Towards an Analysis of Minimaxing of Game Trees with Random Leaf Values *

Mark Levene
University College London
University of London
Gower Street, London
WC1E 6BT, U.K.
email: mlevene@cs.ucl.ac.uk

Trevor Fenner
Birkbeck College
University of London
Malet Street, London
WC1E 7HX, U.K.
email: trevor@dcs.bbk.ac.uk

Abstract

Random minimaxing introduced in [BEAL94] is the process of using a random static evaluation function for scoring the leaf nodes of a full-width game tree and then computing the best move using the standard minimax procedure. The experiments carried out by Beal and Smith [BEAL94] using random minimaxing in chess showed that the strength of play increases as the depth of the lookahead is increased. We investigate random minimaxing from a combinatorial point of view in order to obtain a theoretical justification for Beal and Smith's experiments. In particular we show that, with respect to chess, random minimaxing with the depth of lookahead equal to two is "stronger" than random minimaxing with the depth of lookahead equal to one, under the assumption that the more a move made by the first player restricts the second player's choice of moves (i.e. the second player's mobility) the better that move is. We conjecture that these results can be generalized for depths of lookahead greater than two.

1 Introduction

A standard search technique used in game playing in order to decide which move to play is the well-known *minimax* procedure [ADEL88, KAIN90, LEVY91]. The depth of lookahead used in the minimax procedure corresponds to the number of levels or *ply* of the game tree being searched; a game tree with $\delta + 1$ levels is called a δ-ply game tree. The scoring of the positions represented by the leaf nodes of a game tree is carried out by a *static evaluation function*, which attaches a numerical value to each of these positions. *Random minimaxing*, introduced in [BEAL94], is the process of applying the minimax procedure to full-width game trees while using a random number generator as the static evaluation function. The experiments carried out by Beal and Smith [BEAL94] using random minimaxing in chess showed that the strength of play increases as the depth of the lookahead is increased.

Herein we investigate random minimaxing from a combinatorial point of view in order to obtain a theoretical justification for Beal and Smith's experiments. We set up the machinery needed for our investigation by defining the probability that a given leaf node be returned by random minimaxing. We give the equations for this probability with respect to δ-ply game trees, where $1 \leq \delta \leq 3$, and indicate how it is possible to construct a general equation when $\delta > 3$. We show that when $\delta = 1$ the probabilities of all leaf nodes are the same but when $\delta > 1$, in general, these probabilities are different; i.e. when $\delta > 1$ random minimaxing does *not* correspond to sampling from a *uniform distribution* [FELL68]. We then investigate the relative probabilities of leaf nodes of 2-ply game trees and show that leaf nodes which have fewer siblings are more probable. We extend this result to the probability that a given move be chosen as a result of random minimaxing and show that moves which reach positions representing nodes which have fewer children are more probable. As a

*Appeared in *ICCA Journal*, **18**, (1995), 20-33.

corollary of this result, we show that in chess random minimaxing with respect to 2-ply game trees is "stronger" than random minimaxing with respect to 1-ply game trees, under the assumption that the more a move made by the first player restricts the second player's choice of moves (i.e. the second player's mobility) the better that move is.

The layout of the rest of the paper is as follows. In Section 2 we introduce the minimax procedure and in Section 3 we introduce random minimaxing. In Section 4 we give equations for enumerations which are needed in the remaining sections. In Section 5 we investigate the probability that a given leaf node be returned by random minimaxing of a game tree and in Section 6 we investigate the relative probabilities of leaf nodes being returned by random minimaxing of 2-ply game trees. In Section 7 we investigate the relative probabilities of moves being chosen as a result of random minimaxing of 2-ply game trees and show that in chess, under the assumption mentioned above, random minimaxing of 2-ply game trees is "stronger" than random minimaxing of 1-ply game trees. Finally, in Section 8 we give our concluding remarks and conjecture that these results can be generalized for depths of lookahead greater than two.

2 The Minimax Procedure

We assume the reader is familiar with the general idea of a minimax procedure and the concept of a game tree [ADEL88, KAIN90, LEVY91], where nodes represent game positions and arcs represent *legal* moves from one position to another; the *root* node represents the current position. When no ambiguity arises we will not distinguish between the nodes and the positions they represent and between the arcs and the moves they represent. We are assuming a two player perfect information game between the first player, called *white*, and the second player, called *black*, where the game has three possible outcomes: win for white (i.e. loss for black), loss for white (i.e. win for black), or draw (see [GUY91] for a precise definition of a game).

In a δ-ply game tree, the root node is at level 0 and all the leaf nodes are at level δ; we assume throughout that $\delta > 0$. Non-leaf nodes of a game tree are called *internal nodes*. A game tree satisfying the condition that each internal node has an arc for each possible legal move from the position represented by that node is called a *full-width* game tree. From now on we assume that all game trees are full-width game trees. We call a game tree *level-regular* if all nodes on the same level have the same number of children.

Let minimax(T, δ, $score$, α) be a procedure which returns the leaf node of the *principal variation* [KAIN90] chosen by minimaxing, where T is the δ-ply game tree whose root represents the current position, *score* is a *static evaluation function*, and α is a natural number representing the maximum possible score. The principal variation is represented by the path from the root to the leaf node returned by minimax(T, $\delta, score, \alpha$). We assume that the scoring of leaf nodes is computed by the function *score*, which returns a natural number between 1 and α inclusive. For the purpose of scoring we have assumed that all leaf nodes are distinct, although in practice two distinct leaf nodes may represent the same position through a transposition of moves [LEVY91].

In general, it is possible that there is more than one principal variation, in which case the minimax procedure returns the set of leaf nodes of all the principal variations. This does not cause us any problems, since we will only be interested to know whether a particular leaf node, say n_0, is returned by the minimax procedure or not.

Let T be a δ-ply game tree with $\delta > 0$ and n be a node in T. We define the following operators on T:

1. $par(n)$ returns the parent node of n; $par(par(\dots par(n)\dots))$, where par appears $t \geq 1$ times, is abbreviated to $par^t(n)$.

2. $ch(n)$ returns the set of children nodes of n; if n is a leaf node then $ch(n) = \emptyset$. We denote the cardinality of this set, i.e. $|ch(n)|$, by $\chi(n)$.

3. *root*(T) returns the single root node of T.

4. *leaves*(T) returns the set of leaf nodes of T.

5. *moves*(T) returns $ch(root(\mathrm{T}))$, i.e. the set of nodes representing the possible positions arrived at after white makes a single move.

6. $sib(n)$ returns $ch(par(n)) - \{n\}$.

For brevity we often write $sc(n)$ for $score(n)$. We will assume that the minimax procedure maximizes the score of siblings at an odd level and minimizes the score of siblings at an even level. In the following we let n_0 be a leaf node returned by the minimax procedure.

We now develop a first order formula, denoted by $\phi_\delta^{n_0}$ (or simply ϕ_δ whenever n_0 is understood from context), where $n_0 \in leaves(\mathrm{T})$, which is true if and only if n_0 is a leaf node returned by minimax(T, $\delta, score, \alpha$). When $\delta = 0$, we define ϕ_0 and ϕ_0' to be true.

The first order formula ϕ_1 is given by

$$\phi_1 \quad = \quad \forall n_1 \in sib(n_0), sc(n_1) \leq sc(n_0).$$

Clearly ϕ_1 is true if and only if n_0 is a leaf node returned by minimax(T, 1, $score, \alpha$). An auxiliary first order formula, denoted by ϕ_1', is defined by

$$\phi_1' \quad = \quad \forall n_1 \in sib(n_0), sc(n_1) \geq sc(n_0).$$

The first order formula ϕ_2 is given by

$$\phi_2 \quad = \quad \phi_1' \wedge \forall n_1 \in sib(par(n_0)), \exists n_2 \in ch(n_1), sc(n_2) \leq sc(n_0).$$

Similarly, it can be verified that ϕ_2 is true if and only if n_0 is a leaf node returned by minimax(T, 2, $score, \alpha$). An auxiliary first order formula, denoted by ϕ_2', is given by

$$\phi_2' \quad = \quad \phi_1 \wedge \forall n_1 \in sib(par(n_0)), \exists n_2 \in ch(n_1), sc(n_2) \geq sc(n_0).$$

The first order formula for ϕ_3 is given by

$$\phi_3 \quad = \quad \phi_2' \wedge \forall n_1 \in sib(par^2(n_0)), \exists n_2 \in ch(n_1), \forall n_3 \in ch(n_2), sc(n_3) \leq sc(n_0).$$

As above, it can be verified that ϕ_3 is true if and only if n_0 is a leaf node returned by minimax(T, 3, $score, \alpha$). The general first order formula ϕ_δ, when δ is *odd*, is given by

$$\begin{aligned}\phi_\delta \quad = \quad & \phi_{\delta-1}' \wedge \forall n_1 \in sib(par^{\delta-1}(n_0)), \exists n_2 \in ch(n_1), \forall n_3 \in ch(n_2), \ldots, \\ & \exists n_{\delta-1} \in ch(n_{\delta-2}), \forall n_\delta \in ch(n_{\delta-1}), sc(n_\delta) \leq sc(n_0),\end{aligned}$$

where $\phi_{\delta-1}'$ is given by

$$\begin{aligned}\phi_{\delta-1}' \quad = \quad & \phi_{\delta-2} \wedge \forall n_1 \in sib(par^{\delta-2}(n_0)), \exists n_2 \in ch(n_1), \forall n_3 \in ch(n_2), \ldots, \\ & \forall n_{\delta-2} \in ch(n_{\delta-3}), \exists n_{\delta-1} \in ch(n_{\delta-2}), sc(n_{\delta-1}) \geq sc(n_0).\end{aligned}$$

Now, it can be verified that ϕ_δ, where δ is odd, is true if and only if n_0 is a leaf node returned by minimax(T, $\delta, score, \alpha$). The general first order formula ϕ_δ, when δ is *even*, is given by

$$\begin{aligned}\phi_\delta \quad = \quad & \phi_{\delta-1}' \wedge \forall n_1 \in sib(par^{\delta-1}(n_0)), \exists n_2 \in ch(n_1), \forall n_3 \in ch(n_2), \ldots, \\ & \forall n_{\delta-1} \in ch(n_{\delta-2}), \exists n_\delta \in ch(n_{\delta-1}), sc(n_\delta) \leq sc(n_0),\end{aligned}$$

where $\phi_{\delta-1}'$ is given by

$$\begin{aligned}\phi_{\delta-1}' \quad = \quad & \phi_{\delta-2} \wedge \forall n_1 \in sib(par^{\delta-2}(n_0)), \exists n_2 \in ch(n_1), \forall n_3 \in ch(n_2), \ldots, \\ & \exists n_{\delta-2} \in ch(n_{\delta-3}), \forall n_{\delta-1} \in ch(n_{\delta-2}), sc(n_{\delta-1}) \geq sc(n_0).\end{aligned}$$

Similarly, it can be verified that ϕ_δ, where δ is even, is true if and only if n_0 is a leaf node returned by minimax(T, $\delta, score, \alpha$).

The following observation is useful in verifying the correctness of ϕ_δ. Given a set of natural numbers S and a natural number μ, if the maximum number in S greater than or equal to μ (respectively, the minimum number in S is less than or equal to μ) then *at least one* number in S must be greater than or equal to μ (respectively, *at least one* number in S must be less than or equal to μ). Correspondingly, if the maximum number in S is less than or equal to μ (respectively, the minimum number in S is greater than or equal to μ) then *all* the numbers in S must be less than or equal to μ (respectively, *all* the numbers in S must be greater than or equal to μ).

3 Random Minimaxing

We assume that a probabilistic function *random*(α) is available and that it returns a natural number uniformly distributed between 1 and α; *random*(α) corresponds to rolling an unbiased α-sided dice. We will use the function *random*(α) as a static evaluation function for scoring the leaf nodes of a game tree.

The procedure minimax(T, $\delta, score, \alpha$) defines a *strategy* of playing a particular game, say chess, with the root node of T representing the current position. We call such a strategy the *game playing automaton* defined by the procedure minimax(T, $\delta, score, \alpha$). We will denote the game playing automaton defined by the procedure minimax(T, δ, *random*, α) by $\mathcal{A}^{(\delta,\alpha)}$ (or simply $\mathcal{A}^{\delta}$ when α is understood from context), with T being dependent on the current position and δ, and where α is fixed. We will refer to the evaluation of minimax(T, $\delta, random, \alpha$) by $\mathcal{A}^{\delta}$ as *random minimaxing.*

We are interested in investigating the probability that $\mathcal{A}^{\delta_1}$ *wins* against $\mathcal{A}^{\delta_2}$, i.e. that $\mathcal{A}^{\delta_1}$ is "stronger" than $\mathcal{A}^{\delta_2}$, where $\delta_2 \leq \delta_1$, under the assumption that it is decided randomly which player will start. We denote this probability by $win(\delta_1, \delta_2)$, where here we discount drawn games. Obviously, $win(\delta, \delta) = 1/2$, so we will assume from now on that $\delta_2 < \delta_1$.

Finally, we will also assume that leaf nodes which represent *terminal* positions, i.e. positions which are won, lost or drawn for the player to move, are also evaluated randomly. We make the following modification to a δ-ply game tree, say T, which has a leaf node representing a terminal position at a level $\delta_0 < \delta$. We extend T by adding nodes representing *dummy positions* and arcs representing *dummy moves* in such a way that all the paths from the root of the extended tree to its leaf nodes are of length δ; T is extended in such a way that every internal node representing a terminal or dummy position has exactly one child. From now on we assume that all δ-ply game trees are conveniently extended in the above manner.

The motivation for evaluating terminal nodes randomly is to avoid giving an unfair advantage to $\mathcal{A}^{\delta_1}$ simply by the fact that a terminal position has been reached and the outcome of the game may be recognized. Our approach to evaluating terminal positions is different from the one taken in [BEAL94] wherein terminal positions are recognized and evaluated as a win, loss or draw according to the rules of the game. In order not to give an unfair advantage to $\mathcal{A}^{\delta_1}$, [BEAL94] augments $\mathcal{A}^{\delta_2}$ with the additional capability of recognizing terminal positions at all levels less than or equal to δ_1.

4 Enumeration Equations

Herein we give equations for enumerations which are needed in order to obtain the results in the remaining sections. The equations are derived from the logical formulae introduced in Section 2.

Throughout the rest of the paper we assume that i is a natural number between 1 and α, which denotes the score of any of the leaf nodes returned by the minimax procedure.

Firstly, we define $LE(\exists, n_j)$ to be the number of 1-ply game trees rooted at n_j having at least one leaf node whose score is less than or equal to i. $LE(\exists, n_j)$ is given by

$$LE(\exists, n_j) = \sum_{k=1}^{\chi(n_j)} \binom{\chi(n_j)}{k} i^k (\alpha - i)^{\chi(n_j)-k},$$

where k represents the number of leaf nodes whose score is less than or equal to i. (For consistency we define 0^0 to be 1.) The above equation can be simplified to

$$LE(\exists, n_j) = \alpha^{\chi(n_j)} - (\alpha - i)^{\chi(n_j)}$$

on using the binomial theorem:

$$(a+b)^m = \sum_{k=0}^{m} \binom{m}{k} a^{m-k} b^k. \tag{1}$$

Similarly, we define $LE(\forall, n_j)$ to be the number of 1-ply game trees rooted at n_j with all of its leaf nodes having scores less than or equal to i. $LE(\forall, n_j)$ is given by

$$LE(\forall, n_j) = i^{\chi(n_j)}.$$

Analogously, we define $GE(\exists, n_j)$ to be the number of 1-ply game trees rooted at n_j having at least one leaf node whose score is greater than or equal to i. $GE(\exists, n_j)$ is given by

$$GE(\exists, n_j) = \sum_{k=1}^{\chi(n_j)} \binom{\chi(n_j)}{k} (\alpha - i + 1)^k (i-1)^{\chi(n_j)-k}.$$

On using (1) the above equation can be simplified to

$$GE(\exists, n_j) = \alpha^{\chi(n_j)} - (i-1)^{\chi(n_j)}.$$

Next we define $GT(\exists, n_j)$ to be the number of 1-ply game trees rooted at n_j having at least one leaf node whose score is greater than i. $GT(\exists, n_j)$ is given by

$$GT(\exists, n_j) = \sum_{k=1}^{\chi(n_j)} \binom{\chi(n_j)}{k} (\alpha - i)^k \; i^{\chi(n_j)-k}.$$

On using (1) the above equation can be simplified to

$$GT(\exists, n_j) = \alpha^{\chi(n_j)} - i^{\chi(n_j)}.$$

We now define $EQ(\exists, n_j)$ to be the number of 1-ply game trees rooted at n_j having one or more leaf nodes whose score is equal to i and such that *all other leaf nodes have scores greater than i.* $EQ(\exists, n_j)$ is given by

$$EQ(\exists, n_j) = \sum_{k=1}^{\chi(n_j)} \binom{\chi(n_j)}{k} (\alpha - i)^{\chi(n_j)-k}.$$

On using (1) the above equation can be simplified to

$$EQ(\exists, n_j) = (\alpha - i + 1)^{\chi(n_j)} - (\alpha - i)^{\chi(n_j)}.$$

Finally, we define $LE(\exists\forall, n_j)$ to be the number of 2-ply game trees rooted at n_j having at least one child such that all of its children (i.e. n_j's grandchildren, which are in fact leaf nodes) have scores less than or equal to i. For this purpose we let $\rho_k(n_j)$ denote the set of all partitions of $ch(n_j)$ into two sets of nodes, N_1 and N_2, such that $|N_1| = k$, where $1 \leq k \leq \chi(n_j)$. The cardinality of the set $\rho_k(n_j)$ is $\binom{\chi(n_j)}{k}$. $LE(\exists\forall, n_j)$ is given by

$$\begin{aligned}
LE(\exists\forall, n_j) &= \sum_{k=1}^{\chi(n_j)} \sum_{(N_1,N_2)\in\rho_k(n_j)} \prod_{n_{q1}\in N_1} LE(\forall, n_{q1}) \prod_{n_{q2}\in N_2} GT(\exists, n_{q2}) \\
&= \sum_{k=1}^{\chi(n_j)} \sum_{(N_1,N_2)\in\rho_k(n_j)} \prod_{n_{q1}\in N_1} i^{\chi(n_{q1})} \prod_{n_{q2}\in N_2} \left(\alpha^{\chi(n_{q2})} - i^{\chi(n_{q2})}\right) \\
&= \prod_{n_q\in ch(n_j)} \alpha^{\chi(n_q)} - \prod_{n_q\in ch(n_j)} \left(\alpha^{\chi(n_q)} - i^{\chi(n_q)}\right) \\
&= \alpha^{\sum_{n_q\in ch(n_j)} \chi(n_q)} - \prod_{n_q\in ch(n_j)} \left(\alpha^{\chi(n_q)} - i^{\chi(n_q)}\right).
\end{aligned}$$

The above derivation requires explanation. Let $C(n_q)$ and $D(n_q)$ be natural numbers for each $n_q \in \chi(n_j)$. Then we can derive the following equality (which is essentially a generalization of the binomial theorem) by expanding its left-hand side.

$$\prod_{n_q\in ch(n_j)} (C(n_q) + D(n_q)) = \sum_{k=0}^{\chi(n_j)} \sum_{(N_1,N_2)\in\rho_k(n_j)} \prod_{n_{q1}\in N_1} C(n_{q1}) \prod_{n_{q2}\in N_2} D(n_{q2}).$$

In order for the summation to start from $k = 1$ we need to subtract $\prod_{n_q\in ch(n_j)} D(n_q)$ from both sides of the equation.

5 The Probabilities of Leaf Nodes in Random Game Playing

We will now analyze the probability that a given leaf node, say n_0, of a δ-ply game tree is returned by the procedure minimax(T, δ, *random*, α).

We denote the number of leaf nodes of a game tree T by #T, i.e. #T = |*leaves*(T)|. (In general, for a set of nodes S, we will denote the cardinality of S by |S|.) In addition, we let $\#\phi_\delta^{n_0}(i)$ denote the number of possible assignments of scores to the leaf nodes of T such that minimax(T, δ, *random*, α) returns n_0 and $sc(n_0) = i$. Finally, $\#\phi_\delta^{n_0}$ is defined by

$$\#\phi_\delta^{n_0} = \sum_{i=1}^{\alpha} \#\phi_\delta^{n_0}(i).$$

That is, $\#\phi_\delta^{n_0}$ is the total number of possible assignments of scores to the leaf nodes of T which make the formula $\phi_\delta^{n_0}$ true.

For simplicity we write $\#\phi_\delta(i)$ and $\#\phi_\delta$ whenever n_0 can be understood from context. We define $\#\phi'_\delta(i)$ and $\#\phi'_\delta$ in an analogous manner.

Proposition 5.1 *The total number of possible assignments of scores to the leaf nodes of a game tree T is given by $\alpha^{\#T}$.* $\square$

It follows that the probability, denoted by prob(n_0), that a leaf node n_0 is returned by minimax(T, δ, *random*, α) is given by

$$\mathrm{prob}(n_0) = \frac{\#\phi_\delta^{n_0}}{\alpha^{\#T}}.$$

It is important to observe that, given leaf nodes $n_1, n_2, \ldots, n_k$, the probability of any of these leaf nodes being returned by minimax(T, δ, *random*, α) is not, in general, the sum of prob(n_j) over $j \in \{1, 2, \ldots, k\}$, since these events are not necessarily *mutually exclusive* [FELL68]. For example, if by chance all the leaf nodes are assigned the same score, then minimax(T, δ, *random*, α) returns all of the leaf nodes.

We begin our analysis by giving the equation for $\#\phi_1$.

Lemma 5.2 *$\#\phi_1$ is given by*

$$\#\phi_1 = \sum_{i=1}^{\alpha} \#\phi_1(i) = \sum_{i=1}^{\alpha} i^{|sib(n_0)|}. \quad \square \tag{2}$$

The following lemma gives the equation for $\#\phi_1'(i)$ which will be useful later.

Lemma 5.3 *$\#\phi_1'(i)$ is given by*

$$\#\phi_1'(i) = (\alpha - i + 1)^{|sib(n_0)|}. \quad \square$$

It follows from the above two lemmas that $\#\phi_1' = \#\phi_1$.

The following lemma, which is an immediate consequence of Lemma 5.2, states the intuitively clear result that for 1-ply game trees all leaf nodes are equally likely to be chosen by the minimax procedure.

Lemma 5.4 *When $\delta = 1$, then $\forall n_0, n_1 \in leaves(T)$,*

$$prob(n_0) = prob(n_1) = \frac{\#\phi_1^{n_0}}{\alpha^{\#T}} = \frac{\#\phi_1^{n_1}}{\alpha^{\#T}}. \quad \square$$

In fact, even before a full analysis is undertaken, we can show that if we restrict ourselves to level-regular game trees then the number of ply of the game trees operated upon does not affect the result of random minimaxing. This does not contradict the results of [BEAL94], since they make no assumption with respect to game trees being level-regular or not. Moreover, the reader can verify that game trees for chess are, in general, not level-regular.

Proposition 5.5 *Assuming that the δ-ply game tree T is level-regular, then $\forall n_0, n_1 \in leaves(T)$, $prob(n_0) = prob(n_1)$.* $\square$

The above proposition follows by symmetry. The result would not hold if the game tree was not level-regular, as can be verified, for example, by considering the leaf nodes a and b of the 2-ply game tree shown in Figure 1 (see Remark 5.8).

Another trivial case, when the probabilities of all leaf nodes are the same and independent of δ, is when $\alpha = 1$. This and the above proposition yield the following corollary.

Corollary 5.6 *Assuming that either all the game trees input to the minimax procedure are level-regular or that $\alpha = 1$, then $win(\delta_1, \delta_2) = 1/2$ for all δ_1 and δ_2.* $\square$

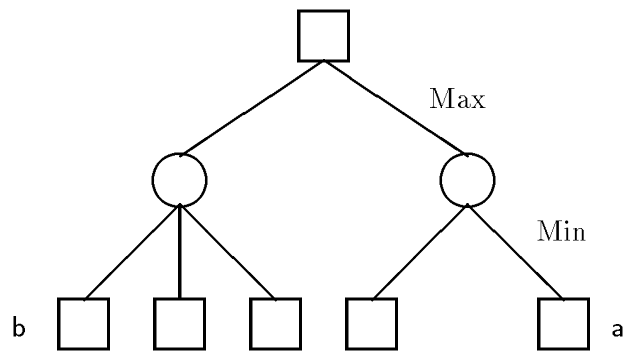

Figure 1: A 2-ply game tree

We are now ready to derive the equation for $\#\phi_2$.

Lemma 5.7 *$\#\phi_2$ is given by*

$$\begin{aligned} \#\phi_2 &= \sum_{i=1}^{\alpha} \#\phi'_1(i) \prod_{n_1 \in sib(par(n_0))} LE(\exists, n_1) \\ &= \sum_{i=1}^{\alpha} (\alpha - i + 1)^{|sib(n_0)|} \prod_{n_1 \in sib(par(n_0))} \left(\alpha^{\chi(n_1)} - (\alpha - i)^{\chi(n_1)}\right). \quad \square \end{aligned} \tag{3}$$

Remark 5.8 *In general, when $\delta = 2$ the probabilities of leaf nodes are not all the same.*

Proof. Let T be the game tree shown in Figure 1 and assume that $\alpha = 2$. It can be verified that prob(a) $= 22/32$, prob(b) $= 16/32$, and thus prob(a) $>$ prob(b). $\square$

The following lemma gives the equation for $\#\phi'_2$, which is needed in order to obtain the equation for $\#\phi_3$.

Lemma 5.9 *$\#\phi'_2(i)$ is given by*

$$\begin{aligned} \#\phi'_2(i) &= \#\phi_1(i) \prod_{n_1 \in sib(par(n_0))} GE(\exists, n_1) \\ &= i^{|sib(n_0)|} \prod_{n_1 \in sib(par(n_0))} (\alpha^{\chi(n_1)} - (i-1)^{\chi(n_1)}). \quad \square \end{aligned}$$

The next lemma provides the equation for $\#\phi_3$.

Lemma 5.10 *$\#\phi_3$ is given by*

$$\begin{aligned} \#\phi_3 &= \sum_{i=1}^{\alpha} \#\phi'_2(i) \prod_{n_1 \in sib(par^2(n_0))} LE(\exists\forall, n_1) \\ &= \sum_{i=1}^{\alpha} \#\phi'_2(i) \prod_{n_1 \in sib(par^2(n_0))} \left(\alpha^{\sum_{n_q \in ch(n_1)} \chi(n_q)} - \prod_{n_q \in ch(n_1)} \left(\alpha^{\chi(n_q)} - i^{\chi(n_q)}\right) \right). \quad \square \end{aligned} \tag{4}$$

Remark 5.11 *In general, when $\delta = 3$ the probabilities of leaf nodes are not all the same.*

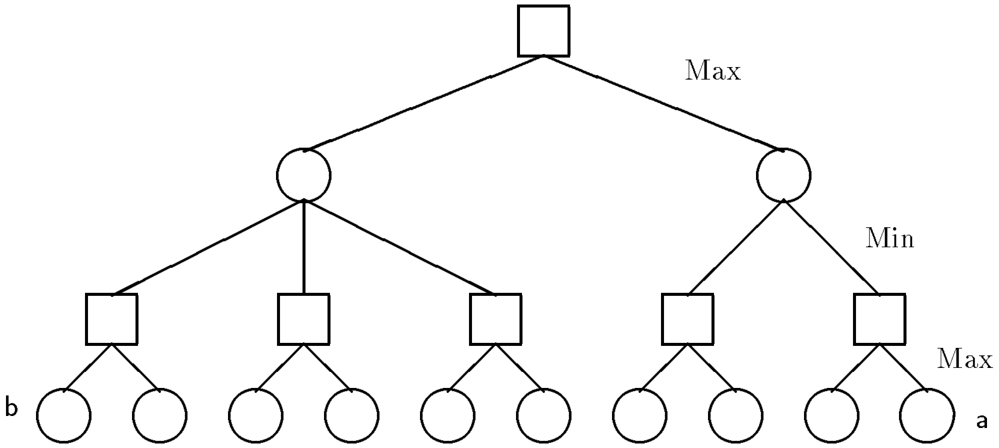

Figure 2: A 3-ply game tree

Proof. Let T be the game tree shown in Figure 2 and assume that $\alpha = 2$. It can be verified that prob(a) $= 532/1024$, prob(b) $= 400/1024$, and thus prob(a) $>$ prob(b). $\square$

We do not provide the general equation for $\#\phi_\delta$, due to its complexity. However, we make the following observation which indicates how the general equation can be constructed. Whenever a node in ϕ_δ is universally quantified we insert a product into the equation and whenever a node in ϕ_δ is existentially quantified we insert a double sum of two products into the equation, which is partitioned according to how many child nodes satisfy the subformula of ϕ_δ beginning at that node. (Cf. the construction of the formula for $LE(\exists\forall, n_j)$ at the end of Section 4.) At the level of the leaf nodes all the siblings of n_0 must have a score less than or equal to $sc(n_0)$.

It is easy to generalize Remarks 5.8 and 5.11 in order to obtain the following remark by considering the extension of the game tree shown in Figure 2 with all internal nodes at levels greater than two having only one child.

Remark 5.12 *In general, when $\delta > 1$ the probabilities of leaf nodes are not all the same.* $\square$

6 The Relative Probabilities of Leaf Nodes in 2-Ply Game Trees

In this section we show that, for $\delta = 2$ and for all $\alpha > 1$, leaf nodes which have fewer siblings are more probable.

For any $n_x, n_y \in moves(\mathrm{T})$, we let

$$M_i(n_x, n_y) = \prod_{n_1 \in moves(T) - \{n_x, n_y\}} \left(\alpha^{\chi(n_1)} - (\alpha - i)^{\chi(n_1)}\right). \tag{5}$$

The next theorem characterizes the relative probabilities of leaf nodes of 2-ply game trees.

Theorem 6.1 *Let T be a 2-ply game tree, and let n_a and n_b be two leaf nodes. Then, assuming that $\alpha > 1$, prob(n_a) $>$ prob(n_b) if and only if $|sib(n_a)| < |sib(n_b)|$.*

Proof. For simplicity, we denote $|\mathrm{sib}(n_a)|$ by s_a and $|\mathrm{sib}(n_b)|$ by s_b. By the definition of the probability of a leaf node it follows that prob(n_a) $>$ prob(n_b) if and only if $\#\phi_2^{n_a} > \#\phi_2^{n_b}$, i.e. if and only if $\sum \#\phi_2^{n_a}(i) > \sum \#\phi_2^{n_b}(i)$.

In the following we assume that $1 \leq i \leq \alpha$. The next two equations follow from (3).

$$\#\phi_2^{n_a}(i) = (\alpha - i + 1)^{s_a} M_i(par(n_a), par(n_b))(\alpha^{s_b+1} - (\alpha - i)^{s_b+1}). \tag{6}$$

$$\#\phi_2^{n_b}(i) = (\alpha - i + 1)^{s_b} M_i(par(n_a), par(n_b))(\alpha^{s_a+1} - (\alpha - i)^{s_a+1}). \tag{7}$$

If. Suppose that $s_a < s_b$. We show that for all $i \in \{1,2,\ldots,\alpha\}$, $\#\phi_2^{n_a}(i) > \#\phi_2^{n_b}(i)$, which implies that $\#\phi_2^{n_a} > \#\phi_2^{n_b}$. From (6) and (7) this holds if and only if

$$(\alpha - i + 1)^{s_a}\left(\alpha^{s_b+1} - (\alpha - i)^{s_b+1}\right) \quad > \quad (\alpha - i + 1)^{s_b}\left(\alpha^{s_a+1} - (\alpha - i)^{s_a+1}\right),$$

or equivalently,

$$\alpha^{s_b+1} - (\alpha - i)^{s_b+1} \quad > \quad (\alpha - i + 1)^{s_b - s_a}\alpha^{s_a+1} - (\alpha - i + 1)^{s_b - s_a}(\alpha - i)^{s_a+1}.$$

It is easy to see that, for all $i \in \{1,2,\ldots,\alpha\}$, $\alpha^{s_b+1} \geq (\alpha - i + 1)^{s_b - s_a}\alpha^{s_a+1}$ with equality if and only if $i = 1$. Also, $(\alpha - i)^{s_b+1} \leq (\alpha - i + 1)^{s_b - s_a}(\alpha - i)^{s_a+1}$ with equality if and only if $i = \alpha$. The result now follows.

Only if. Suppose that $\#\phi_2^{n_a} > \#\phi_2^{n_b}$. It follows that $s_a \neq s_b$, since if $s_a = s_b$ then $\#\phi_2^{n_a} = \#\phi_2^{n_b}$ would follow from equations (6) and (7). Now, if $s_b < s_a$ then, by the *if* part of the theorem, it would follow that $\#\phi_2^{n_b} > \#\phi_2^{n_a}$, which leads to a contradiction. The result follows. □

7 2-Ply Random Minimaxing Wins Against 1-Ply Random Minimaxing

In this section we extend the results of Section 6 by extending the probability of leaf nodes to the probability that a given node in *moves*(T) be chosen as a result of random minimaxing. We then give a parallel result to Theorem 6.1 by showing that nodes in *moves*(T) which have fewer children are more probable. This result holds for all $\alpha > 1$. We show that the following assumption, when made for the game of chess, results in the corollary that in chess random minimaxing with respect to 2-ply game trees is "stronger" than random minimaxing with respect to 1-ply game trees.

The mobility assumption *The more a move made by white restricts black's choice of moves the better that move is.*

Let σ be an assignment of scores to the leaf nodes of a game tree T, i.e. σ is a static evaluation function. Furthermore, let $\#[n](i)$, where $n \in moves(\mathrm{T})$, denote the number of distinct assignments σ of scores to the leaf nodes of T such that n is included in a principal variation induced by one of the leaf nodes n_0 returned by minimax(T, δ, σ, α) where $sc(n_0) = i$. Now let

$$\#[n] = \sum_{i=1}^{\alpha} \#[n](i).$$

When $\delta = 2$, then

$$\begin{aligned} \#[n] \quad = \quad & |\,\{\sigma \mid \sigma \text{ is an assignment of scores to the leaf nodes of T and} \\ & \text{some } n_0 \in ch(n) \text{ is returned by } minimax(T, 2, \sigma, \alpha)\}\,|\,. \end{aligned}$$

We extend prob(n_0) where $n_0 \in leaves(\mathrm{T})$ to prob(n) where $n \in moves(\mathrm{T})$ as follows:

$$\mathrm{prob}(n) = \frac{\#[n]}{\alpha^{\#T}}.$$

The next result follows by symmetry.

Proposition 7.1 *Let T be a δ-ply game tree, $n_x, n_y \in moves(T)$, and let the subtrees of T rooted at n_x and n_y be isomorphic. Then $prob(n_x) = prob(n_y)$.* $\square$

From now on we will restrict our analysis to δ-ply game trees with $\delta \in \{1,2\}$. In what follows let T be a 2-ply game tree and $n \in moves(\mathrm{T})$. We are now ready to derive the equation for $\#[n]$.

Lemma 7.2 *$\#[n]$ is given by*

$$\begin{aligned}\#[n] &= \sum_{i=1}^{\alpha} EQ(\exists, n) \prod_{n_1 \in sib(n)} LE(\exists, n_1) \\ &= \sum_{i=1}^{\alpha} \left((\alpha - i + 1)^{\chi(n)} - (\alpha - i)^{\chi(n)}\right) \prod_{n_1 \in sib(n)} \left(\alpha^{\chi(n_1)} - (\alpha - i)^{\chi(n_1)}\right). \quad \square \qquad (8)\end{aligned}$$

Let T be the game tree shown in Figure 1 and assume that $\alpha = 2$. Using Lemma 7.2 it can be verified that prob(par(a)) = 29/32, prob(par(b)) = 25/32, and thus prob(par(a)) > prob(par(b)).

For a particular score i,

$$\#[n](i) = \left((\alpha - i + 1)^{\chi(n)} - (\alpha - i)^{\chi(n)}\right) \prod_{n_1 \in sib(n)} \left(\alpha^{\chi(n_1)} - (\alpha - i)^{\chi(n_1)}\right). \qquad (9)$$

Our next objective is to study the value of $win(2, 1)$. The next theorem characterizes the relative probabilities of moves of 2-ply game trees (cf. Theorem 6.1).

Theorem 7.3 *Let T be a 2-ply game tree, and let $n_x, n_y \in moves(T)$ represent two possible moves. Then, assuming that $\alpha > 1$, $prob(n_x) > prob(n_y)$ if and only if $\chi(n_x) < \chi(n_y)$.*

Proof. For simplicity we denote $\chi(n_x)$ by c_x and $\chi(n_y)$ by c_y. By the definition of the probability of a node representing a move it follows that prob$(n_x) >$ prob(n_y) if and only if $\#[n_x] > \#[n_y]$, i.e. if and only if $\sum \#[n_x](i) > \sum \#[n_y](i)$.

In the following we assume that $1 \le i \le \alpha$. The next two equations follow from (9).

$$\#[n_x](i) = ((\alpha - i + 1)^{c_x} - (\alpha - i)^{c_x})\, M_i(n_x, n_y)\, (\alpha^{c_y} - (\alpha - i)^{c_y}). \qquad (10)$$

$$\#[n_y](i) = ((\alpha - i + 1)^{c_y} - (\alpha - i)^{c_y})\, M_i(n_x, n_y)\, (\alpha^{c_x} - (\alpha - i)^{c_x}). \qquad (11)$$

If. Suppose that $c_x < c_y$. In order to prove the result we claim that $\#[n_x](1) = \#[n_y](1)$ and that $\#[n_x](i) > \#[n_y](i)$ for all $i \in \{2, 3, \ldots, \alpha\}$. This implies that $\#[n_x] > \#[n_y]$.

The first part of the claim is immediate since putting $i = 1$ in (10) and (11) gives

$$\#[n_x](1) = (\alpha^{c_x} - (\alpha - 1)^{c_x})\, M_1(n_x, n_y)\, (\alpha^{c_y} - (\alpha - 1)^{c_y}) = \#[n_y](1).$$

When i > 1, the second part of the claim follows from (10) and (11) if and only if

$$((\alpha - i + 1)^{c_x} - (\alpha - i)^{c_x})\, (\alpha^{c_y} - (\alpha - i)^{c_y}) > ((\alpha - i + 1)^{c_y} - (\alpha - i)^{c_y})\, (\alpha^{c_x} - (\alpha - i)^{c_x}),$$

or equivalently,

$$\frac{\alpha^{c_y} - (\alpha - i)^{c_y}}{\alpha^{c_x} - (\alpha - i)^{c_x}} > \frac{(\alpha - i + 1)^{c_y} - (\alpha - i)^{c_y}}{(\alpha - i + 1)^{c_x} - (\alpha - i)^{c_x}}. \qquad (12)$$

If $\alpha = i$ then the result is immediate since $\frac{\alpha^{c_y}}{\alpha^{c_x}} > 1$, so we may assume that $\alpha > i$. Inequality (12) holds if and only if

$$\left(\frac{\left(\frac{\alpha}{\alpha-i}\right)^{c_y} - 1}{\left(\frac{\alpha}{\alpha-i}\right)^{c_x} - 1} \right) > \left(\frac{\left(\frac{\alpha-i+1}{\alpha-i}\right)^{c_y} - 1}{\left(\frac{\alpha-i+1}{\alpha-i}\right)^{c_x} - 1} \right).$$

Now, let β denote $\left(\frac{\alpha}{\alpha-i}\right)^{c_x}$ and γ denote $\left(\frac{\alpha-i+1}{\alpha-i}\right)^{c_x}$. On letting $\psi = \frac{c_y}{c_x} - 1$, it remains to show that

$$\frac{\beta^{\psi+1} - 1}{\beta - 1} \quad > \quad \frac{\gamma^{\psi+1} - 1}{\gamma - 1} \tag{13}$$

where $\beta > \gamma > 1$ and $\psi > 0$. Let $f(x) = \frac{x^{\psi+1}-1}{x-1}$ for $x > 1$. Differentiating $f(x)$ we get

$$\frac{df}{dx} \quad = \quad \frac{(1+\psi)x^{\psi}(x-1) - (x^{\psi+1} - 1)}{(x-1)^2} \quad = \quad \frac{(\psi x - \psi - 1)x^{\psi} + 1}{(x-1)^2}.$$

It remains to show that

$$(x-1)^2 \frac{df}{dx} = (\psi x - \psi - 1)x^{\psi} + 1 > 0. \tag{14}$$

It is well known that $e^t > 1+t$ for $t \neq 0$. So, on setting $t = x-1$, we get $e^{x-1} > x$. Furthermore, since $\psi > 0$, we obtain

$$e^{(x-1)\psi} > x^{\psi}. \tag{15}$$

Now, on setting $t = -(x-1)\psi$ in the inequality $e^t > 1+t$, noting that $x > 1$, we get

$$e^{-(x-1)\psi} > 1 - (x-1)\psi = 1 + \psi - \psi x. \tag{16}$$

Since $e^{-(x-1)\psi} > 0$ and both sides of (15) are positive, we may multiply together the left-hand sides of (15) and (16) and the right-hand sides of (15) and (16) to obtain

$$e^{-(x-1)\psi} e^{(x-1)\psi} > (1 + \psi - \psi x)x^{\psi}. \tag{17}$$

Simplifying (17) we get

$$1 > (1 + \psi - \psi x)x^{\psi},$$

which yields (14). It follows that $df/fx > 0$ for $x > 1$, which implies that f is strictly increasing. Therefore $f(\beta) > f(\gamma)$, proving (13) holds as required.

Only if. Suppose that $\#[n_x] > \#[n_y]$. It follows that $c_x \neq c_y$, since if $c_x = c_y$ then $\#[n_x] = \#[n_y]$ would follow from equations (10) and (11). Now, if $c_y < c_x$ then, by the *if* part of the theorem, it would follow that $\#[n_y] > \#[n_x]$, which leads to a contradiction. The result follows. □

The following corollary is a direct application of Theorem 7.3 and Lemma 5.4 to the game of chess. It shows that, under the mobility assumption, 2-ply game trees for chess are more effective than 1-ply game trees when using random minimaxing. This result gives a theoretical justification of the central experimental result given in [BEAL94] when $\delta \leq 2$.

Corollary 7.4 *Assume that the game under consideration is chess and that the mobility assumption holds, then win(2, 1) > 1/2.* □

Corollary 7.4 makes sense due to the fact that, by Theorem 7.3, random minimaxing with $\delta = 2$ has the indirect ability to recognize checkmate, since giving checkmate will leave the opponent with no choice of move (i.e. $\chi(n) = 1$). This follows by our extension of the game tree at terminal positions as described at the end of Section 3. Furthermore, random minimaxing with $\delta = 2$ will, in general, favour moves which give checks, since such moves normally restrict the opponent's choice of reply.

We observe that the above result is true for any game for which the mobility assumption is valid. In particular, it seems that this is true for the game of *Othello*, for which restricting the opponent's choice of moves is a well-known strategy [LEE90].

It is obvious that more leaf nodes are considered when a larger value of δ is utilized by the minimax procedure. In [BEAL94] it was suggested that the main reason that $win(2, 1) > 1/2$ is due to the fact that, when a larger number of positions is considered (i.e. the "mobility" is greater), it is more likely that a higher score will be returned and a better move chosen.

Theorem 7.3 refines the explanation of why $win(2, 1) > 1/2$, stating that it is mainly due to the irregularity of 2-ply game trees compared with the level-regularity of 1-ply game trees, and also the fact that random minimaxing is compatible with the mobility assumption. In particular, moves which reach positions representing nodes which have fewer children are more likely to be chosen by random minimaxing with respect to 2-ply game trees.

8 Concluding Remarks

We have presented a preliminary analysis of random minimaxing from a combinatorial point of view. Our results confirm those given in [BEAL94] for 2-ply game trees and provide some interesting insights into the nature of random game playing. In Corollary 7.4 we showed that in chess, under the mobility assumption, random minimaxing with respect to 2-ply game trees is "stronger" than with respect to 1-ply game trees. This will also hold for any other game for which the mobility assumption is valid. Furthermore, this result can also be viewed as providing some justification for the advice commonly given to chess players that "any plan is better than no plan".

We are currently attempting to extend our results, in particular Theorem 7.3, to δ-ply game trees for $\delta > 2$. We conjecture that, as for 2-ply game trees, whether or not the probability of a node n_x is greater than (or equal to) the probability of another node n_y is independent of α, provided $\alpha > 1$.

We are also trying to establish a theoretical justification for deducing that $win(\delta_1, \delta_2) > 1/2$ when $\delta_2 < \delta_1$, under the following

Extended mobility assumption *The more a move made by white restricts black's choice of moves and the less this move restricts white's subsequent choice of moves the better that move is.*

This suggests the following primitive strategy for white: *minimize black's mobility and maximize white's mobility.*

References

[ADEL88] G.M. Adelson-Velsky, V.L. Alazarov and M.V. Donsky, *Algorithms for Games.* NY, Springer-Verlag, 1988. ISBN 3-540-96629-3.

[BEAL94] D. Beal and M.C. Smith, Random evaluation in chess, *International Computer Chess Association Journal*, **17**, (1994), 3-9. ISSN 0920-234X.

[FELL68] W. Feller, *An Introduction to Probability Theory and its Applications, Third Edition.* NY, John Wiley and Sons, 1968. ISBN 0-471-25711-7.

[GUY91] R.K. Guy, What is a game? In: *Proceedings of Symposia in Applied Mathematics, Volume 43, Combinatorial Games*, 1-21, 1991. ISBN 0-8218-0166-X.

[KAIN90] H. Kaindl, Tree searching algorithms. In: *Computers, Chess and Cognition*, Eds. T.A. Marsland and J. Schaeffer, NY, Springer-Verlag, 133-158, 1990. ISBN 0-387-97415-6.

[LEE90] K.-F. Lee and S. Mahajan, The development of a world class Othello program. *Artificial Intelligence*, **43**, (1990), 21-36. ISSN 0004-3702.

[LEVY91] D. Levy and M. Newborn, *How Computers Play Chess.* New York, W.H. Freeman, 1991. ISBN 0-7167-8121-2.