On the Behavior of UCT in Synthetic Search Spaces

Raghuram Ramanujan
Dept. of Computer Science
Cornell University
Ithaca, NY 14853, USA
raghu@cs.cornell.edu

Ashish Sabharwal
IBM Watson Research Center
Yorktown Heights, NY, 10598, USA
ashish.sabharwal@us.ibm.com

Bart Selman
Dept. of Computer Science
Cornell University
Ithaca, NY 14853, USA
selman@cs.cornell.edu

Abstract

UCT and Minimax are two of the most prominent tree-search based adversarial reasoning strategies for a variety of challenging domains, such as Chess and Go. Their complementary strengths in different domains have been the motivation for several works attempting to achieve a better understanding of their vastly different behavior. Rather than using complex games as a testbed for deriving indirect insights into UCT and Minimax, we propose a relatively simple model on which sampling-based techniques, in particular UCT, significantly outperform Minimax. The simplicity of the model enables novel analytical computations of Minimax's decision accuracy, while an extension of this model incorporating search traps and correlated heuristic noise confirms that UCT's performance begins to deteriorate as more and more traps are added.

Introduction

Minimax search with alpha-beta pruning has endured for many decades as the algorithm of choice for tree search in complete information games. It has produced game-playing programs that surpass human-level play in the games of Chess and Checkers (Campbell, Hoane, and Hsu 2002; Schaeffer et al. 1992). The game of Go, however, remains something of a final frontier. Due to its large branching factor and the lack of good board evaluation functions, Minimax-style full-width search approaches have been mostly ineffective in this domain. The emergence of MoGo (Gelly and Silver 2007) marked the first significant improvement in the strength of Go-playing programs in many years. Underpinning its success is Upper Confidence bounds for Trees (UCT), a novel sampling-based tree search algorithm (for a detailed description of the algorithm, see Kocsis and Szepesvári 2006), that has since been successfully applied to many other challenging domains (Balla and Fern 2009; Finnsson and Björnsson 2008). Intriguingly, however, efforts to replicate this success in domains where Minimax is the "gold-standard" (such as Chess) have been mostly futile. Thus, a big open question remains — is it possible to characterize aspects of search domains that make them more or less favorable to UCT- style search algorithms?

Prior attempts to answer this question have focused on specific domains such as Chess (Ramanujan, Sabharwal, and Selman 2010b), or Mancala (Ramanujan and Selman 2011). While those studies have been useful in building an intuitive understanding of the strengths and weaknesses of UCT, the domains themselves are too complicated for any mathematical analysis. In this paper, we address this issue by studying UCT, Minimax, and other simple alternatives such as leaf averaging, on a family of synthetic tree models.

Our most basic model, to our knowledge, is the first simple family of trees on which a UCT-style sampling based search strategy clearly outperforms Minimax (with alpha-beta pruning). Briefly, this model produces bounded depth trees where there is precisely one optimal action at each node. A player picking a sub-optimal action suffers a fixed additive cost of k. The tree is constructed in a top-down fashion and the true minimax value for any node is known by design. The heuristic evaluation is based on an *independent random noise* model. Specifically, the heuristic value of a node is the sum of its true Minimax value and some Gaussian noise.

The simplicity of this model enables us to provide *analytical* equations capturing the distribution of the backed-up heuristic estimate for any node. We use these to compute the decision accuracy of Minimax in this model, i.e., how often Minimax selects the optimal child of the root node of the search tree. While a closed form solution for decision accuracy is hard to obtain, we use numerical computation with Matlab to obtain decision accuracy figures. We also discuss a simpler analytical approximation that relies on the fact that the minimum and maximum of two Gaussian random variables can often be approximated well by another Gaussian variable, with easily computable mean and variance.

As a comparison, we estimate the decision accuracy of UCT on this tree model through simulations. UCT is modified to use the same heuristic model as Minimax to estimate the value of leaf nodes, in place of traditional random playouts. We find that UCT clearly outperforms Minimax in our basic tree model and the gap in performance grows with deeper search trees. What makes UCT work well in this model is the fact that randomly sampling nodes of the tree yields a fairly reliable estimate of the true Minimax value of the tree, despite the noise in the heuristic. Good guidance from sampling is one of the keys to UCT's success, and this aspect is clearly demonstrated in this simple model.

While the independent noise assumption allows for ease

of analysis, it is not fully realistic — in real games, there are often correlations between the heuristic evaluations of closely related positions. In particular, a heuristic that erroneously evaluates a certain position is more likely to misclassify similar positions as well. Thus, we study a second model where the independent noise assumption is relaxed. In trees grown using this model, we show that UCT's performance is greatly affected by the prevalence of shallow *search traps*, i.e., nodes with sub-optimal moves that, if taken, lead to an easy win for the opponent in just a few more moves. In particular, UCT outperforms Minimax when very few traps are present in the search tree, but this trend is reversed when more traps are added. This is in line with observations from real world games.

We begin the rest of the paper by describing the general search tree model to be used in our studies. We first describe the simplest form of the model with fixed k and fixed noise variance σ. We present results on the decision accuracy of various search strategies as well as a derivation of analytical expressions that can be evaluated numerically in order to compute the decision accuracy of Minimax, exactly or approximately. We then present and discuss the correlated noise model and the impact of search traps on UCT, and conclude with a brief summary.

Game Tree Model

Numerous synthetic game tree models have been proposed in the literature, particularly to study the phenomenon of look-ahead pathology in Minimax search. The existing approaches to tree construction may be broadly classified into two types — bottom-up and top-down. In the bottom-up approach, the values of the leaves of the tree are carefully specified, either as win/loss values (Nau 1982; Pearl 1983) or as real numbers (Luštrek, Gams, and Bratko 2005). The value of the game (as well as the minimax values of the internal nodes) is computed using a full-depth alpha-beta search, which in practice limits the size of the trees that may be studied. Moreover, the value of each leaf is independent of the value of other leaves. This does not accurately model real games where values of sibling nodes tend to be correlated. In the top-down approach, values are assigned to edges in the game tree. The value of a leaf is then defined by the sum of the edge values on the path from the root node to the leaf (Nau 1982; Scheucher and Kaindl 1998). These "incremental" tree models do introduce correlations among sibling nodes; however, they still suffer from the problem that computing the true minimax value of any internal node in the tree involves performing a search. More recently, Furtak and Buro introduced the *prefix value* game tree model which builds upon the top-down formulation of Scheucher and Kaindl. Notably, no search is required to determine the true value of internal nodes in this tree model which allows simulation and analysis of different search procedures on arbitrarily large games (Furtak and Buro 2009). We describe the details of this model, and our extensions, below.

Without loss of generality, we restrict our study to trees where the maximizing player (henceforth, Max) is on move at the root. The minimax values of nodes are drawn from the set of integers, with positive values representing wins for Max and the rest indicating wins for Min, the minimizing player. For the sake of simplicity, we disallow draws.

Let $m(v)$ represent the true minimax value of a node v. The key insight exploited by the model is the fact that making a move can never *increase* the value of a position for the player on move. Moreover, at least one child of v must have the same value as v. Thus, given a node v, we grow the sub-tree rooted at v as follows:

- Denote the set of children of v by $V = \{v_1, v_2, \ldots, v_b\}$, corresponding to action choices $A = \{a_1, a_2, \ldots, a_b\}$.
- Pick an a_i uniformly at random from A — this is designated as the optimal action at v
- Assign $m(v_i) = m(v)$. If Max is on move at v, then for all $j \neq i$, $m(v_j) = m(v) - k$. If Min is on move at v, then for all $j \neq i$, $m(v_j) = m(v) + k$.

Here, k is a constant that quantifies the cost incurred by the player on move for taking a sub-optimal action. The process is seeded by setting the minimax value of the root node to $+1$. This is necessary to ensure that Max is faced with an interesting decision at the root node. A parameter d_{max} controls the depth of the tree.

The tree model as described grows complete trees to the specified depth. However, many real games do not exhibit such regularity and contain a proliferation of early terminal nodes or "trap states". Recent work has shown that the prevalence of trap states has an impact on the performance of sampling-style algorithms such as UCT (Ramanujan, Sabharwal, and Selman 2010a; Ramanujan and Selman 2011). To better mimic such search spaces, we extend the basic tree model by introducing a new parameter δ that is a threshold on the maximum magnitude the minimax value of a node can assume. If $|m(v)| > \delta$ for a node v, it is marked as a terminal node, with the sign of $m(v)$ determining the outcome of the game. The value of δ thus has an inverse relationship to the density of trap states in the game tree — figure 1 illustrates this phenomenon concretely.

Finally, in most interesting real-world games, a complete search to the horizon of the game is infeasible. The typical work-around is to set a depth (or time) cut-off and to evaluate the leaf nodes at the frontier of the search using a heuristic function. In our game tree model, we generate heuristic values for nodes by adding Gaussian noise to their true minimax values. We introduce and discuss two different approaches to generating this noise in the following sections.

Independent Noise Model

In this model, the heuristic value $h(v)$ of a node v is defined as follows:

$$h(v) = m(v) + X$$

Here, $X \sim \mathcal{N}(0, \sigma)$, i.e., X is a Gaussian random variable with mean 0 and standard deviation σ, which is a parameter of the model. h is re-scaled to fall in the interval $(0, 1)$, with the values 0 and 1 used to denote terminal loss and win

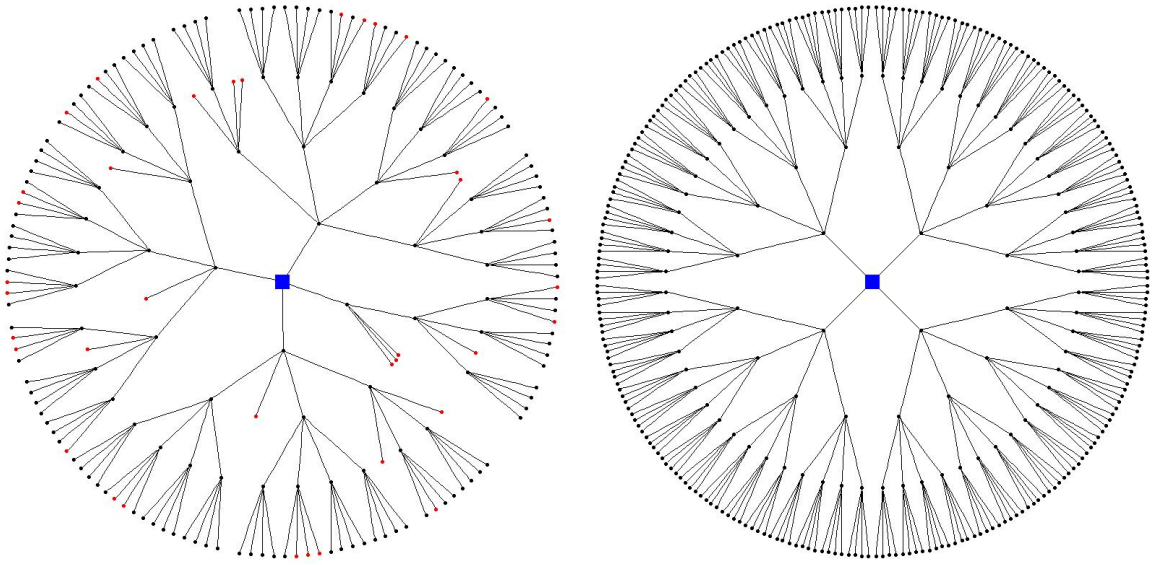

Figure 1: Game trees with many search traps (left, $\delta = 5$) and no traps (right, $\delta = 25$). The value of k is chosen uniformly at random from the set $\{1, \ldots, 8\}$ for each sub-optimal action.

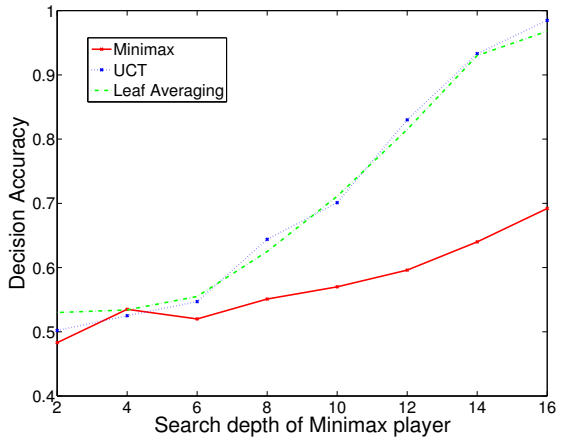

Figure 2: Decision accuracy of Minimax, UCT and Leaf Averaging search versus search effort, with $\sigma = 30$, $k = 1$, $\delta = d_{max} = \infty$ and branching factor $= 2$

positions, respectively, for Max. The noise sampled to compute the heuristic value of a node is independent of the noise sampled at other nodes.

Figure 2 illustrates the variation in the decision accuracy of three search procedures — Minimax, UCT and Leaf Averaging — as the search effort (quantified by the search depth of the Minimax player) is varied with the other parameters fixed to their indicated values. The leaf averaging approach is a naïve decision procedure that works as follows: for each child d of a given node, the *average* utility of the leaf nodes in the sub-tree rooted at d is computed. This average value is treated as a heuristic estimate of the utility of the child d — the best child is then selected based on this estimate. For UCT, we use an exploration bias parameter of 0.2, which was empirically found to produce the best performance on this family of trees. To make a fair comparison, we normalize for search effort by allowing UCT and leaf averaging to expand at most the number of nodes examined by Minimax. The average decision accuracy is computed over 5000 trees per data point.

We make two observations. Firstly, our tree model is not pathological (Beal 1980; Nau 1982) as evidenced by the fact that searching deeper leads to a consistently higher decision accuracy for Minimax. Further, UCT (and leaf averaging) significantly outperform Minimax with increasing search effort. This is significant — to our knowledge, this is the first realistic synthetic tree model where UCT demonstrably outperforms Minimax search. The surprisingly good performance of leaf averaging offers a clue to UCT's impressive performance; namely, sampling in conjunction with averaging offers very reliable estimates of the value of states in this tree model. We prove this statement more rigorously.

Let T be a tree of depth d generated by the model. For a non-leaf node v in T, one may attempt to estimate its true minimax value $m(v)$ by simply sampling the estimated values of several leaf nodes in the subtree rooted at v and taking the average of these values. In general, this average may not converge to $m(v)$, but for the basic model with fixed k, fixed σ, and independent noise discussed above, it actually does converge to $m(v)$ as long as there are an even number of levels below v till the leaves. To see this, consider Y_v, the average of the estimated values of *all* leaf nodes in the subtree T_v consisting of v and its descendents in T; the depth d_v of T_v is even. Note that Y_v is a random variable whose value depends on the Gaussian noise injected by the heuristic when estimating $m(v)$ for leaves v.

Proposition 1. *The expected value of Y_v is $m(v)$.*

Proof. Assume without loss of generality that the player on move at v is Max. We prove that $\mathbb{E}[Y_v] = m(v)$ by induction on d_v.

For the base case, when d_v is 0, $Y_v = m(v) + X$ where X is a random variable drawn from the distribution $\mathcal{N}(0, \sigma)$. It follows that $\mathbb{E}[Y_v] = m(v) + \mathbb{E}[N] = m(v)$.

For the inductive step, let $v_{i,j}, 1 \leq i, j \leq b$, be the b^2 descendents of v two levels below it in T_v, with $v_{i,1}, 1 \leq i \leq b$, as the optimal moves at that level. Here b denotes the branching factor. Let U denote the set of these b^2 nodes. For any $u \in U$, let Y_u be a random variable whose value is the average of the estimated values of all leaf nodes under u. Then $Y_v = \sum_{u \in U} Y_u$ and therefore $\mathbb{E}[Y_v] = \sum_{u \in U} \mathbb{E}[Y_u]$. By induction hypothesis, $\mathbb{E}[Y_u] = m(u)$. Hence, we have $\mathbb{E}[Y_v] = \sum_{u \in U} m(u)$. We will argue that this latter sum equals $m(v)$.

To see this, note that by construction of T_v, we know that $m(u) = m(v) + k$ for $u \in \{v_{1,2}, v_{1,3}, \ldots, v_{1,b}\}$, that $m(u) = m(v) - k$ for $u \in \{v_{2,1}, v_{3,1}, \ldots, v_{b,1}\}$, and $m(u) = m(v)$ for all other nodes u in U. In other words, there are as many $m(v) - k$ values two levels below v as there are $m(v) + k$ values, and the remaining values equal $m(v)$. Hence the summation $\sum_{u \in U} m(u)$ equals $m(v)$, as desired. □

Analytical Computation of Decision Accuracy

A key benefit of this model is that it allows us to analytically compute the decision accuracy of Minimax, without relying on simulations based on samples from the Gaussian noise distribution. We discuss this approach next.

Exact Method Consider a tree T. For simplicity, let us assume that the branching factor of T is 2. As before, for a node v, $m(v)$ denotes the true minimax value of v. The heuristic minimax value of v, computed by taking the min or the max, as appropriate, of the heuristic minimax values of the two children v_L and v_R of v, is a random variable X_v whose value is ultimately a function of the true values of the leaves of T in the subtree under v, as well as the Gaussian noise at these leaves. Consider the cumulative probability distribution function (CDF) of X_v, defined as $F_v(x) = \Pr[X_v \leq x]$. Then we can write F_v for any internal node v recursively as follows:

$$F_v(x) = \begin{cases} F_{v_L}(x) F_{v_R}(x) & \text{if } v \text{ is Max} \\ 1 - (1 - F_{v_L}(x))(1 - F_{v_R}(x)) & \text{if } v \text{ is Min} \end{cases}$$

Here, by "v is Max" we mean the player on move at v is Max; similarly for Min. When v is a leaf node of T with true value x^*, then F_v is simply the CDF of the Gaussian distribution centered around x^* and with the specified standard deviation σ, i.e., $F_v(x) = \Phi(\frac{x - x^*}{\sigma})$.

Now, if u is the root node (w.l.o.g., a Max node) of T with children u_L and u_R, then we can compute the decision accuracy of Minimax at u by evaluating the following integral:

$$\begin{aligned} \text{decAcc}_{\text{MM}}(u) &= \Pr[X_{u_L} \geq X_{u_R}] \\ &= \int \Pr[X_{u_L} = x] \Pr[X_{u_R} \leq x] \, dx \\ &= \int f_{u_L}(x) F_{u_R}(x) \, dx \end{aligned}$$

where $f_{u_L}(x)$ denotes the probability density function of X_{u_L} at value x and equals the derivative of F_{u_L} evaluated at value x.

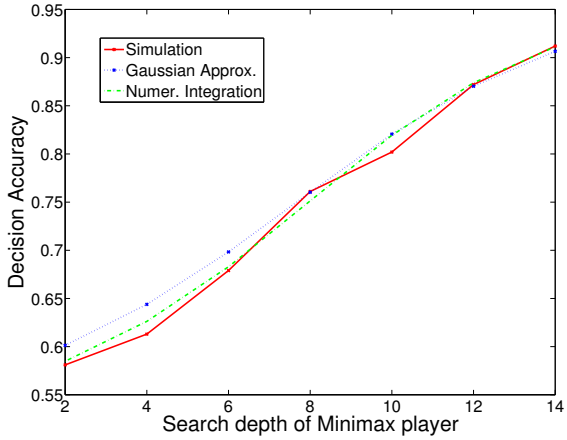

Figure 3: Decision accuracy of Minimax search as predicted by our Gaussian approximation and numerical integration methods, compared to actual simulation results

The green curve in figure 3 depicts the results of a numerical evaluation of the above integral in order to compute the decision accuracy of Minimax as a function of search depth. An important aspect that makes this numerical integration feasible is the fact that we can define the CDF of a node at any point x recursively as a combination of other CDFs at the same point x. The numerical evaluation is performed using Matlab[1], with the "quadl" integration method, a tolerance level of 10^{-6}, and the integral evaluated in the large but bounded range $[-1000, 1000]$. The σ value used in the tree model is 4. As can be seen, this numerical computation, based on the analytical integral discussed above, provides a very good estimate of the decision accuracy of Minimax, and is much smoother than the simulation based curve shown in red.

Gaussian Approximation An alternative way to estimate the decision accuracy analytically is to use the fact that the min and max of two Gaussians are often well approximated by yet another Gaussian whose mean and variance can be easily computed. This approximation is especially accurate when the two constituent Gaussians have the same variance – which is indeed the case in the current tree model.

As above, let $f_v(x)$ denote the probability density function (PDF) of X_v evaluated at value x. If v is a leaf of the

[1] `http://www.mathworks.com/products/matlab`

tree T, then, by construction, f_v is simply $\mathcal{N}(m(v),\sigma)$. If v is an internal node with children v_L and v_R, let us assume we have recursively computed the Gaussian approximations $\mathcal{N}(\mu_{v_L},\sigma_{v_L})$ and $\mathcal{N}(\mu_{v_R},\sigma_{v_R})$ of their true probability density functions, respectively. Then, following the moment-matching approach outlined by Nadarajah and Kotz, we can approximate the probability density function for v (assuming v is a Max node) as $\mathcal{N}(\mu_v,\sigma_v)$ (Nadarajah and Kotz 2008), where:

$$\rho = \mu_{v_L} - \mu_{v_R}$$

$$\theta = \sqrt{\sigma_{v_L}^2 + \sigma_{v_R}^2}$$

$$\mu_v = \mu_{v_L}\Phi(\frac{\rho}{\theta}) + \mu_{v_R}\Phi(-\frac{\rho}{\theta}) + \theta\phi(\frac{\rho}{\theta})$$

$$\sigma_v^2 = (\sigma_{v_L}^2 + \mu_{v_L}^2)\Phi(\frac{\rho}{\theta}) + (\sigma_{v_R}^2 + \mu_{v_R}^2)\Phi(-\frac{\rho}{\theta}) + (\mu_{v_L} + \mu_{v_R})\theta\phi(\frac{\rho}{\theta}) - \mu_v^2$$

Here, $\phi(\cdot)$ and $\Phi(\cdot)$ are the PDF and CDF respectively of the standard normal distribution. We refer the reader to Nadarajah and Kotz 2008 for similar analytical expressions for μ_v and σ_v when v is a Min node. Given root node u with children u_L and u_R, the decision accuracy of Minimax equals $\Pr[X_{u_L} \geq X_{u_R}] = \Pr[X_{u_L} - X_{u_R} \geq 0]$. With Gaussian approximations for X_{u_L} and X_{u_R} in hand, we can in turn approximate $X_{u_L} - X_{u_R}$ by a Gaussian $\mathcal{N}(\mu_{u_L} - \mu_{u_R}, \sqrt{\sigma_{u_L}^2 + \sigma_{u_R}^2})$. The decision accuracy may be computed directly using this Gaussian PDF.

The blue curve in Figure 3 shows the resulting estimates of decision accuracy. As can be seen, the Gaussian approximation is somewhat over-optimistic for small values of d (specifically, for $d \leq 10$ in this case) but then begins to coincide with the numerical estimate of the exact analytical expression discussed previously.

Enhanced Independent Noise Model

While the basic tree model as presented is easy to analyze, figure 2 shows that even a very naïve sampling-based strategy does quite well on those trees. We make the following two observations in the basic model:

1. The noise in the heuristic estimate of a node is constant *regardless* of the depth of the node. In reality, heuristic estimates tend to improve as one encounters nodes closer to terminal states.
2. The penalty term assigned to sub-optimal moves (k) is a constant. However, in real games, sub-optimal moves are not all equally bad — some are worse than others.

Based on these insights, we make two slight modifications to our basic model. In particular, rather than use a constant σ for the standard deviation of the Gaussian noise, we define it as a function:

$$\sigma(v) = \min\{(\delta - |m(v)|), (d_{max} - d(v))\} \quad (1)$$

where $d(v)$ represents the depth of the node v from the root. With this formulation, the magnitude of the noise becomes height-dependent — the closer a node is to a terminal state, the more accurate the heuristic value. Moreover, we also modify k to be a random variable drawn uniformly at random from the set $\{1,\ldots,k_{max}\}$. For this modified tree model, we re-plot the decision accuracies of Minimax, UCT and leaf averaging while varying the search effort (figure 4). As can be seen in the plot, UCT is still the best performing search method, with Minimax the worst of the three. Moreover, in this model, UCT demonstrates a clear edge over the simple sampling algorithm, demonstrating that some careful balancing of exploration and exploitation *does* boost the search performance over naïve sampling.

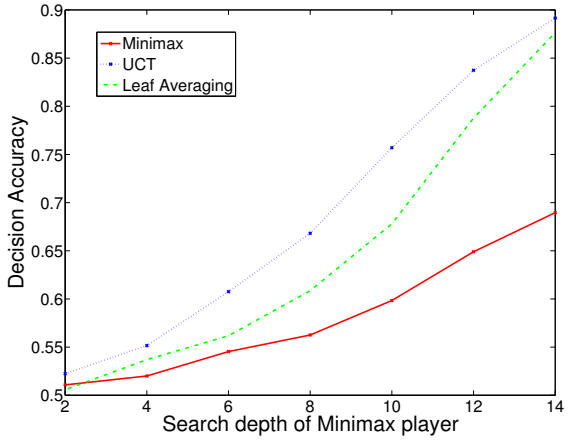

Figure 4: Decision accuracy of Minimax, UCT and Tree Averaging search versus search effort, with height-dependent σ and k as a uniform random variable ($k_{max} = 8$). All other parameters are as in figure 2.

Correlated Noise Model

In this section, we relax the independent noise assumption (in the extended model) and examine the relative performance of UCT and Minimax on the resulting game trees. Since heuristic evaluations of similar positions often tend to be correlated, this approach models real games more accurately. In this model, the noise used to generate the heuristic value of a node v is no longer independent of the noise used at other nodes. In particular, the noise X no longer has a mean of 0; rather, $X \sim \mathcal{N}(\mu,\sigma(v))$, where σ is as defined in equation 1. μ is a value drawn from $\{-s,0,+s\}$, depending on the heuristic value assigned to p, the parent of v (in our experiments, s is fixed at 5). If p is a winning position for Max, which is misclassified as a loss by the heuristic ("false loss"), then $\mu = -s$. This increases the likelihood that $h(v)$ will be negative as well. Symmetrically, when p is a false win, $\mu = +s$; otherwise $\mu = 0$. As before, $h(v)$ is rescaled to the interval $(0,1)$.Under this scheme, heuristic evaluation errors tend to be reinforced down the tree. In other words, if the heuristic makes a mistake in evaluating a particular state

p, it is more likely to make mistakes when evaluating the children of p as well.

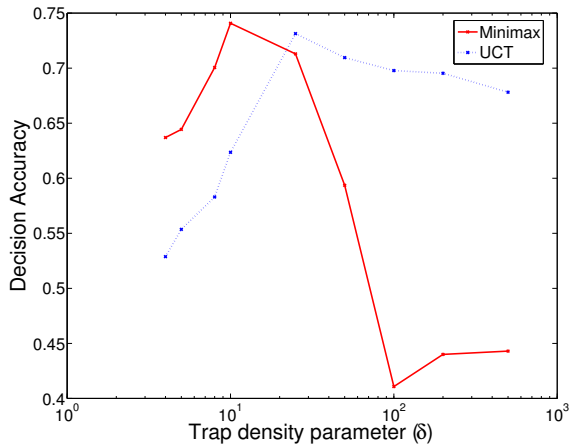

Figure 5: Decision accuracy of Minimax and UCT on trees using the correlated noise model, as the trap density parameter δ is varied. Higher values of δ correspond to fewer traps.

In this model, UCT displays sensitivity to trap states as illustrated in figure 5. UCT exhibits superior decision accuracy in state spaces with few search traps (i.e., large values of δ) compared to Minimax. As the density of search traps is increased (δ lowered), Minimax overtakes UCT as the better search procedure. This confirms similar observations from real games such as Chess and Mancala (Ramanujan, Sabharwal, and Selman 2010a; 2010b; Ramanujan and Selman 2011). Moreover, it offers an explanation for the success of UCT in the game of Go. Namely, the scarcity of trap states in the early stages of the game allows UCT to build up a positional advantage that it can then exploit in the end-game phase.

Conclusions

In this work, we proposed relatively simple synthetic game tree search models that bring out key aspects of dominant adversarial search strategies. These synthetic models serve as a vehicle to better understand the strengths of UCT, Minimax, and simpler alternatives in a controlled setting, complementing previous work done in the context of complex games such as Chess, Go, and Mancala. The simplicity of the basic model allows the use of analytical and numerical methods to compute the true backed-up value of the Minimax tree in the presence of independent heuristic noise. For the more realistic model with correlated noise, UCT exhibits a characteristic which mirrors it's behavior in many real games, namely, the strong dependence of it's performance on the presence of search traps.

References

Balla, R.-K., and Fern, A. 2009. UCT for tactical assault planning in real-time strategy games. In *21st IJCAI*, 40–45.

Beal, D. 1980. An analysis of minimax. In M.R.B., C., ed., *Advances in Computer Chess 2*, 103–109. Edinburgh University Press.

Campbell, M.; Hoane, Jr., A. J.; and Hsu, F.-h. 2002. Deep Blue. *Artif. Intell.* 134:57–83.

Finnsson, H., and Björnsson, Y. 2008. Simulation-based approach to general game playing. In *AAAI-08*, 259–264.

Furtak, T., and Buro, M. 2009. Minimum proof graphs and fastest-cut-first search heuristics. In *21st IJCAI*, 492–498.

Gelly, S., and Silver, D. 2007. Combining online and offline knowledge in UCT. In *24th ICML*, 273–280.

Kocsis, L., and Szepesvári, C. 2006. Bandit based Monte-Carlo planning. In *17th ECML*, volume 4212 of *LNCS*, 282–293.

Luštrek, M.; Gams, M.; and Bratko, I. 2005. Why minimax works: an alternative explanation. In *19th IJCAI*, 212–217.

Nadarajah, S., and Kotz, S. 2008. Exact distribution of the max/min of two gaussian random variables. *IEEE Trans. VLSI Syst.* 16(2):210–212.

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

Pearl, J. 1983. On the nature of pathology in game searching. *Artif. Intell.* 20(4):427–453.

Ramanujan, R., and Selman, B. 2011. Trade-offs in sampling-based adversarial planning. To appear in 21st ICAPS.

Ramanujan, R.; Sabharwal, A.; and Selman, B. 2010a. On adversarial search spaces and sampling-based planning. In *20th ICAPS*, 242–245.

Ramanujan, R.; Sabharwal, A.; and Selman, B. 2010b. Understanding sampling style adversarial search methods. In *26th UAI*.

Schaeffer, J.; Culberson, J.; Treloar, N.; Knight, B.; Lu, P.; and Szafron, D. 1992. A world championship caliber checkers program. *Artif. Intell.* 53:273–289.

Scheucher, A., and Kaindl, H. 1998. Benefits of using multivalued functions for minimaxing. *AI J.* 99(2):187 – 208.