and their moves are hidden from each other (e.g., after P1's first move, P2 knows that P1 has moved, but not whether the move was L or R). Each node is labeled with its associated history (e.g., $\langle L \rangle$ and $\langle R \rangle$ for the two children of the root node). The information set of the player to move is indicated by a dotted box (e.g., after P1's first move, P2's information set is $\{\langle L \rangle, \langle R \rangle\}$).

2.2 Strategies

In a perfect-information game, a player a_i's *strategy* is a function $\sigma_i(m|s)$ that returns the probability p that a_i will make move m in state s. For imperfect-information games, where a_i will not always know the exact state he/she is in, σ_i is a function of an information set rather than a state; hence $\sigma_i(m|I)$ is the probability that a_i will make move m when their information set is I. We let $M(I)$ be the set of moves available in information set I.

If σ_i is a mixed strategy, then for every information set $I \in \mathcal{I}_i$ where it is a_i's move, there may be more than one move $m \in M(I)$ for which $\sigma_i(m|I) > 0$. But if σ_i is a pure strategy, then there will be a unique move $m_I \in M(I)$ such that $\sigma_i(m|I) = 0 \; \forall m \neq m_I$ and $\sigma_i(m_I|I) = 1$; and in this case we will use the notation $\sigma_i(I)$ to refer to m_I.

If $h = \langle m_1, m_2, \ldots, m_n \rangle$ is a history, then its probability $P(h)$ can be calculated from the players' strategies. Suppose a_1's and a_2's strategies are σ_1 and σ_2. In the special case where a_1 has the first move and the players move in strict alternation,

$$P(h|\sigma_1, \sigma_2) = \sigma_1(m_1|h_0)\sigma_2(m_2|h_1)\ldots\sigma_1(m_j|h_{j-1}), \sigma_2(m_{j+1}|h_j), \ldots, \tag{1}$$

where $h_j = \langle m_1, \ldots, m_j \rangle$ ($h_0 = \langle\rangle$). More generally,

$$P(h|\sigma_1, \sigma_2) = \prod_{j=0}^{n-1} \sigma_{a(h_j)}(m_{j+1}|h_j). \tag{2}$$

Given σ_1, σ_2, and any information set I, the conditional probability of any $h \in I$ is the normalized probability

$$P(h|I, \sigma_1, \sigma_2) = \frac{P(h|\sigma_1, \sigma_2)}{\sum_{h' \in I} P(h'|\sigma_1, \sigma_2)}. \tag{3}$$

2.3 Perfect Recall

Perfect recall means that every player always remembers all the moves they've made – we can't have two histories in player a_i's information set which disagree on what player a_i did at some point in the past. One can get a more detailed explanation of perfect and imperfect recall in perfect information games in [Osborne and Rubinstein 1994].

In a game of perfect recall, it is easy to show that if $I \in \mathcal{I}_1$, then all histories in I have the same sequence of moves for a_1, whence the probability of h given I is conditionally independent of σ_1. If $h = \langle m_1, m_2, \ldots, m_n \rangle$, then

$$P(h|I, \sigma_1, \sigma_2) = P(h|I, \sigma_2) = \frac{\prod_{a(h_j)=a_2} \sigma_2(m_{j+1}|h_j)}{\sum_{h' \in I} \prod_{a(h'_j)=a_2} \sigma_2(m_{j+1}|h'_j)}. \tag{4}$$

An analogous result, with the subscripts 1 and 2 interchanged, holds when $I \in \mathcal{I}_2$.

2.4 Utility and Expected Utility

If a history h takes us to the game's end, then h is *terminal*, and we let $U(h)$ be the *utility* of h for player a_1. Since the game is zero-sum, it follows that a_2's utility is $-U(h)$.

If a_1 and a_2 have strategies σ_1 and σ_2, then the expected utility for a_i is

$$EU(\sigma_1, \sigma_2) = \sum_{h \in T} P(h|\sigma_1, \sigma_2) U(h), \tag{5}$$

where T is the set of all terminal histories, and $P(h|\sigma_1, \sigma_2)$ is as in Eq. (2). Since the game is zero-sum, it follows that a_2's expected utility is $-EU(\sigma_1, \sigma_2)$.

For the expected utility of an individual history h, there are two cases:

Case 1: History h is terminal. Then h's expected utility is just its actual utility, i.e.,

$$EU(h|\sigma_1, \sigma_2) = EU(h) = U(h). \tag{6}$$

Case 2: History h ends at a state where it is a_i's move. Then h's expected utility is a weighted sum of the expected utilities for each of a_i's possible moves, weighted by the probabilities of a_i making those moves:

$$\begin{aligned} EU(h|\sigma_1, \sigma_2) &= \sum_{m \in M(h)} \sigma_i(m|h) \cdot EU(h \circ m|\sigma_1, \sigma_2) \\ &= \sum_{m \in M(h)} \sigma_i(m|[h]_i) \cdot EU(h \circ m|\sigma_1, \sigma_2), \end{aligned} \tag{7}$$

where $\circ$ denotes concatenation.

The following lemma shows that the recursive formulation in Eqs. (6–7) matches the notion of expected utility given in Eq. 5.

LEMMA 1. *For any strategies σ_1 and σ_2, $EU(\langle\rangle|\sigma_1, \sigma_2)$ (the expected utility of the empty initial history as computed via the recursive Equations 6 and 7) equals $EU(\sigma_1, \sigma_2)$.*

Sketch of proof. This is shown by showing, by induction on the length of h, the more general statement that

$$EU(h|\sigma_1, \sigma_2) = \sum_{h' \in T, h' = h \circ m_k, \circ \cdots \circ, m_n} P(h'|\sigma_1, \sigma_2) U(h') / P(h|\sigma_1, \sigma_2), \quad (8)$$

where k is one greater than the size of h and n is the size of each h' as appropriate. The base case occurs when h is terminal, and the inductive case assumes Eq. 8 holds for histories of length $m+1$ to show algebraically that Eq. 8 holds for histories of length m. □

The expected utility of an information set $I \in H$ is the weighted sum of the expected utilities of its histories:

$$EU(I|\sigma_1, \sigma_2) = \sum_{h \in I} P(h|I, \sigma_1, \sigma_2) EU(h|\sigma_1, \sigma_2). \quad (9)$$

COROLLARY 2. *For any strategies σ_1 and σ_2, and player a_i, $EU([\langle\rangle]_i|\sigma_1, \sigma_2)$ (the expected utility of the initial information set for player a_i) equals $EU(\sigma_1, \sigma_2)$.*

3 Finding a Strategy

We now develop the theory for a game-tree search technique that exploits an opponent model.

3.1 Optimal Strategy

Suppose a_1's and a_2's strategies are σ_1 and σ_2, and let I be any information set for a_1. Let $M^*(I|\sigma_1, \sigma_2)$ be the set of all moves in $M(I)$ that maximize a_1's expected utility at I, i.e.,

$$\begin{aligned} M^*(I|\sigma_1, \sigma_2) &= \operatorname*{argmax}_{m \in M(I)} EU(I \circ m|\sigma_1, \sigma_2) \\ &= \left\{ m^* \in M(I) \;\middle|\; \begin{array}{r} \forall m \in M(I),\ \sum_{h \in I} P(h|I, \sigma_1, \sigma_2) EU(h \circ m^*|\sigma_1, \sigma_2) \\ \geq\ \sum_{h \in I} P(h|I, \sigma_1, \sigma_2) EU(h \circ m|\sigma_1, \sigma_2) \end{array} \right\}. \end{aligned} \quad (10)$$

Since we are considering only finite games, every history has finite length. Thus by starting at the terminal states and going backwards up the game tree, applying Eqs. (7) and (9) at each move, one can compute a strategy σ_1^* such that:

$$\sigma_1^*(m|I) = \begin{cases} 1/|M^*(I, \sigma_1^*, \sigma_2)|, & \text{if } m \in M^*(I|\sigma_1^*, \sigma_2), \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

THEOREM 3. *Let σ_2 be a strategy for a_2, and σ_1^* be as in Eq. (11). Then σ_1^* is σ_2-optimal.*

Sketch of proof. Let $\bar{\sigma}_1$ be any σ_2-optimal strategy. The basic idea is to show, by induction on the lengths of histories in an information set I, that $EU(I|\sigma_1^*, \sigma_2) \geq EU(I|\bar{\sigma}_1, \sigma_2)$.

The induction goes backwards from the end of the game: the base case is where I contains histories of maximal length, while the inductive case assumes the inequality holds when I contains histories of length $k+1$, and shows it holds when I contains histories of length k. The induction suffices to show that $EU([\langle\rangle]_1|\sigma_1^*, \sigma_2) \geq EU([\langle\rangle]_1|\bar{\sigma}_1, \sigma_2)$, whence from Lemma 1, $EU(\sigma_1^*, \sigma_2) \geq EU(\bar{\sigma}_1, \sigma_2)$. □

Computing σ_1^* is more difficult than computing an optimal strategy in a perfect-information game. Reif [Reif 1984] has shown that the problem of finding a strategy with a guaranteed win is doubly exponential for imperfect-information games (this corresponds to finding σ_1 such that for all σ_2, σ_1 wins).

In the minimax game-tree search algorithms used in perfect-information games, one way of dealing with the problem of intractability is to approximate the utility value of a state by searching to some limited depth d, using a *static evaluation function* $\mathcal{E}(\cdot)$ that returns approximations of the expected utilities of the nodes at that depth, and pretending that the values returned by $\mathcal{E}$ are the nodes' actual utility values. In imperfect-information games we can compute approximate values for EU in a similar fashion:

$$EU_d(h|\sigma_1^*,\sigma_2) = \begin{cases} \mathcal{E}(h), & \text{if } d=0, \\ U(h), & \text{if } h \text{ is terminal}, \\ \sum_{m\in M(h)} \sigma_2(m|[h]_2)\cdot EU_{d-1}(h\circ m|\sigma_1^*,\sigma_2), & \text{if it's } a_2\text{'s move}, \\ EU_{d-1}(h\circ \operatorname{argmax}_{m\in M(h)}(EU_d([h\circ m]_1|\sigma_1^*,\sigma_2))), & \text{if it's } a_1\text{'s move}, \end{cases} \tag{12}$$

$$EU_d(I|\sigma_1^*,\sigma_2) = \sum_{h\in I} P(h|I,\sigma_1^*,\sigma_2)\cdot EU_d(h|I,\sigma_1^*,\sigma_2). \tag{13}$$

3.2 Opponent Models

Eqs. (11–12) assume that a_1 knows a_2's strategy σ_2, an assumption that is quite unrealistic in practice. A more realistic assumption is that a_1 has a *model* of a_2 that provides an approximation of σ_2. For example, in perfect-information games, the well-known minimax formula corresponds to an opponent model in which the opponent always chooses the move whose utility value is lowest. We now consider two opponent models for imperfect-information games: the *overconfident* and *paranoid* models.

Overconfidence. The overconfident model assumes a_2 is just choosing moves at random from a uniform distribution; i.e., it assumes a_2's strategy is $\sigma_2(m|I) = 1/|M(I)|$ for every $m \in M(I)$, and second, that a_1's strategy is σ_2-optimal. If we let $OU_d(h) = EU_d(h|\sigma_1^*,\sigma_2)$ and $OU_d(I) = EU_d(I|\sigma_1^*,\sigma_2)$ be the expected utilities for histories and information sets under these assumptions, then it follows from Eqs. (12–13) that:

$$OU_d(h) = \begin{cases} \mathcal{E}(h), & \text{if } d=0, \\ U(h), & \text{if } h \text{ is terminal}, \\ \sum_{m\in M(h)} \frac{OU_{d-1}(h\circ m)}{|M(h)|}, & \text{if it's } a_2\text{'s move}, \\ OU_{d-1}(h\circ \operatorname{argmax}_{m\in M(h)} OU_d([h\circ m]_1)), & \text{if it's } a_1\text{'s move}, \end{cases} \tag{14}$$

$$OU_d(I) = \sum_{h\in I} (1/|I|)\cdot OU_d(h). \tag{15}$$

If the algorithm searches to a limited depth (Eq. 12 with $d < \max_{h\in H}|h|$), we will refer to the resulting strategy as *limited-depth overconfident*. If the algorithm searches to the end of the game (i.e., $d \geq \max_{h \in H} |h|$), we will refer to the resulting strategy as *full-depth overconfident*; and in this case we will usually write $OU(h)$ rather than $OU_d(h)$.

Paranoia. The paranoid model assumes that a_2 will always make the worst possible move for a_1, i.e., the move that will produce the minimum expected utility over all of the histories in a_1's information set. This model replaces the summation in the third line of Eq. (12) with a minimization:

$$PU_d(h) = \begin{cases} \mathcal{E}(I), & \text{if } d = 0, \\ U(h), & \text{if } h \text{ is terminal}, \\ PU_{d-1}(h \circ \operatorname{argmin}_{m \in M(h)}(\min_{h' \in [h]_1} PU_d([h \circ m]))), & \text{if it's } a_2\text{'s move}, \\ PU_{d-1}(h \circ \operatorname{argmax}_{m \in M(h)}(\min_{h' \in [h]_1} PU_d([h \circ m]))), & \text{if it's } a_1\text{'s move}, \end{cases} \tag{16}$$

$$PU_d(I) = \min_{h \in I} PU_d(h). \tag{17}$$

Like we did for overconfident search, we will use the terms *limited-depth* and *full-depth* to refer to the cases where $d < \max_{h \in H} |h|$ and $d \geq \max_{h \in H} |h|$, respectively; and for a full-depth paranoid search, we will usually write $PU(h)$ rather than $PU_d(h)$.

In perfect-information games, $PU(h)$ equals h's minimax value. But in imperfect-information games, h's minimax value is the minimum Eq. (11) over all possible values of σ_2; and consequently $PU(h)$ may be less than h's minimax value.

3.3 Comparison with the Minimax Theorem

The best known kinds of strategies for zero-sum games are the strategies based on the famous Minimax Theorem [von Neumann and Morgenstern 1944]. These *minimax strategies* tacitly incorporate an opponent model that we will call the *minimax model*. The minimax model, overconfident model, and paranoid model each correspond to differing assumptions about a_2's knowledge and competence, as we will now discuss.

Let Σ_1 and Σ_2 be the sets of all possible pure strategies for a_1 and a_2, respectively. If a_1 and a_2 use mixed strategies, then these are probability distributions P_1 and P_2 over Σ_1 and Σ_2. During game play, a_1 and a_2 will randomly choose pure strategies σ_1 and σ_2 from P_1 and P_2. Generally they will do this piecemeal by choosing moves as the game progresses, but game-theoretically this is equivalent to choosing the entire strategy all at once.

Paranoia: If a_1 uses a paranoid opponent model, this is equivalent to assuming that a_2 knows in advance the pure strategy σ_1 that a_1 will choose from P_1 during the course of the game, and that a_2 can choose the optimal counter-strategy, i.e., a strategy $P_2^{\sigma_1}$ that minimizes σ_1's expected utility. Thus a_1 will want to choose a σ_1 that has the highest possible expected utility given $P_2^{\sigma_1}$. If there is more than one such σ_1, then a_1's strategy can be any one of them or can be an arbitrary probability distribution over all of them.

Minimax: If a_2 uses a minimax opponent model, this is equivalent to assuming that a_2 will know in advance what a_1's mixed strategy P_1 is, and that a_2 will be competent enough to choose the optimal counter-strategy, i.e., a mixed strategy $P_2^{P_1}$ that minimizes

P_1's expected utility. Thus a_1 will want to use a mixed strategy P_1 that has the highest possible expected utility given $P_2^{P_1}$.

In perfect-information games, the minimax model is equivalent to the paranoid model. But in imperfect-information games, the minimax model assumes a_2 has less information than the paranoid model does: the minimax model assumes that a_2 knows the probability distribution P_1 over a_1's possible strategies, and the paranoid model assumes that a_2 knows which strategy a_1 will choose from P_1.

Overconfidence: If a_1 uses an overconfident opponent model, this equivalent to assuming that a_2 knows nothing about (or is not competent enough to figure out) how good or bad each move is, whence a_2 will use a strategy $P_2^=$ in which all moves are equally likely. In this case, a_1 will want to choose a strategy σ_1 that has the highest expected utility given $P_2^=$. If there is more than one such σ_1, then a_1's strategy can be any one of them or can be an arbitrary probability distribution over all of them.

In both perfect- and imperfect-information games, the overconfident model assumes a_2 has much less information (and/or competence) than in the minimax and paranoid models.

3.4 Handling Large Information Sets

Information sets can be quite large. When they are too large for techniques like the above to run in a reasonable amount of time, there are several options.

Game simplification reduces the size of the information set by creating an analogous game with smaller information sets. This technique has worked particularly well in poker [Billings, Burch, Davidson, Holte, Schaeffer, Schauenberg, and Szafron 2003; Gilpin and Sandholm 2006a; Gilpin and Sandholm 2006b], as it is possible to create a "simpler" game which preserves win probabilities (within some ϵ). However, these approaches apply only to variants of poker, and the technique is not easily generalizable. Given an arbitrary game G other than poker, we know of no general-purpose way of producing a simpler game whose expected utilities accurately reflect expected utilities in G.

State aggregation was first used in the game of sprouts [Applegate, Jacobson, and Sleator 1991], and subsequently has been used in computer programs for games such as bridge (e.g., [Ginsberg 1999]), in which many of the histories in an information set are similar, and hence can be reasoned about as a group rather than individually. For example, if one of our opponents has an ace of hearts and a low heart, it usually does not matter *which* low heart the opponent has: generally all low hearts will lead to an identical outcome, so we need not consider them separately. The aggregation reduces the computational complexity by handling whole sets of game histories in the information set at the same time. However, just as with game simplification, such aggregation techniques are highly game dependent. Given an arbitrary game G, we do not know of a general-purpose way to aggregate states of G in a way that is useful for computing expected utility values in G.

Unlike the previous two techniques, **statistical sampling** [Corlett and Todd 1985] is general enough to fit any imperfect-information game. It works by selecting a manageable subset of the given, large, information set, and doing our computations based on that.

Since we are examining game playing across several imperfect-information games we use the third technique. Let us suppose Γ is an expected utility function such as OU_d or PU_d. In statistical sampling we pick $I' \subset I$ and compute the value of $\Gamma(I')$ in place of $\Gamma(I)$. There are two basic algorithms for doing the sampling:

1. **Batch:** Pick a random set of histories $I' \subset I$, and compute $\Gamma_s(I')$ using the equations given earlier.

2. **Iterative:** Until the available time runs out, repeatedly pick a random $h \in I$, compute $\Gamma(\{h\})$ and aggregate that result with all previous picks.

The iterative method is preferable because it is a true anytime algorithm: it continues to produce increasingly accurate estimates of $\Gamma(I)$ until no more time is available. In contrast, the batch method requires guessing how many histories we will be able to compute in that time, picking a subset I' of the appropriate size, and hoping that the computation finishes before time is up. For more on the relative advantages of iterative and batch sampling, see [Russell and Wolfe 2005].

Statistical sampling, unlike game simplification and state aggregation, can be used for arbitrary imperfect-information games rather than just on games that satisfy special properties. Consequently, it is what we use in our experiments in Section 5.

4 Analysis

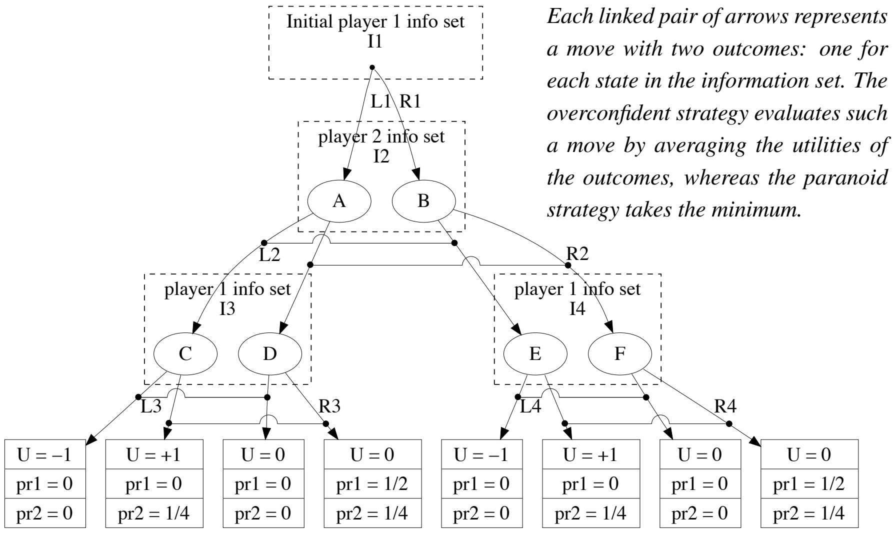
Since paranoid and overconfident play both depend on opponent models that may be unrealistic, which of them is better in practice? The answer is not completely obvious. Even in games where each player's moves are completely hidden from the other player, it is not hard to create games in which the paranoid strategy outplays the overconfident strategy and vice-versa. We now give examples of games with these properties.

Figures 2 and 3, respectively, are examples of situations in which paranoid play outperforms overconfident play and vice versa. As in Figure 1, the games are shown in tree form in which each dotted box represents an information set. At each leaf node, U is the payoff for player 1. Based on these values of U, the table gives, the probabilities of moving left (L) and right (R) at each information set in the tree, for both the overconfident and paranoid strategies. At each leaf node, pr1 is the probability of reaching that node when player 1 is overconfident and player 2 is paranoid, and pr2 is the probability of reaching that node when player 2 is overconfident and player 1 is paranoid.

In Figure 2, the paranoid strategy outperforms the overconfident strategy, because of the differing choices the strategies will make at the information set I2:

- Suppose player 1 is overconfident and player 2 is paranoid. Then at information set I2, player 2 assumes its opponent will always choose the worst possible response. Hence when choosing a move at I2, player 2 thinks it will lose if it chooses L2 and will tie if it chooses R2, so it chooses R2 to avoid the anticipated loss.

- Suppose player 1 is paranoid and player 2 is overconfident. Then at information set I2, player 2 assumes its opponent is equally likely to move left or right. Hence when


Initial player 1 info set
I1
L1
R1
player 2 info set
I2
A
B
L2
R2
player 1 info set
I3
C
D
player 1 info set
I4
E
F
L3
R3
L4
R4
U = −1
pr1 = 0
pr2 = 0
U = +1
pr1 = 0
pr2 = 1/4
U = 0
pr1 = 0
pr2 = 0
U = 0
pr1 = 1/2
pr2 = 1/4
U = −1
pr1 = 0
pr2 = 0
U = +1
pr1 = 0
pr2 = 1/4
U = 0
pr1 = 0
pr2 = 0
U = 0
pr1 = 1/2
pr2 = 1/4
Each linked pair of arrows represents a move with two outcomes: one for each state in the information set. The overconfident strategy evaluates such a move by averaging the utilities of the outcomes, whereas the paranoid strategy takes the minimum.

Info set	Overconfident strategy		Paranoid strategy	
I1	P(L1) = 1/2	P(R1) = 1/2	P(L1) = 1/2	P(R1) = 1/2
I2	P(L2) = 1/2	P(R2) = 1/2	P(L2) = 0	P(R2) = 1
I3	P(L3) = 0	P(R3) = 1	P(L3) = 0	P(R3) = 1
I4	P(L4) = 0	P(R4) = 1	P(L4) = 0	P(R4) = 1

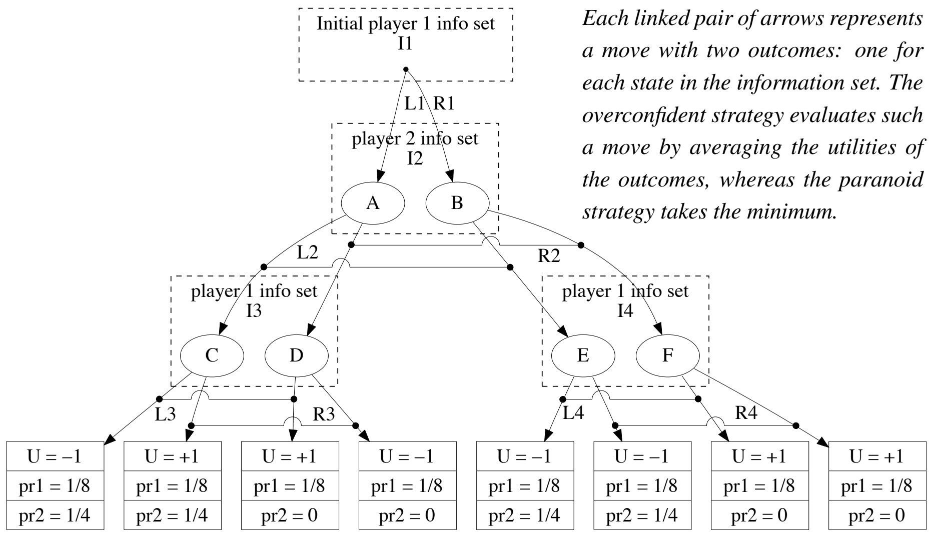
Figure 2. An imperfect-information game in which paranoid play beats overconfident play. If an overconfident player plays against a paranoid player and each player has an equal chance of moving first, the expected utilities are -0.25 for the overconfident player and 0.25 for the paranoid player.

choosing a move at I2, player 2 thinks that both moves have the same expected utility, so it will choose between them at random—which is a mistake, because its paranoid opponent will win the game by moving right in both information sets I3 and I4.

Figure 3 shows a game in which the overconfident strategy outperforms the paranoid strategy. Again, the pertinent information set is I2:

- Suppose overconfident play is player 1 and paranoid play is player 2. Then paranoid play, assuming the worst, believes both move L2 and R2 are losses. R2 is a loss because the opponent may have made move R1 resulting in a forced loss for player 2 at node F, and L2 is a loss because the opponent may have made move L1 and then may make move R4 resulting in a loss for player 2. Since there is a potential loss in all cases, paranoid play chooses both cases with equal probability.

- When overconfident play is player 2, it makes move L2 at I2, on the theory that the opponent was equally likely to make moves L1 and R1 and therefore giving it a 50% probability of ending up in node E, which is a forced win for player 2. Against


Initial player 1 info set
I1
L1
R1
player 2 info set
I2
A
B
L2
R2
player 1 info set
I3
C
D
player 1 info set
I4
E
F
L3
R3
L4
R4
U = −1
pr1 = 1/8
pr2 = 1/4
U = +1
pr1 = 1/8
pr2 = 1/4
U = +1
pr1 = 1/8
pr2 = 0
U = −1
pr1 = 1/8
pr2 = 0
U = −1
pr1 = 1/8
pr2 = 1/4
U = −1
pr1 = 1/8
pr2 = 1/4
U = +1
pr1 = 1/8
pr2 = 0
U = +1
pr1 = 1/8
pr2 = 0
Each linked pair of arrows represents a move with two outcomes: one for each state in the information set. The overconfident strategy evaluates such a move by averaging the utilities of the outcomes, whereas the paranoid strategy takes the minimum.

Info set	Overconfident strategy		Paranoid strategy	
I1	P(L1) = 1/2	P(R1) = 1/2	P(L1) = 1/2	P(R1) = 1/2
I2	P(L2) = 1	P(R2) = 0	P(L2) = 1/2	P(R2) = 1/2
I3	P(L3) = 1/2	P(R3) = 1/2	P(L3) = 1/2	P(R3) = 1/2
I4	P(L4) = 1/2	P(R4) = 1/2	P(L4) = 1/2	P(R4) = 1/2

Figure 3. An imperfect-information game where overconfident play beats paranoid play. If an overconfident player plays against a paranoid player and each player has an equal chance of moving first, the expected utilities are 0.25 for the overconfident player and -0.25 for the paranoid player.

paranoid play as player 1, this is a good move, since paranoid play actually does make moves L1 and R1 with 50% probability.

These two examples show that neither strategy is guaranteed to be better in all cases: sometimes paranoid play outperforms overconfident play, and sometimes vice versa. So to determine their relative worth, deeper analysis is necessary.

4.1 Analysis of Overconfidence Performance in Perfect Information Games

Let s be a state in a perfect-information zero-sum game. We will say that a child s' of s is *minimax-optimal* if $\mu(s') \geq \mu(s'')$ for every child s'' of s, where $\mu(s)$ is the minimax value for the player to move at s. A *minimax strategy* is any strategy that will always move to a minimax-optimal node. In the game-tree search literature, minimax strategies have often been called "perfect play" because they produce the highest possible value against an opponent who is also using a minimax strategy.

In perfect-information zero-sum games, $PU(s) = \mu(s)$ at every state s, hence full-depth paranoid play is a minimax strategy. Surprisingly, if the only outcomes are wins and losses (or equivalently, utility values of 1 and -1), full-depth *overconfident* play is also a minimax strategy. To prove this result, we first need a lemma:

LEMMA 4. *Let G be any finite two-player perfect-information game whose outcomes all have utility 1 or -1. At every state s, if $\mu(s) = 1$ then $OC(s) = 1$, and if $\mu(s) = -1$ then $OC(s) \in [-1, 1)$.*

Sketch of proof. This is proven by induction on the height of the state s under consideration. The base case occurs for with terminal nodes of height 0 for which the lemma follows trivially. The inductive case supposes the lemma holds for all states of height k and shows algebraically for states s of height $k+1$ in each of four possible cases: (1) if it is a_1's move and $\mu(s) = -1$ then $OC(s) \in [-1, 1)$, (2) if it is a_1's move and $\mu(s) = 1$ then $OC(s) = 1$, (3) if it is a_2's move and $\mu(s) = -1$ then $OC(s) \in [-1, 1)$, and (4) if it is a_2's move and $\mu(s) = 1$ then $OC(s) = 1$. Since the game allows only wins and losses (so that $\mu(s)$ is 1 or -1), these are all the possibilities. $\square$

THEOREM 5. *Let G be any finite two-player perfect-information game whose outcomes all have utility 1 or -1. At every nonterminal state s, the overconfident strategy, σ_O, will move to a state s' that is minimax-optimal.*

Proof. Immediate from the lemma. $\square$

This theorem says that in head-to-head play in perfect-information games allowing only wins or losses, the full-depth overconfident and full-depth paranoid strategies will be evenly matched. In the experimental section, we will see this to hold in practice.

4.2 Discussion

Paranoid play. When using paranoid play a_1 assumes that a_2 has always and will always make the worst move possible for a_1, but a_1 does this *given only a_1's information set*. This means that for any given information set, the paranoid player will find the history in the information set that is least advantageous to itself and make moves as though that were the game's actual history *even when the game's actual history is any other member of the information set*. There is a certain intuitively appealing protectionism occurring here: an opponent that happens to have made the perfect moves cannot trap the paranoid player. However, it really is not clear exactly how well a paranoid player will do in an imperfect-information game, for the following reasons:

- There is no reason to necessarily believe that the opponent has made those "perfect" moves. In imperfect-information games, the opponent has different information than the paranoid player, which may not give the opponent enough information to make the perfect moves paranoid play expects.

- Against non-perfect players, the paranoid player may lose a lot of potentially winnable games. The information set could contain thousands of histories in which a particular move m is a win; if that move is a loss on just one history, and there is another move m' which admits no losses (and no wins), then m will not be chosen.[3]

- In games such as kriegspiel, in which there are large and diverse information sets, usually every information set will contain histories that are losses, hence paranoid play will evaluate all of the information sets as losses. In this case, all moves will look equally terrible to the paranoid player, and paranoid play becomes equivalent to random play.[4]

We should also note the relationship paranoid play has to the "imperfection" of the information in the game. A game with large amounts of information and small information sets should see better play from a paranoid player than a game with large information sets. The reason for this is that as we get more information about the actual game state, we can be more confident that the move the paranoid player designates as "worst" is a move the opponent can discover and make in the actual game. The extreme of this is a perfect information game, where paranoid play has proven quite effective: it is minimax search. But without some experimentation, it is not clear to what extent smaller amounts of information degrade paranoid play.

Overconfident play. Overconfident play assumes that a_2 will, with equal probability, make all available moves regardless of what the available information tells a_2 about each move's expected utility. The effect this has on game play depends on the extent to which a_2's moves diverge from random play. Unfortunately for overconfidence, many interesting imperfect-information games implicitly encourage non-random play. In these games the overconfident player will not adequately consider the risks of its moves. The overconfident player, acting under the theory that the opponent is unlikely to make a particular move, will many times not protect itself from a potential loss.

However, depending on the amount of information in the imperfect-information game, the above problem may not be as bad as it seems. For example, consider a situation where a_1, playing overconfidently, assumes the opponent is equally likely to make each of the ten moves available in a_1's current information set. Suppose that each move is clearly the best move in exactly one tenth of the available histories. Then, despite the fact that the opponent is playing a deterministic strategy, random play is a good opponent model given the information set. This sort of situation, where the model of random play is reasonable despite it being not at all related to the opponent's actual mixed strategy, is more likely to occur in games where there is less information. The larger the information set, the more likely it is that every move is best in enough histories to make that move as likely to occur as any other. Thus in games where players have little information, there may be a slight advantage to overconfidence.

[3]This argument assumes that the paranoid player examines the entire information set rather than a statistical sample as discussed in Section 3.4. If the paranoid player examines a statistical sample of the information set, there is a good chance that the statistical sample will not contain the history for which m is a loss. Hence in this case, statistical sampling would actually *improve* the paranoid player's play.

[4]We have verified this experimentally in several of the games in the following section, but omit these experiments due to lack of space.

Comparative performance. The above discussion suggests that (1) paranoid play should do better in games with "large" amounts of information, and (2) overconfident play might do better in games with "small" amounts of information. But will overconfident play do better than paranoid play? Suppose we choose a game with small amounts of information and play a paranoid player against an overconfident player: what should the outcome be? Overconfident play has the advantage of probably not diverging as drastically from the theoretically correct expected utility of a move, while paranoid play has the advantage of actually detecting and avoiding bad situations – situations to which the overconfident player will not give adequate weight.

Overall, it is not at all clear from our analysis how well a paranoid player and an overconfident player will do relative to each other in a real imperfect-information game. Instead, experimentation is needed.

5 Experiments

In this section we report on our experimental comparisons of overconfident versus paranoid play in several imperfect-information games.

One of the games we used was kriegspiel, an imperfect-information version of chess [Li 1994; Li 1995; Ciancarini, DallaLibera, and Maran 1997; Sakuta and Iida 2000; Parker, Nau, and Subrahmanian 2005; Russell and Wolfe 2005]. In kriegspiel, neither player can observe anything about the other player's moves, except in cases where the players directly interact with each other. For example, if a_1 captures one of a_2's pieces, a_2 now knows that a_1 has a piece where a_2's piece used to be. For more detail, see Section 5.2.

In addition, we created imperfect-information versions of three perfect-information games: P-games [Pearl 1984], N-games [Nau 1982a], and a simplified version of kalah [Murray 1952]. We did this by hiding some fraction $0 \leq h \leq 1$ of each player's moves from the other player. We will call h the *hidden factor*, because it is the fraction of information that we hide from each player: when $h = 0$, each player can see all of the other player's moves; when $h = 1$, neither player can see any of the other player's moves; when $h = 0.2$, each player can see 20% of the other player's moves; and so forth.

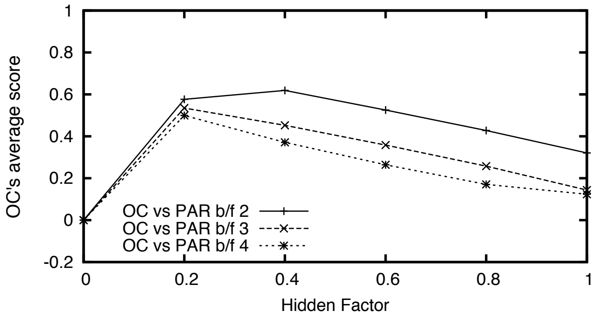
In each experiment, we played two players head-to-head for some number of trials, and averaged the results. Each player went first on half of the trials.

5.1 Experiments with Move-Hiding

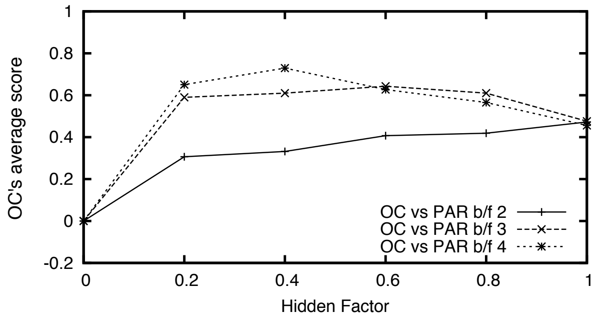
We did experiments in move-hiding variants of simple perfect information games. These experiments were run on 3.4 GHz Xeon processors with at least 2 GB of RAM per core. The programs were written in OCaml. All games were 10-ply long, and each player searched all the way to the end of the game.

Hidden-move P-game experiments. P-games were invented by Judea Pearl [Pearl 1981], and have been used in many studies of game-tree search (e.g., [Nau 1982a; Pearl 1984]). They are two-player zero-sum games in which the game tree has a constant branching factor b, fixed game length d, and fixed probability P_0 that the first player wins at any given leaf

(a) *Hidden-move P-games.* Each data point is an average of at least 72 trials.



(b) *Hidden-move N-games.* Each data point is an average of at least 39 trials.



(c) *Hidden-move kalah.* Each data point is an average of at least 125 randomly generated initial states.

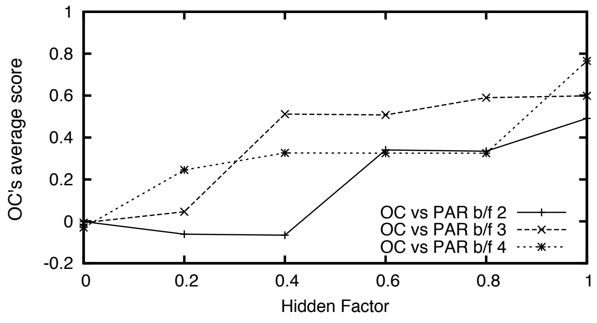


Figure 4. Average scores for overconfident (OC) play against paranoid (PAR) play.

node.[5] One creates a P-game by randomly assigning "win" and "loss" values to the b^d leaf nodes.

We did a set of experiments with P-games with $P_0 = 0.38$, which is the value of P_0 most likely to produce a nontrivial P-game [Nau 1982b]. We used depth $d = 10$, and varied the branching factor b. We varied the hidden factor h from 0 to 1 by increments of 0.2, so that the number of hidden moves varied from 0 to 10. In particular, we hid a player's m^{th} move if $\lfloor m \cdot h \rfloor > \lfloor (m-1) \cdot h \rfloor$. For instance, in a game where each player makes 5 moves and the hidden factor is 0.6, then the 2^{nd}, 4^{th}, and 5^{th} moves of both players are hidden.

For each combination of parameters, we played 2000 games: 1000 in which one of the

[5]Hence [Pearl 1984] calls P-games (d, b, P_0)-games.

players moved first, and 1000 in which the other player moved first. Thus in each of our figures, each data point is the average of 2000 runs.

Figure 4(a) shows the results of head-to-head play between the overconfident and paranoid strategies. These results show that in hidden-move P-games, paranoid play does indeed perform worse than overconfident play with hidden factors greater than 0. The results also confirm theorem 5, since overconfident play and paranoid play did equally well with hidden factor 0. From these experiments, it seems that paranoid play may not be as effective in imperfect-information games as it is in perfect information games.

Hidden-move N-game experiments. P-games are known to have a property called *game-tree pathology* that does not occur in "natural" games such as chess [Nau 1982a], and we wanted to ascertain whether this property might have influenced our experimental results on hidden-move P-games. N-games are similar to P-games but do not exhibit game-tree pathology, so we did a similar set of experiments on hidden-move N-games.

An N-game is specified by a triple (d, b, P_0), where d is the game length, b is the branching factor, and P_0 is a probability. An N-game specified by this triple has a game tree of height d and branching factor b, and each arc in the game tree is randomly assigned a value of $+1$ with probability P_0, or -1 otherwise. A leaf node is a win for player 1 (and a loss for player 2) if the sum of the values on the arcs between the root and the leaf node is greater than zero; otherwise the leaf node is a loss for player 1 (and a win for player 2).

Figure 4(b) shows our experimental results for hidden-move N-games. Just as before, overconfident and paranoid play did equally well with hidden factor 0, and overconfident play outperformed paranoid play with hidden factors greater than 0.

Kalah experiments. Kalah [Murray 1952] is also called mankalah, mancala, warri, and other names. It is an ancient African game played on a board with a number of pits that contain seeds, in which the objective is to acquire more seeds than the opponent, either by moving them to a special pit (called a kalah) or by capturing them from the opponent's pits.

In kalah, there are two rows of 6 pits. Flanking the rows of pits on both sides are the larger kalahs. Players sit on opposite sides of the board with one of the rows of pits nearer to each player. Each player owns the kalah on their left. The game starts with 6 stones in each of the pits except the kalahs. The player moves by picking up all the stones from one of the pits in the near row and placing one stone in each pit clockwise around the board including their kalah but excluding the opponent's kalah. If the last stone is placed in their kalah, the player moves again. If the last stone is placed in an empty pit, the player moves all stones from the opposite pit to their kalah. The game ends when the player to move has no moves because all pits on their side are empty. At that point, all stones in pits on the other player's side are placed in the player to move's kalah and the player with the most stones wins; ties occur when both plays own the same number of stones.

Because of the computation requirements of playing a full game of kalah, our experiments were on a simplified version of kalah that we call randomized kalah. The game differs from kalah in several ways:

- We vary the number of pits on the board. This varies the branching factor.

- To ensure a constant branching factor, we allow players to "move" from a pit that contains no stones. These are null moves that have no effect on the board.
- We end the game after 10 ply, to ensure that the algorithms can search the entire tree.
- We eliminate the move-again rule, to ensure alternating moves by the players.
- We start with a random number of stones in each pit to ensure that at each branching factor there will be games with non-trivial decisions.

Since randomized kalah is directly motivated by a very old game that people still play, its game trees are arguably much less "artificial" than those of P-games or N-games.

The results of playing overconfidence versus paranoia in hidden-move versions of randomized kalah are shown in Figure 4(c). The results are roughly similar to the P-game and N-game results, in the sense that overconfidence generally outperforms paranoia; but the results also differ from the P-game and N-game results in several ways. First, overconfidence generally does better at high hidden factors than at low ones. Second, paranoia does slightly better than overconfidence at hidden factor 0 (which does not conflict with Theorem 5, since kalah allows ties). Third, paranoia does better than overconfidence when the branching factor is 2 and the hidden factor is 0.2 or 0.4. These are the only results we saw where paranoia outperformed overconfidence.

The fact that with the same branching factor, overconfidence outperforms paranoia with hidden factor 0.6, supports the hypothesis that as the amount of information in the game decreases, paranoid play performs worse with respect to overconfident play. The rest of the results support that hypothesis as well: overconfidence generally increases in performance against paranoia as the hidden factor increases.

5.2 Kriegspiel Experiments

For experimental tests in an imperfect-information game people actually play, we used kriegspiel, an imperfect-information version of chess in which the players cannot see their opponent's pieces. Kriegspiel is useful for this study because (i) it is clearly a game where each player has only a small amount of information about the current state, and (ii) due to its relationship to chess, it is complicated enough strategically to allow for all sorts of subtle and interesting play. A further advantage to kriegspiel is that it is played competitively by humans even today [Li 1994; Li 1995; Ciancarini, DallaLibera, and Maran 1997].

Kriegspiel is a chess variant played with a chess board. When played in person, it requires three chess kits: one for each player and one for the referee. All boards are set up as in normal chess, but neither player is allowed to see their opponent's or the referee's board. The players then move in alternation as in standard chess, keeping their moves hidden from the other player. All player's moves are also played by the referee on the referee's board. Since neither player can see the referee's board, the referee acts as a mediator, telling the players if the move they made is legal or illegal, and giving them various other observations about the move made. We use the ICC's kriegspiel observations, described at

Table 1. Average scores for overconfident play against paranoid play, in 500 kriegspiel games using the ICC ruleset. d is the search depth.

Over-	Paranoid		
confident	$d = 1$	$d = 2$	$d = 3$
$d = 1$	+0.084	+0.186	+0.19
$d = 2$	+0.140	+0.120	+0.156
$d = 3$	+0.170	+0.278	+0.154

Table 2. Average scores for overconfident and paranoid play against HS, with 95% confidence intervals. d is the search depth.

d	Paranoid	Overconfident
1	–0.066 ± 0.02	+0.194 ± 0.038
2	+0.032 ± 0.035	+0.122 ± 0.04
3	+0.024 ± 0.038	+0.012 ± 0.042

`http://www.chessclub.com/help/Kriegspiel`. Observations define the information sets. Any two histories that have the same observations at each move and all the same moves for one of the players are in the same information set.

When played on the internet, the referee's job can be automated by a computer program. For instance, on the Internet Chess Club one can play kriegspiel, and there have been thousands of kriegspiel games played on that server.

We ran our experiments on a cluster of computers runing linux, with between 900 MB and 1.5 GB RAM available to each process. The processors were Xeons, Athlons, and Pentiums, ranging in clockspeed from 2 GHz to 3.2 GHz. We used time controls and always forced players in the same game to ensure the results were not biased by different hardware. The algorithms were written in C++. The code used for overconfident and paranoid play is the same, with the exception of the opponent model. We used a static evaluation function that was developed to reward conservative kriegspiel play, as our experience suggests such play is generally better. It uses position, material, protection and threats as features.

The algorithms used for kriegspiel are depth-limited versions of the paranoid and overconfident players. To handle the immense information-set sizes in kriegspiel, we used iterative statistical sampling (see Section 3.4). To get a good sample with time control requires limiting the search depth to at most three ply. Because time controls remain constant, the lower search depths are able to sample many more histories than the higher search depths.

Head-to-head overconfident vs. paranoid play. We did experiments comparing overconfident play to paranoid play by playing the two against each other. We gave the algorithms 30 seconds per move and played each of depths one, two, and three searches against each other. The results are in Table 1. In these results, we notice that overconfident play consistently beats paranoid play, regardless of the depth of either search. This is consistent with our earlier results for hidden-move games (Section 5.1); and, in addition, it shows overconfident play doing better than paranoid play in a game that people actually play.

HS versus overconfidence and paranoia. We also compared overconfident and paranoid play to the hybrid sampling (HS) algorithm from our previous work [Parker, Nau, and Subrahmanian 2005]. Table 2 presents the results of the experiments, which show overconfidence playing better than paranoia except in depth three search, where the results are inconclusive. The inconclusive results at depth three (which are an average over 500 games) may be due to the sample sizes achieved via iterative sampling. We measured an average of 67 histories in each sample at depth three, which might be compared to an average of 321 histories in each sample at depth two and an average of 1683 histories at depth one. Since both algorithms use iterative sampling, it could be that at depth three, both algorithms examine insufficient samples to do much better than play randomly.

In every case, overconfidence does better than paranoia against HS. Further, overconfidence outperforms HS in every case (though sometimes without statistical significance), suggesting that information-set search is an improvement over the techniques used in HS.

6 Related Work

There are several imperfect-information game-playing algorithms that work by treating an imperfect-information game as if it were a collection of perfect-information games [Smith, Nau, and Throop 1998; Ginsberg 1999; Parker, Nau, and Subrahmanian 2005]. This approach is useful in imperfect-information games such as bridge, where it is not the players' moves that are hidden, but instead some information about the initial state of the game. The basic idea is to choose at random a collection of states from the current information set, do conventional minimax searches on those states as if they were the real state, then aggregate the minimax values returned by those searches to get an approximation of the utility of the current information set. This approach has some basic theoretical flaws [Frank and Basin 1998; Frank and Basin 2001], but has worked well in games such as bridge.

Poker-playing computer programs can be divided into two major classes. The first are programs which attempt to approximate a Nash equilibrium. The best examples of these are PsOpti [Billings, Burch, Davidson, Holte, Schaeffer, Schauenberg, and Szafron 2003] and GS1 [Gilpin and Sandholm 2006b]. The algorithms use an intuitive approximation technique to create a simplified version of the poker game that is small enough to make it feasible to find a Nash equilibrium. The equilibrium can then be translated back into the original game, to get an approximate Nash equilibrium for that game. These algorithms have had much success but differ from the approach in this paper: unlike any attempt to find a Nash equilibrium, information-set search simply tries to find the optimal strategy against a given opponent model. The second class of poker-playing programs includes Poki [Billings, Davidson, Schaeffer, and Szafron 2002] which uses expected value approximations and opponent modeling to estimate the value of a given move and Vexbot [Billings, Davidson, Schauenberg, Burch, Bowling, Holte, Schaeffer, and Szafron 2004] which uses search and adaptive opponent modeling.

The above works have focused specifically on creating successful programs for card games (bridge and poker) in which the opponents' moves (card plays, bets) are observable. In these games, the hidden information is which cards went to which players when the cards were dealt. Consequently, the search techniques are less general than information-set search, and are not directly applicable to hidden-move games such as kriegspiel and the other games we have considered in this paper.

7 Conclusion

We have introduced a recursive formulation of the expected value of an information set in an imperfect information game. We have provided analytical results showing that this expected utility formulation plays optimally against any opponent if we have an accurate model of the opponent's strategy.

Since it is generally not the case that the opponent's strategy is known, the question then arises as to what the recursive search should assume about an opponent. We have studied two opponent models, a "paranoid" model that assumes the opponent will choose the moves that are best for them, hence worst for us; and an "overconfident" model that assumes the opponent is making moves purely at random.

We have compared the overconfident and paranoid models in kriegspiel, in an imperfect-information version of kalah, and in imperfect-information versions of P-games [Pearl 1984] and N-games [Nau 1982a]. In each of these games, the overconfident strategy consistently outperformed the paranoid strategy. The overconfident strategy even outperformed the best of the kriegspiel algorithms in [Parker, Nau, and Subrahmanian 2005].

These results suggest that the usual assumption in perfect-information game tree search—that the opponent will choose the best move possible—is not as effective in imperfect-information games.

Acknowledgments: This work was supported in part by AFOSR grant FA95500610405, NAVAIR contract N6133906C0149, DARPA IPTO grant FA8650-06-C-7606, and NSF grant IIS0412812. The opinions in this paper are those of the authors and do not necessarily reflect the opinions of the funders.

References

Applegate, D., G. Jacobson, and D. Sleator (1991). Computer analysis of sprouts. Technical report, Carnegie Mellon University.

Billings, D., N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, and D. Szafron (2003). Approximating game-theoretic optimal strategies for full-scale poker. In *IJCAI*, pp. 661–668.

Billings, D., A. Davidson, J. Schaeffer, and D. Szafron (2002). The challenge of poker. *Artif. Intell. 134*, 201–240.

Billings, D., A. Davidson, T. Schauenberg, N. Burch, M. Bowling, R. Holte, J. Schaeffer, and D. Szafron (2004). Game tree search with adaptation in stochastic imperfect information games. *Computers and Games 1*, 21–34.

Ciancarini, P., F. DallaLibera, and F. Maran (1997). Decision Making under Uncertainty: A Rational Approach to Kriegspiel. In J. van den Herik and J. Uiterwijk (Eds.), *Advances in Computer Chess 8*, pp. 277–298.

Corlett, R. A. and S. J. Todd (1985). A monte-carlo approach to uncertain inference. In *AISB-85*, pp. 28–34.

Frank, I. and D. Basin (2001). A theoretical and empirical investigation of search in imperfect information games. *Theoretical Comp. Sci. 252*, 217–256.

Frank, I. and D. A. Basin (1998). Search in games with incomplete information: A case study using bridge card play. *Artif. Intell. 100*(1-2), 87–123.

Gilpin, A. and T. Sandholm (2006a). Finding equilibria in large sequential games of imperfect information. In *EC '06*, pp. 160–169.

Gilpin, A. and T. Sandholm (2006b). A texas hold'em poker player based on automated abstraction and real-time equilibrium computation. In *AAMAS '06*, pp. 1453–1454.

Ginsberg, M. L. (1999). GIB: Steps toward an expert-level bridge-playing program. In *IJCAI-99*, pp. 584–589.

Li, D. (1994). *Kriegspiel: Chess Under Uncertainty*. Premier.

Li, D. (1995). *Chess Detective: Kriegspiel Strategies, Endgames and Problems*. Premier.

Murray, H. J. R. (1952). *A History of Board Games other than Chess*. London, UK: Oxford at the Clarendon Press.

Nau, D. S. (1982a). An investigation of the causes of pathology in games. *Artif. Intell. 19*(3), 257–278.

Nau, D. S. (1982b). The last player theorem. *Artif. Intell. 18*(1), 53–65.

Nau, D. S. (1983). Decision quality as a function of search depth on game trees. *JACM 30*(4), 687–708.

Osborne, M. J. and A. Rubinstein (1994). *A Course In Game Theory*. MIT Press.

Parker, A., D. Nau, and V. Subrahmanian (2005, August). Game-tree search with combinatorially large belief states. In *IJCAI*, pp. 254–259.

Pearl, J. (1981, August). Heuristic search theory: Survey of recent results. In *Proc. Seventh Internat. Joint Conf. Artif. Intel.*, Vancouver, Canada, pp. 554–562.

Pearl, J. (1984). *Heuristics: intelligent search strategies for computer problem solving*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Reif, J. (1984). The complexity of two-player games of incomplete information. *Jour. Computer and Systems Sciences 29*, 274–301.

Russell, S. and J. Wolfe (2005, August). Efficient belief-state and-or search, with application to kriegspiel. In *IJCAI*, pp. 278–285.

Sakuta, M. and H. Iida (2000). Solving kriegspiel-like problems: Exploiting a transposition table. *ICCA Journal 23*(4), 218–229.

Smith, S. J. J., D. S. Nau, and T. Throop (1998). Computer bridge: A big win for AI planning. *AI Magazine 19*(2), 93–105.

von Neumann, J. and O. Morgenstern (1944). *Theory of Games and Economic Behavior*. Princeton University Press.