Decision Quality As a Function of Search Depth on Game Trees

DANA S. NAU

University of Maryland, College Park, Maryland

Abstract. One important application of tree searching is "looking ahead" on a decision tree or game tree to try to predict the results of a decision. To guarantee correct results, substantial portions of the tree must be completely searched, which is physically impossible for very large trees.

Artificial intelligence researchers have obtained good results by searching the tree to some arbitrary depth and using a static evaluation function to estimate the values of the nodes at that depth. It is generally believed that when this is done, the quality of the decision improves as the search depth increases. This belief is based purely on empirical evidence.

The author has developed a mathematical theory modeling the effects of search depth on the probability of making a correct decision. In this theory, the errors made by the evaluation function are modeled as independent, identically distributed random errors superimposed on the true values of the nodes evaluated. This research has produced the surprising result that there is an infinite class of game trees for which searching deeper does not increase the probability of making a correct decision, but instead causes the decision to become more and more random. The paper contains a mathematical proof of this statement, experimental verification of it, and a discussion of its significance.

Categories and Subject Descriptors: G.1.2 [**Numerical Analysis**]: Approximation—*minimax approximation and algorithms*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*trees*; H.1.1 [**Models and Principles**]: Systems and Information Theory—*value of information*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*graph and tree search strategies*

General Terms: Algorithms, Measurement, Performance, Theory

Additional Key Words and Phrases: Decision analysis, game playing, pathology

1. *Introduction*

Trees are used in nearly every area of computer science. However, many of their properties are not well understood, and this paper is concerned with some of them. Specifically, the relationships between depth of tree search and quality of behavior in decision making are studied.

Problem reduction is widely used in artificial intelligence and in several other areas of computer science. Problem reduction trees and game trees are basically the same, and using the problem reduction approach may be viewed as formulating the problem to be solved as a decision problem on a game tree. Decisions are made by searching to the end of the tree to predict the results of each possible decision.

Most of the material in this paper is adapted from the author's Ph.D. dissertation, which was completed at Duke University in August 1979. This work was supported in part by a National Science Foundation graduate fellowship, in part by a James B. Duke graduate fellowship, and in part by National Science Foundation Grant ENG-7822159 to the Laboratory for Pattern Analysis at the University of Maryland.

Author's address: Computer Science Department, University of Maryland, College Park, MD 20742.

Since the number of nodes in a tree generally increases exponentially with the depth of the tree, it is usually infeasible to do a complete search of a large game tree. Techniques such as the alpha–beta search procedure [2, 5, 9, 11, 17, 18, 26], which guarantee correct results without looking at every node in the tree, have been developed. However, such procedures still require a complete search of some portions of the tree. Game trees are often so large that this is physically impossible.

In game-playing computer programs, good results have been obtained by searching the tree only to some severely limited depth, using a *static evaluation function* to estimate the utility values of the nodes at that depth, and then proceeding, in the usual manner,[1] to compute utility values for the shallower nodes in the tree as if the estimated utility values were in fact correct [3, 6, 7, 9, 18, 22, 23, 25, 28]. This technique, which we call *heuristic game tree searching*, is used implicitly in "real life" decision situations, although it does not seem to have been studied by decision analysts.

There is almost universal agreement that when a heuristic game tree search is used, increasing the depth of the search improves the quality of the decision. This has been dramatically demonstrated with recent game-playing computer programs [3, 22, 28, 29], but such results are purely empirical. The author knows of no previous theoretical investigation of the effects of search depth on the quality of a decision.

This paper is concerned with a mathematical theory modeling the effects of searching deeper on the probability that a decision is correct. In this theory, the errors made by the evaluation function are modeled as independent, identically distributed random values superimposed on the true values of the nodes evaluated. The major result of this study is that contrary to the conventional belief, there is a large class of game trees and evaluation functions such that as long as the search does not reach the end of the tree (in which case a correct decision could be guaranteed), deeper search will not increase the probability that a decision is correct, *but will instead cause the decision to become increasingly random.*

As an example, Figure 1 illustrates how the probability of correct decision varies on a game tree $G(1, 1)$ which is discussed later. On $G(1, 1)$, the probability of making a correct choice of move is $\frac{1}{2}$ if the moves are chosen at random. The behavior of evaluation functions with five different error distributions is shown. At shallow search depths, some of them give very good results, and some give very poor results. But as the search depth increases, the probability of correct decision converges to $\frac{1}{2}$ in each case. This figure is discussed further in Section 6.1.

This idea may seem counterintuitive. One might suppose that since searching deeper gives more information, it cannot possibly make the decision worse. But the reader should keep in mind that the evaluation function merely *estimates* the quality of game tree positions, and these estimates may often be in error. There are some game trees for which searching deeper causes the real information to be lost in errors and noise.

Section 2 is a presentation of the mathematical model on which these results are based. Section 3 is a description of an infinite class of game trees having a regular enough structure that nontrivial theorems can be proved about them. Section 4 contains a number of preliminary theorems which are used in Section 5 to prove the central mathematical results of this paper. Relevant experimental results are summarized in Section 6, and concluding remarks appear in Section 7. A glossary of terms and symbols appears at the end of the paper.

[1] In computer game playing, "minimaxing" is normally used. This corresponds to the maximin decision criterion of decision analysis [10, 31].

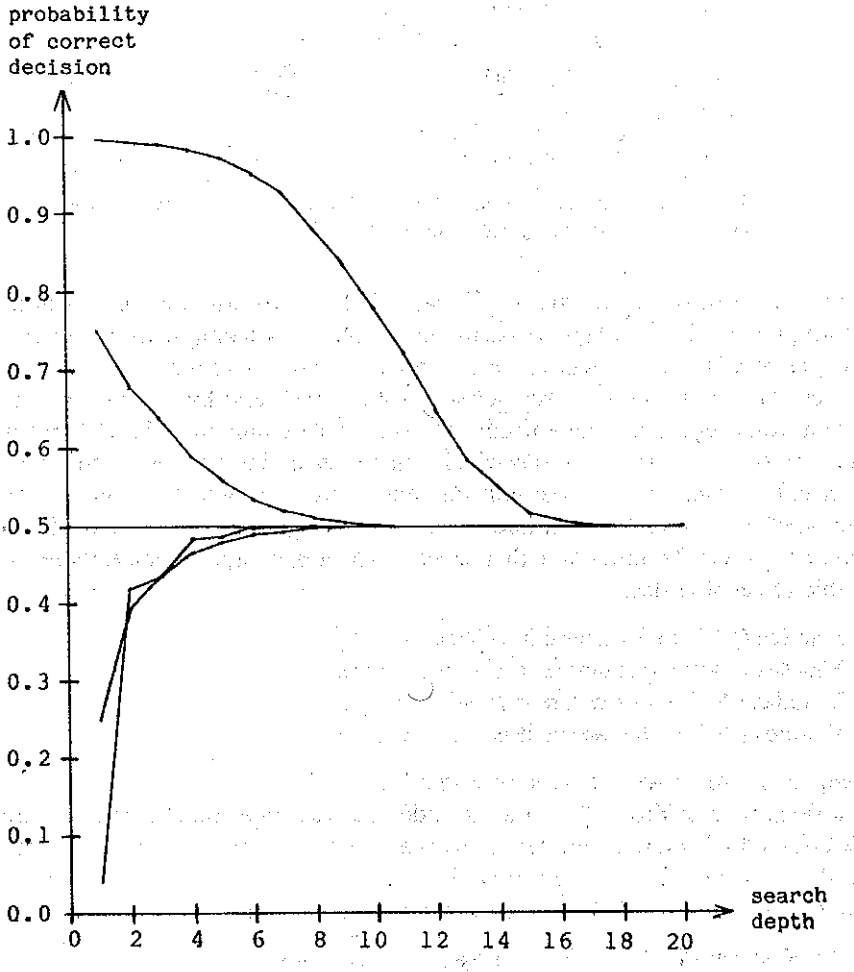

FIG. 1. Probability of correct decision as a function of search depth on the game tree $G(1, 1)$, for five different evaluation functions. On $G(1, 1)$, the probability of correct decision is 0.5 if the choice is made at random. For each of the five functions, this value is approached as the search depth increases.

2. *A Mathematical Model*

This section is a discussion of the mathematical model used in this paper. It covers games and game trees, minimaxing, a stochastic model of evaluation function errors, and how the probability that a decision is correct depends on these errors.

2.1. GAMES AND GAME TREES. By the word "game" we mean a zero-sum, perfect-information game in which the play strictly alternates between two players, and in which the outcome is not determined even partially by chance (as would occur, for example, in dice games). On his move, each player is allowed only a finite number of moves among which to choose, but we do not require that every game end in a finite number of moves.

We also stipulate that no game can have draws. Although this restriction simplifies the mathematics in this paper, it can easily be removed. Any game G containing draws can easily be transformed into two games G' and G'' without draws, by mapping draws into wins and losses, respectively. The properties of G are easily

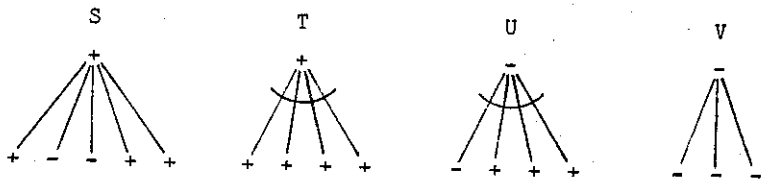

FIG. 2. Examples of S, T, U, and V nodes. Nodes where it is Min's move are indicated by the arcs drawn beneath them.

determined from the properties of G' and G''. However, the mathematical model is now simpler, for if a finite game has no draws, then it is a simple inductive proof that every position is a forced win for either one player or the other.

Given these restrictions, every game can be represented by a *game tree* in which the root node represents the current position in the game and the children of each node represent the game positions which can be reached in one move from that node. Let us call the two players Max and Min, and assign the signs "+" and "−" to nodes which are forced wins for Max and Min, respectively. If the game tree is finite, then it can be proved by induction that every node has a sign, whence there are four possible types of nodes:

(1) S nodes ("+" nodes where it is Max's move);
(2) T nodes ("+" nodes where it is Min's move);
(3) U nodes ("−" nodes where it is Min's move);
(4) V nodes ("−" nodes where it is Max's move).

Examples of the nodes are given in Figure 2.

As illustrated in Figure 2, S and U nodes are the only nodes at which it makes a difference which move is chosen. Thus these nodes are called *critical* nodes, and T and V nodes are called *noncritical* nodes.

Since the play strictly alternates between Max and Min,

(1) the children of an S node are always T and U nodes;
(2) the children of a T node are always S nodes;
(3) the children of a U node are always V and S nodes;
(4) the children of a V node are always U nodes.

These and other properties of the four types of nodes are summarized in Table I.

On an infinite game tree, not every node need be a forced win or forced loss. However, the "+" and "−" labeling (and thus the four types S, T, U, and V) may be extended to every node in the tree by assigning "+" and "−" labels in any way such that the following four conditions hold:

(1) For "+" nodes where it is Max's move, at least one child is a "+" node.
(2) For "+" nodes where it is Min's move, all children are "+" nodes.
(3) For "−" nodes where it is Min's move, at least one child is a "−" node.
(4) For "−" nodes where it is Max's move, all children are "−" nodes.

2.2 CHOOSING A MOVE. The use of static evaluation functions and minimaxing to choose moves in a game was first proposed by Shannon [24] and is now well known. An evaluation function associates with each game position a number indicating how good the position is estimated to be. High numbers mean good positions for Max (and therefore bad ones for Min); low numbers mean the reverse. If the function is to be implemented on a computer, then its range of possible values must

TABLE I. CHARACTERISTICS OF GAME TREE NODES

Type	S	T	U	V
Sign	"+"	"+"	"−"	"−"
Player to move	Max	Min	Min	Max
Critical node	Yes	No	Yes	No
Types of successor nodes	T,U	S	V,S	U

be finite. We take this range to be the finite set of integers $\{0, 1, \ldots, r\}$ for some integer r.

If e is an evaluation function for a game tree G and g is a node of G, then the *depth d minimax value* for g using e is defined as

$$e_d(g) = \begin{cases} e(g) & \text{if } d = 0 \text{ or } g \text{ is a terminal node,} \\ \max\{e_{d-1}(g') \mid g' \text{ is a child of } g\} & \\ \qquad \text{if } d > 0 \text{ and it is Max's move at } g, & \\ \min\{e_{d-1}(g') \mid g' \text{ is a child of } g\} & \\ \qquad \text{if } d > 0 \text{ and it is Min's move at } g. & \end{cases} \tag{2.1}$$

For each integer $d > 0$, choosing a move using a depth d minimax search consists of computing the depth $d-1$ minimax values for each child of the current node and moving to the child having the best value, that is, the largest value if it is Max's move, or the smallest value if it is Min's move. If several nodes share this same value, the choice is made at random among them. If $d = 0$, then the choice is made at random among all possible moves.

2.3. A STOCHASTIC MODEL OF EVALUATION FUNCTIONS. By choosing either accurate or inaccurate evaluations for nodes far down in a game tree, evaluation functions can be constructed whose performance either improves or degrades at increasing search depths. Thus no universal statements can be made about the quality of evaluation functions as a function of search depth. However, a mathematical model can be developed which makes probabilistic rather than universal predictions.

Ideally, an evaluation function would return 0 on all "−" nodes and r on all "+" nodes. Such behavior would result in perfect play. However, the estimations made by an evaluation function are usually somewhat (and sometimes drastically) in error [28, 29]. We model the error as a random value added to or subtracted from the ideal value. Increasing the amount of error made by an evaluation function e on some node g means increasing $e(g)$ from the ideal value of 0 if g is a "−" node, or decreasing $e(g)$ from the ideal value of r if g is a "+" node. Thus we define the error made by e at g to be

$$\text{err}_e(g) = \begin{cases} e(g) - 0 & \text{if } g \text{ is a "−" node,} \\ r - e(g) & \text{if } g \text{ is a "+" node.} \end{cases} \tag{2.2}$$

We assume that the errors $\{\text{err}_e(g) \mid g \text{ is a node}\}$ are independent, identically distributed random variables with probability function (p.f.)

$$f_{\text{err}_e}(i) = \Pr[\text{err}_e(g) = i \mid g \text{ is a node}]. \tag{2.3}$$

This assumption has the advantage of making the problem mathematically tractable. How well it corresponds to reality is discussed in Section 7.

Because of the definition of $\text{err}_e(g)$, eq. (2.3) is equivalent to assuming that the p.f. $f_e^+(i)$ for the values returned by e on "+" nodes is a mirror image of the p.f.

$f_e^-(i)$ for the values returned by e on "−" nodes. In particular,

$$\begin{aligned} f_e^+(i) &= \Pr[e(g) = i \mid g \text{ is a "+" node}] \\ &= \Pr[\mathrm{err}_e(g) = r - i \mid g \text{ is a "+" node}] && \text{(from eq. (2.2))} \\ &= \Pr[\mathrm{err}_e(g) = r - i] && \text{(since } \{\mathrm{err}_e(g)\} \text{ are identically distributed)} \\ &= f_{\mathrm{err}_e}(r - i) && \text{(from eq. (2.3))} \end{aligned}$$

and

$$\begin{aligned} f_e^-(i) &= \Pr[e(g) = i \mid g \text{ is a "−" node}] \\ &= \Pr[\mathrm{err}_e(g) = i \mid g \text{ is "−" node}] && \text{(from eq. (2.2))} \\ &= \Pr[\mathrm{err}_e(g) = i] && \text{(since } \{\mathrm{err}_e(g)\} \text{ are identically distributed)} \\ &= f_{\mathrm{err}_e}(i) && \text{(from eq. (2.3)),} \end{aligned}$$

whence the p.f. for the values e returns on an arbitrary node g is

$$f_{e(g)}(i) = \begin{cases} f_e^+(i) & \text{if } g \text{ is a "+" node,} \\ f_e^-(i) = f_e^+(r - i) & \text{if } g \text{ is a "−" node.} \end{cases} \tag{2.4}$$

Thus the values that e returns are completely characterized by f_e^+ and r.

Let h and h' be the smallest and largest numbers e returns with nonzero probability on nonterminal "+" nodes of a game tree. If $h = r - h'$, then we say that e is *symmetrically bounded* by h.

From eq. (2.4) we get

$$\Pr[e(g) = i \mid g \text{ is a "−" node}] = \Pr[e(g) = r - i \mid g \text{ is a "+" node}], \tag{2.5}$$

which means that if e is symmetrically bounded by h, then h and $h' = r - h$ are also the smallest and largest numbers e returns with nonzero probability on nonterminal "−" nodes. This means that the highest possible value for $e(g)$ may occur not only on "+" nodes but also on "−" nodes, and the lowest possible value for $e(g)$ may occur not only on "−" nodes but also on "+" nodes. This is not an entirely unreasonable occurrence, since the evaluation functions used in chess-playing computer programs can typically be made to return very high values for very dire positions if the amount of material on the board is made high enough [30].

3. *"Homogeneous" Games*

3.1. The Basic Definition. If G is a game tree, then the only nodes of G for which it matters which move is chosen are the critical nodes, that is, the S and U nodes. For such nodes we call the choice *correct* if the child chosen is a forced win node for the player making the choice. At a critical node g, the probability that a correct decision is made at g is determined by the evaluation function e, the depth d of the minimax search, and the subtree of G rooted at g.

Since the structure of the subtree rooted at G may vary markedly depending on G and g, it is very difficult to make any general statements about how the probability of correct decision varies with d. We now consider a class of game trees which have a regular enough structure that such statements *can* be made.

Definition 3.1. Let $m \geq 1$ and $n \geq 1$ be integers, and let x be one of the letters S, T, U, and V. Then $G_x(m, n)$ is the unique infinite game tree such that

(1) the root of $G_x(m, n)$ is an x node;

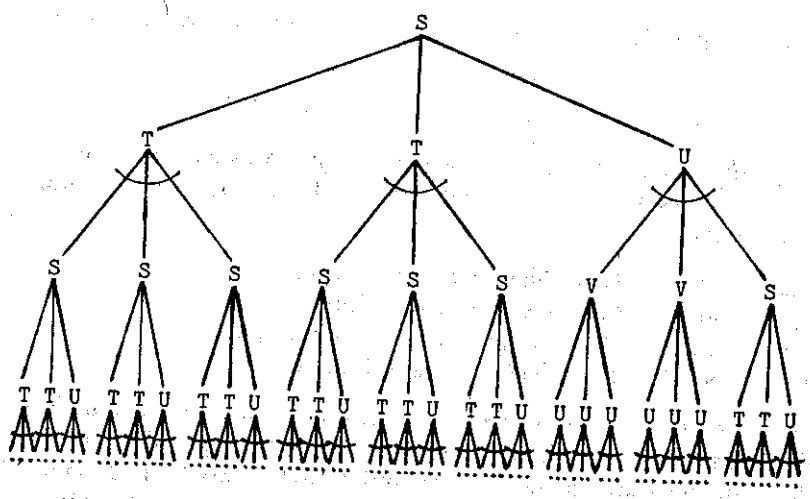

FIG. 3. The tree $G_S(2, 1)$.

(2) every critical node in $G_x(m, n)$ has m children of the same sign, followed by n children of opposite sign;
(3) every node of $G_x(m, n)$ has $m + n$ children.

$G_S(2, 1)$ is illustrated in Figure 3.

We let $G(m, n)$ denote any one of $G_S(m, n)$, $G_T(m, n)$, $G_U(m, n)$, and $G_V(m, n)$. The relevance of the properties of $G(m, n)$ to finite games is that infinitely many finite games may be formed from $G(m, n)$ simply by turning selected nodes into terminal nodes. If G' is such a truncation of $G(m, n)$, and if the subtree of G' having g as its root is complete to at least depth d, then a depth d minimax search from g in G' and a depth d minimax search from g in $G(m, n)$ will yield exactly the same results. This allows us to prove results about $G(m, n)$ with the assurance that they also apply to truncations of $G(m, n)$.

3.2. INDEPENDENCE RESULTS. Let e be a symmetrically bounded evaluation function for $G(m, n)$. Then several independence conditions hold, as discussed below.

3.2.1. *Evaluation Function Values on Distinct Nodes.* Let g be any node of $G(m, n)$, and let $X = \{x_1, x_2, \ldots, x_k\}$ be the set of all depth d descendants of g, in order from left to right. Let g' be distinct from g and from the members of X. If type(g) is known, then type(x_i) is determined for every $x \in X$ (for example, if x is an S node and $d = 1$, then $x_1, \ldots, x_m$ are T nodes and $x_{m+1}, \ldots, x_k$ are U nodes). But $e(x_i)$, given type(x_i), is independent of $e(g')$;[2] that is, for all j,

$$\Pr[e(x_i) = j \mid \text{type}(x_i)] = \Pr[e(x_i) = j \mid \text{type}(x_i), e(g')]. \tag{3.1}$$

But $e_d(g)$ is a function of $\{e(x_i) \mid x_i \in X\}$. Thus $e_d(g)$, given type(g), is a function of random variables which are independent of $e(g')$, and thus $e(g)$, given type(g), is independent of $e(g')$; that is,

$$\Pr[e_d(g) = j \mid \text{type}(g)] = \Pr[e_d(g) = j \mid \text{type}(g), e_d(g')]. \tag{3.2}$$

3.2.2. *Nodes of Known Type.* Suppose that type(g) = type (g'). Then the subtrees

[2] Note however, that if type(x_i) is not known, then $e(x_i)$ is not necessarily independent of $e(g')$; i.e., it is not necessarily true that $\Pr[e(x_i) = j] = \Pr[e(x_i) = j \mid e(g')]$.

rooted at g and g' have exactly the same structure, whence (by means of a simple inductive proof),

$$f_{e_d(g)}(i) = f_{e_d(g')}(i) \qquad \text{for all} \quad i, \tag{3.3}$$

where $f_{e_d(g)}$ and $f_{e_d(g')}$ are the p.f.s for $e_d(g)$ and $e_d(g')$, respectively. Thus $f_{e_d(g)}$ is the same for all S nodes g of $G(m, n)$, and similarly for the T, U, and V nodes. This allows us to define

(1) $s_d = f_{e_d(g)}$ and $S_d = F_{e_d(g)}$, where g is any S node;[3]
(2) $t_d = f_{e_d(g)}$ and $T_d = F_{e_d(g)}$, where g is any T node;
(3) $u_d = f_{e_d(g)}$ and $U_d = F_{e_d(g)}$, where g is any U node; and
(4) $v_d = f_{e_d(g)}$ and $V_d = F_{e_d(g)}$, where g is any V node.

Suppose that either g is an S node and g' is a U node, or g is a T node and g' is a V node. Then it is easy to see that the subtrees rooted at g and g' are isomorphic under the mapping which maps S, T, U, and V nodes into U, V, S, and T nodes, respectively. Thus if $\{x_1, \ldots, x_k\}$ and $\{x'_1, \ldots, x'_k\}$ are the depth d children of g and g', respectively, and if $x_i \in X$ is an S, T, U, or V node, then x'_i is a U, V, S, or T node, respectively. But since S and T nodes are "+" nodes and U and V nodes are "−" nodes, this means that one of $\{x_i, x'_i\}$ is a "+" node and the other is a "−" node. Thus from eq. (2.4) the p.f. for $e(x_i)$ is a mirror image of the p.f. for $e(x'_i)$. It follows by induction that the p.f. $f_{e_d(g)}$ is a mirror image of the p.f. $f_{e_d(g')}$; that is,

$$s_d(i) = u_d(r - i) \qquad \text{and} \qquad t_d(i) = v_d(r - i). \tag{3.4}$$

3.2.3. *The Probability of Correct Decision.* The only nodes at which it makes a difference what move is made are the critical nodes, that is, the S and V nodes. If g is an S node, then it is Max's move at g, and g has m "+" and n "−" children. If $d = 0$, then Max is not searching at all, so each choice of move is equally likely. The move is correct only if a "+" node is chosen, so then the *probability of correct decision* is

$$D(G(m, n), g, e, d) = \frac{m}{m + n} \qquad \text{if} \quad d = 0. \tag{3.5}$$

The same equation holds if g is a V node.

Suppose $d \geq 1$, and let

$$H = \begin{cases} \max\{e_{d-1}(g') \mid g' \text{ is a child of } g\} & \text{if} \quad g \text{ is a Max node,} \\ \min\{e_{d-1}(g') \mid g' \text{ is a child of } g\} & \text{if} \quad g \text{ is a Min node.} \end{cases}$$

If Max (or Min) uses a depth d minimax search, then he will choose any child g' of g such that $e_{d-1}(g') = H$. Suppose I "+" nodes and J "−" nodes get this value. Since it is equally likely that any one of them will be chosen, the probability of correct decision is $I/(I + J)$. But H, I, and J vary randomly from 0 to r, 0 to m, and 0 to n, respectively. Thus the probability of correct decision is

$$D(G(m, n), g, e, d) = \sum_{i=0}^{m} \sum_{j=0}^{n} \frac{i}{i + j} \sum_{k=0}^{n} \Pr[I = i, J = j, H = k] \qquad \text{if} \quad d > 0. \tag{3.6}$$

If g is an S node, then all "+" children of g are T nodes and all "−" children of g are U nodes. The event $\{I = i, J = j, H = k\}$ occurs if e_{d-1} returns the value k on

[3] If X is a random variable with p.f. $f(x)$, then $F(x) = \sum_{i \leq x} f(i)$ is the cumulative distribution function (c.d.f.) for X. Thus $f_1 = f_2$ if and only if $F_1 = F_2$.

TABLE II. APPROXIMATE VALUES OF $w(n)$, FOR $n = 1, 2, \ldots, 50$

n	$w(n)$	n	$w(n)$	n	$w(n)$	n	$w(n)$	n	$w(n)$
1	0.50000	11	0.15560	21	0.10271	31	0.07872	41	0.06463
2	0.38197	12	0.14745	22	0.09955	32	0.07700	42	0.06352
3	0.31767	13	0.14024	23	0.09662	33	0.07536	43	0.06246
4	0.27551	14	0.13382	24	0.09387	34	0.07380	44	0.06144
5	0.24512	15	0.12805	25	0.09130	35	0.07231	45	0.06045
6	0.22191	16	0.12283	26	0.08889	36	0.07088	46	0.05950
7	0.20346	17	0.11809	27	0.08662	37	0.06952	47	0.05858
8	0.18835	18	0.11375	28	0.08448	38	0.06822	48	0.05770
9	0.17570	19	0.10977	29	0.08245	39	0.06697	49	0.05684
10	0.16492	20	0.10610	30	0.08054	40	0.06577	50	0.05601

i of the T children and j of the U children, and values less than k on $m - i$ of the T children and $n - j$ of the U children. Since the types of the nodes are known, eq. 3.2 states that the returned values are independent. Thus

$$\begin{aligned}
&\Pr[I = i, J = j, H = k] \\
&\quad = \binom{m}{i} (\Pr[e_{d-1}(g') = k \mid g \text{ is a } T \text{ node}])^i (\Pr[e_{d-1}(g') < k \mid g \text{ is a } T \text{ node}])^{m-i} \\
&\quad\quad \cdot \binom{n}{j} (\Pr[e_{d-1}(g') = k \mid g \text{ is a } U \text{ node}])^j (\Pr[e_{d-1}(g') < k \mid g \text{ is a } U \text{ node}])^{n-j} \\
&\quad = \binom{m}{i} (t_{d-1}(k))^i (T_{d-1}(k-1))^{m-i} \binom{n}{j} (u_{d-1}(k))^j (U_{d-1}(k-1))^{n-j}. \qquad (3.7)
\end{aligned}$$

Note that this (and hence $D(G(m, n), g, e, d)$) is a continuous function of $\{t_{d-1}(k) \mid k = 0, 1, \ldots, r\}$ and $\{u_{d-1}(k) \mid k = 0, 1, \ldots, r\}$ which is independent of which S node g is.

Similarly, if g is a V node, then $\Pr[I = i, J = j, H = k]$ (and hence $D(G(m, n), g, e, d)$) is a continuous function of $\{v_{d-1}(k) \mid k = 0, 1, \ldots, r\}$ and $\{s_{d-1}(k) \mid k = 0, 1, \ldots, r\}$ which is independent of which T node g is.[4]

4. *Preliminary Theorems*

Section 5 contains the central theorems of this paper. These results require several preliminary results which are presented here.

THEOREM 4.1. *For every $n > 0$, the equation $(1 - (1 - x)^n)^n = x$ has exactly one root $w(n)$ in the interval $(0, 1)$. $w(n)$ converges monotonically to 0 as n increases.*

THEOREM 4.2. *Let $z_0 \in [0, 1]$, and let $n > 1$. For every integer $i \geq 0$, let $z_{i+1} = (1 - (1 - z_i)^n)^n$. Then*

(1) *if $z_0 < w(n)$, then $\lim_{i \to \infty} z_i = 0$;*
(2) *if $z_0 > w(n)$, then $\lim_{i \to \infty} z_i = 1$;*
(3) *if $z_0 = w(n)$, then $z_i = w(i)$ for all i.*

The function $w(n)$ and several of its properties were independently discovered by Baudet [2], Pearl [19], and the author [13, 14]. Proofs of Theorems 4.1 and 4.2 appear in the appendix to [14]. Approximate values of $w(n)$ appear in Table II.

[4] It can further be shown [13] that the value of $D(G(m, n), g, e, d)$ is independent of whether g is an S node or a V node. However, this result is not needed in the present paper.

We define

$$x_{i+1} = (1 - (1 - x_i)^m(1 - y_i)^n)^{m+n} \tag{4.1}$$

and

$$y_{i+1} = (1 - (1 - y_i)^{m+n})^m(1 - (1 - x_i)^m(1 - y_i)^n)^n, \tag{4.2}$$

where $m \geq 1$ and $n \geq 1$ are real, $i \geq 0$ is an integer, and $x_0, y_0 \in [0, 1]$.

THEOREM 4.3. *The following properties hold:*

(1) $x_i, y_i \in [0, 1]$ *for all* i.
(2) *If* $y_0 = 0$, *then* $y_i = 0$ *for all* i.
(3) *If* $x_0 = y_0 = 0$, *then* $x_i = y_i = 0$ *for all* i.
(4) *If* $y_0 = 1$, *then* $x_i = y_i = 1$ *for all* $i \geq 1$.

PROOF. Immediate, by induction on i. □

THEOREM 4.4. *If* $x_0 > w(m + n)$ *and* $y_0 > w(m + n)$, *then*

$$\lim_{i\to\infty} x_i = \lim_{i\to\infty} y_i = 1.$$

PROOF. For integer $i \geq 0$, let $z_{i+1} = (1 - (1 - z_i)^{m+n})^{m+n}$, where $z_0 = \min(x_0, y_0)$. Then $w(m + n) < z_0 < 1$, so from Theorem 4.2, $\lim_{i\to\infty} z_i = 1$.

We now prove by induction that for every integer $i \geq 0$,

$$z_i \leq x_i \leq 1 \qquad \text{and} \qquad z_i \leq y_i \leq 1.$$

This statement holds for $i = 0$, because $z_0 = \min(x_0, y_0)$. Suppose it holds for $i = k$. Then

$$\begin{aligned} x_{k+1} &= (1 - (1 - x_k)^m(1 - y_k)^n)^{m+n} \\ &\geq (1 - (1 - z_k)^m(1 - z_k)^n)^{m+n} = z_{k+1}, \end{aligned}$$

and

$$\begin{aligned} y_{k+1} &= (1 - (1 - y_k)^{m+n})^m(1 - (1 - x_k)^m(1 - y_k)^n)^n \\ &\geq (1 - (1 - z_k)^{m+n})^m(1 - (1 - z_k)^m(1 - z_k)^n)^n = z_{k+1}, \end{aligned}$$

and from Theorem 4.3, $x_{k+1} \leq 1$ and $y_{k+1} \leq 1$. Thus the statement holds for $i = k + 1$.

From the above, it follows that

$$\lim_{i\to\infty} x_i = \lim_{i\to\infty} y_i = 1. \qquad \square$$

COROLLARY 4.4.1. *Let* $x_0 > 0$ *and* $y_0 > 0$. *Then there is an* L *such that if* $m + n > L$,

$$\lim_{i\to\infty} x_i = \lim_{i\to\infty} y_i = 1. \tag{4.3}$$

PROOF. From Theorem 4.1, there is an L such that if $m + n > L$, then $w(m + n) < \min(x_0, y_0)$, whence eq. (4.3) follows immediately from Theorem 4.4. □

5. *The Central Results*

Let g be a critical node of $G(m, n)$, and suppose a depth d minimax search is done from g using a symetrically bounded evaluation function. For all but finitely many values of m and n, the deeper the search is, the more likely it is that every child of g

will receive exactly the same minimax value. But if this occurs, then all possible moves will be equally likely, regardless of whether the moves are good ones or bad ones. These statements are proved in this section.

If g is an S or T node, then g is a "+" node. Thus

$$s_0(i) = t_0(i) = f_{e(g)}(i) = f_e^+(i),$$

whence

$$S_0(i) = T_0(i) = \sum_{j=0}^{i} f_e^+(j) \in [0, 1]. \tag{5.1}$$

However, if g is a U or V node, then g is a "−" node. Thus

$$u_0(i) = v_0(i) = f_{e(g)}(i) = f_e^-(i),$$

whence

$$U_0(i) = V_0(i) = \sum_{j=0}^{i} f_e^-(j) = \sum_{j=r-i}^{r} f_e^+(j) \in [0, 1]. \tag{5.2}$$

If g is an S node, then it is Max's move at g; so from eq. (2.1),

$$e_{d+1}(g) = \max\{e_d(g') | g' \text{ is a child of } g\}.$$

Thus for each i, $e_{d+1}(g) \leq i$ if and only if for every child g' of g, $e_d(g') \leq i$. But m of the children of g are T nodes, and n of them are U nodes. Thus

$$S_{d+1}(i) = \Pr[e_{d+1}(g) \leq i] = (T_d(i))^m (U_d(i))^n. \tag{5.3}$$

If g is a T node, then it is Min's move at g; so from eq. (2.1),

$$e_{d+1}(g) = \min\{e_d(g') | g' \text{ is a child of } g\}.$$

Thus for each i, $e_{d+1}(g) > i$ if and only if for every child g' of g, $e_d(g') > i$. But all $m + n$ of the children of g are S nodes. Thus

$$1 - T_{d+1}(i) = \Pr[e_{d+1}(g) > i] = (1 - S_d(i))^{m+n}. \tag{5.4}$$

Similarly, we have

$$1 - U_{d+1}(i) = (1 - V_d(i))^m (1 - S_d(i))^n \tag{5.5}$$

and

$$V_{d+1}(i) = (U_d(i))^{m+n}. \tag{5.6}$$

Simple algebraic manipulation of eqs. (5.3)–(5.6) yields

$$\begin{aligned} S_{d+2}(i) &= (1 - (1 - V_d(i))^{m+n})^m (1 - (V_d(i))^m (S_d(i))^n)^n; \\ 1 - T_{d+2}(i) &= (1 - (T_d(i))^m (U_d(i))^n)^{m+n} \end{aligned} \tag{5.7}$$

$$\begin{aligned} &= (1 - (1 - [1 - T_d(i)])^m (1 - [1 - U_d(i)])^n)^{m+n}; \\ 1 - U_{d+2}(i) &= (1 - (U_d(i))^{m+n})^m (1 - (T_d(i))^m (U_d(i))^n)^n \end{aligned} \tag{5.8}$$

$$\begin{aligned} &= (1 - (1 - [1 - U_d(i)])^{m+n})^m \\ &\cdot (1 - (1 - [1 - T_d(i)])^m (1 - [1 - U_d(i)])^n)^n; \end{aligned} \tag{5.9}$$

$$V_{d+2}(i) = (1 - (1 - V_d(i))^m (1 - S_d(i))^n)^{m+n}. \tag{5.10}$$

Thus the pair of sequences $(\{1 - T_{2d}(i)\}_{d=0}^{\infty}, \{1 - U_{2d}(i)\}_{d=0}^{\infty})$ and the pair of sequences $(\{V_{2d}(i)\}_{d=0}^{\infty}, \{S_{2d}(i)\}_{d=0}^{\infty})$ are instances of the pair of sequences $(\{x_d(i)\}_{d=0}^{\infty}, \{y_d(i)\}_{d=0}^{\infty})$ described in Section 4.

THEOREM 5.1. *Let e be an evaluation function on $G(m, n)$ with range $\{0, 1, \ldots, r\}$, and let e be symmetrically bounded by h. If $\min(f_e(h), f_e(r-h)) > w(m+n)$, then for $i = 0, 1, \ldots, r$,*

$$\lim_{d\to\infty} v_{2d}(i) = \lim_{d\to\infty} s_{2d}(i) = \begin{cases} 1 & \text{if } i = h, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\lim_{d\to\infty} t_{2d}(i) = \lim_{d\to\infty} u_{2d}(i) = \begin{cases} 1 & \text{if } i = r-h, \\ 0 & \text{otherwise.} \end{cases}$$

PROOF. Suppose $w(m+n) < \min(f_e(h), f_e(r-h))$. Then from eq. (5.1) and (5.2), if $i \geq h$, then

$$V_0(i) = \sum_{j=r-h}^{r} f_e^+(j) \geq f_e^+(r-h) > w(m+n)$$

and

$$S_0(i) = \sum_{j=0}^{i} f_e^+(j) \geq f_e^+(h) > w(m+n).$$

Thus from Theorem 4.4 and eqs. (5.7) and (5.10),

$$\lim_{d\to\infty} V_{2d}(i) = \lim_{d\to\infty} S_{2d}(i) = 1.$$

But since h is a symmetric bound for e, if $i < h$, then

$$V_0(i) = \sum_{j=0}^{i} f_e^-(j) = 0 \quad \text{and} \quad S_0(i) = \sum_{j=0}^{i} f_e^+(j) = 0,$$

so from Theorem 4.3 and eq. (5.7) and (5.10),

$$\lim_{d\to\infty} V_{2d}(i) = \lim_{d\to\infty} S_{2d}(i) = 0.$$

Thus for $i = 0, 1, \ldots, r$,

$$\lim_{d\to\infty} v_{2d}(i) = \lim_{d\to\infty} V_{2d}(i) - \lim_{d\to\infty} V_{2d}(i-1) = \begin{cases} 1 & \text{if } i = h, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\lim_{d\to\infty} s_{2d}(i) = \lim_{d\to\infty} S_{2d}(i) - \lim_{d\to\infty} S_{2d}(i-1) = \begin{cases} 1 & \text{if } i = h, \\ 0 & \text{otherwise.} \end{cases}$$

The proof for t_{2d} and u_{2d} is similar. □

COROLLARY 5.1.1. *Under the assumptions of Theorem* 5.1,

$$\lim_{d\to\infty} v_{2d+1}(i) = \lim_{d\to\infty} s_{2d+1}(i) = \begin{cases} 1 & \text{if } i = r-h, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\lim_{d\to\infty} t_{2d+1}(i) = \lim_{d\to\infty} u_{2d+1}(i) = \begin{cases} 1 & \text{if } i = h, \\ 0 & \text{otherwise.} \end{cases}$$

PROOF. Immediate from Theorem 5.1 and eqs. (5.3)–(5.6). □

THEOREM 5.2. *Under the assumptions of Theorem* 5.1,

$$\lim_{d\to\infty} D_{m,n}(e, d) = \frac{m}{m+n}.$$

PROOF. Let g be an S node, and let H, I, and J be the same as in eq. (3.6). If for some d and h we had

$$t_{d-1}(j) = \begin{cases} 1 & \text{if } j = h, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$u_{d-1}(j) = \begin{cases} 1 & \text{if } j = h, \\ 0 & \text{otherwise,} \end{cases}$$

then all of the children of g would receive the same minimax value h, whence we would have

$$\Pr[I = i, J = j, H = k] = \begin{cases} 1 & \text{if } i = m, \quad j = n, \quad \text{and } k = h, \\ 0 & \text{otherwise,} \end{cases}$$

whence $D_{m,n}(e, d) = m/(m + n)$. But from Section 3.2.4, $D(G(m, n), g, e, d)$ is a continuous function of $\{t_{d-1}(i) \mid i = 0, 1, \ldots, r\}$ and $\{u_{d-1}(i) \mid i = 0, 1, \ldots, r\}$. Thus from Theorem 5.1 and Corollary 5.1.1,

$$\lim_{d \to \infty} D_{m,n}(e, d) = \frac{m}{m + n}.$$

The same result is obtained in a similar manner if g is a U node. □

To summarize the results of Section 5, suppose a player is choosing a move at a critical node of $G(m, n)$, using an evaluation function e which is symmetrically bounded by h. For all but finitely many of the $G(m, n)$, $w(m + n) < \min(f_e(h), f_e(r - h))$, whence Theorems 5.1 and 5.2 apply. Theorem 5.1 states that as the search depth increases, it becomes increasingly likely that the moves among which the player is choosing will all have the same minimax value. But if this occurs, a move must be chosen at random, and each possible choice will be equally likely. Thus, as stated in Theorem 5.2, as the search depth increases the probability of correct decision converges to $m/(m + n)$. When these events occur, we say that $G(m, n)$ is *e-pathological*. Thus we have

COROLLARY 5.2.1. *For every symmetrically bounded evaluation function e, all but finitely many of the G(m, n) are e-pathological.*

An example of pathological behavior on the game tree $G(1, 1)$ is given in Figure 1. This example is further discussed in Section 6.1.

Figure 4 summarizes the theorems presented in this section. In this figure, every tree $G(m, n)$ is represented by the pair of integers (m, n). The diagonal line consists of the smallest values of m and n such that $\min(f_e(h), f_e(r - h) \geq w(m + n)$, where e is an evaluation function symmetrically bounded by h. Area 1 is the area in which pathology is predicted by the theorems, and area 2—the finite area—is an area in which game trees may be either pathological or nonpathological. The occurrence of pathology in area 2 has been investigated experimentally. A few of the results of this experimentation are discussed in Section 6.

6. *Experimental Results*

Theorems 5.2 and 5.3 have been verified in numerous experimental tests. Indeed, such experiments were responsible for the mathematical insights which led to the proofs of the theorems. A few of the experiments are now discussed.

6.1. EXAMPLES OF PATHOLOGICAL BEHAVIOR. Suppose Max is choosing a move at a critical node of $G(m, n)$, using a symetrically bounded evaluation function e.

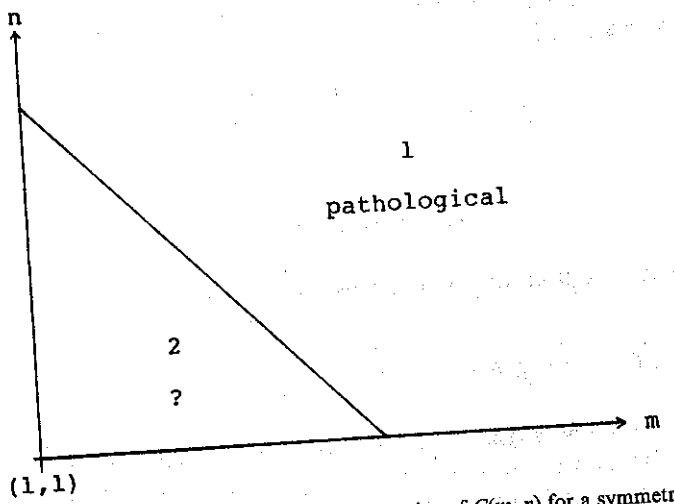

FIG. 4. Pathological and nonpathological behavior of $G(m, n)$ for a symmetrically bounded evaluation function e, as a function of m and n. The numbered areas are for reference in the text.

Then it must be that Max is at an S node and is choosing among T and U nodes. For a depth d minimax search, the p.f.s. for the minimax values of these nodes are T_{d-1} and U_{d-1}. Pathology occurs when there is an i such that

$$\lim_{d\to\infty} t_{2d}(j) = \lim_{d\to\infty} u_{2d}(j) = \lim_{d\to\infty} t_{2d-1}(r-j)$$
$$= \lim_{d\to\infty} u_{2d-1}(r-j) = \begin{cases} 1 & \text{if } j = i, \\ 0 & \text{otherwise.} \end{cases}$$

Table III illustrates this on the game tree $G(1, 1)$. This table gives the p.f.s for the minimax values and the probability of correct decision at various search depths, using the evaluation function e_2 of Table IV.

In Figure 1, the probability of correct decision on $G(1, 1)$ is graphed as a function of d for five different evaluation functions. From top to bottom, the curves are for e_1, e_2, e_3, e_4, and e_5, as defined in Table IV. When pathology occurs on other game trees $G(m, n)$, the behavior is much the same, except that the limiting value is $m/(m + n)$ rather than $\frac{1}{2}$.

6.2. WHICH TREES ARE PATHOLOGICAL? The results in Section 5 predict pathology on all games trees $G(m, n)$ such that $m + n$ is greater than some value L which depends on the error distribution of the evaluation function being used, but they say nothing about whether pathology occurs for $m + n \le L$. Pathology occurs on many of these trees as well. For example, Table V indicates for each m and n whether $G(m, n)$ is e_6-pathological, where e_6 is as defined in Table IV. In this table, the entry "A" indicates that

$$\lim_{d\to\infty} t_{2d}(i) = \lim_{d\to\infty} u_{2d-1}(r-i) = \begin{cases} 1 & \text{if } i = 3, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\lim_{d\to\infty} u_{2d}(i) = \lim_{d\to\infty} t_{2d-1}(r-i) = \begin{cases} 1 & \text{if } i = 2, \\ 0 & \text{otherwise;} \end{cases}$$

TABLE III. MINIMAX VALUES AND PROBABILITIES OF CORRECT DECISION ON $G(1, 1)$ AT VARIOUS SEARCH DEPTHS, USING THE EVALUATION FUNCTION e_2 OF TABLE IV

	$t_{d-1}(i)$ or $v_{d-1}(3-i)$				$u_{d-1}(i)$ or $s_{d-1}(3-i)$				
d	$i=0$	$i=1$	$i=2$	$i=3$	$i=0$	$i=1$	$i=2$	$i=3$	$D(e_2, d)$
1	0.10	0.20	0.30	0.40	0.40	0.30	0.20	0.10	0.750
3	0.08	0.30	0.41	0.21	0.19	0.40	0.32	0.09	0.639
5	0.03	0.37	0.52	0.08	0.05	0.45	0.45	0.05	0.560
7	0.00	0.36	0.63	0.01	0.00	0.40	0.59	0.01	0.522
9	0.00	0.27	0.73	0.00	0.00	0.28	0.72	0.00	0.507
11	0.00	0.14	0.86	0.00	0.00	0.15	0.85	0.00	0.502
13	0.00	0.04	0.96	0.00	0.00	0.04	0.96	0.00	0.500
15	0.00	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.500
2	0.19	0.32	0.33	0.16	0.46	0.33	0.17	0.04	0.689
4	0.17	0.48	0.32	0.04	0.28	0.50	0.21	0.02	0.593
6	0.09	0.66	0.25	0.00	0.12	0.68	0.20	0.00	0.535
8	0.02	0.82	0.16	0.00	0.03	0.83	0.14	0.00	0.510
10	0.00	0.92	0.08	0.00	0.00	0.92	0.08	0.00	0.502
12	0.00	0.98	0.02	0.00	0.00	0.98	0.02	0.00	0.500
14	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500

TABLE IV. P.F.S. FOR SIX EVALUATION FUNCTIONS

	$i=0$	$i=1$	$i=2$	$i=3$	$i=4$	Elsewhere
$f_{e_1}^{+}(i)$	1/781	5/781	25/781	125/781	625/781	0
$f_{e_2}^{+}(i)$	0.1	0.2	0.3	0.4	0	0
$f_{e_3}^{+}(i)$	0.2	0.2	0.2	0.2	0.2	0
$f_{e_4}^{+}(i)$	0.4	0.3	0.2	0.1	0	0
$f_{e_5}^{+}(i)$	625/781	125/781	25/781	5/781	1/781	0
$f_{e_6}^{+}(i)$	6/78	15/78	24/78	33/78	0	0

the entry "B" indicates that

$$\lim_{d\to\infty} t_{2d}(i) = \lim_{d\to\infty} u_{2d-1}(r-i) = \lim_{d\to\infty} u_{2d}(i)$$

$$= \lim_{d\to\infty} t_{2d-1}(r-i) = \begin{cases} 1 & \text{if } i=2, \\ 0 & \text{otherwise;} \end{cases}$$

and the entry "C" indicates that

$$\lim_{d\to\infty} t_{2d}(i) = \lim_{d\to\infty} U_{2d-1}(r-i) = \lim_{d\to\infty} u_{2d}(i)$$

$$= \lim_{d\to\infty} t_{2d-1}(r-i) = \begin{cases} 1 & \text{if } i=3, \\ 0 & \text{otherwise.} \end{cases}$$

Pathology occurs in the latter two cases.

The shape of the nonpathological region for e_6 is fairly typical. Not only does pathology occur when $m+n$ is large, but also when the ratio m/n is small. This corresponds well with our intuitions about what it is that evaluation functions measure.

An evaluation function is supposed to return a value indicating whether a game position is good or bad. In the game of chess, for example, good positions are often characterized in terms of such things as the number of pieces, their mobility, how

TABLE V. PATHOLOGY OF $G(m,n)$ AS A FUNCTION OF m AND n, FOR THE EVALUATION FUNCTION e_6 OF TABLE IV[a]

	n																			
m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*
2	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	C^*	C^*	C^*	C^*	C^*	C^*
3	A	B^*	B^*	B^*	B^*	B^*	B^*	B^*	B^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
4	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
5	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
6	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
7	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
8	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
9	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
10	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
11	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
12	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
13	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
14	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
15	A	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
16	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
17	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
18	A	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
19	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
20	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
21	A	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*
22	A	A	A	A	A	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*	C^*

[a] An asterisk indicates pathology. Its absence indicates that $D(e_6,d)$ converges to 1. Below the diagonal line is where Theorem 5.3 says that pathology must occur. The other information contained in this table is explained in the text.

well the king is protected, etc., but a good position is generally one from which a player has a variety of good moves, and a bad one is generally one from which a player has few (if any) good moves.

If m/n is small, such a characterization of a good position cannot be made. This is because every forced win node in $G(m, n)$ has many children which are forced loss nodes and only a few children which are forced win nodes, and vice versa. Thus the good positions look bad and the bad positions look good. Under such conditions, it is not so surprising that pathology would occur. This is discussed further in Section 7.3.2.

Additional experimental results [13] indicate that pathology may be more likely to occur when the evaluation function has a restricted range (so that fine distinctions among various nodes cannot be made), and provide information about the significance of the limits and rates of convergence of the minimax values.

7. *Discussion and Conclusions*

7.1. SUMMARY OF RESULTS. Games trees are a widely used model of situations in which a sequence of decisions must be made. In this model, decisions are made by searching the tree in an effort to predict the possible results of the decisions. On large game trees which cannot be searched completely, it is typical to search to a limited depth and use a static evaluation function to estimate the values of the nodes at that depth. When this is done, the conventional belief is "the deeper the search, the better the decision." We have examined this statement mathematically, to try to discover to what extent it is true.

The mathematical model used for this investigation has the following major features:

(1) Game tree nodes are classified as critical if it makes a difference what decision is made, and noncritical otherwise.
(2) The evaluation function is assumed to make independent, identically distributed random errors on all nodes, in such a way that the probability distribution for the values returned on forced win nodes is a mirror image of the probability distribution for the values returned on forced loss nodes.
(3) The highest and lowest values which the evaluation function returns with nonzero probability are returned with nonzero probability on both forced win and forced loss nodes.
(4) The use of the minimax propagation rule is assumed.
(5) The quality of a decision at a critical node is characterized as the probability that it is "correct," that is, the probability that the move is to a forced win child rather than a forced loss child.

There is an infinite class of game trees $\{G(m, n) \mid m$ and n are positive integers$\}$ which have a regular enough structure that the probability of correct decision on these trees can be analyzed mathematically. As was shown in Section 5, all but finitely many of these trees are *pathological* in the sense that as long as the search does not reach the end of the game tree (in which case a correct decision could be guaranteed), increasing the search depth does not improve the quality of the decision, but instead causes the choice to become more and more random. This occurs as described below.

Suppose a player is making a decision using a minimax search at a node for which it makes a difference what decision is made. If the search depth d is large and d is odd, then all children of g will be likely to have high minimax values. If d is large

and even, then all children of g will be likely to have low minimax values. Increasing the value of d makes it increasingly likely that all nodes receive exactly the *same* high or low value, whence the choice of move must be made purely at random.

This "manic-depressive" behavior has also been observed in computer programs which play chess, checkers, and other games [14, 28, 29], except that it does not get out of hand: even though all nodes tend to get high or low values, enough of a numerical distinction remains between the values of good and bad moves that a good decision can be made.

7.2. THE HORIZON EFFECT. If a minimax search extends all the way to the end of the game tree, pathology does not occur, since in this case perfect play can be guaranteed. Some readers have pointed out that stopping just before a terminal node thus gives rise to an ultimate "horizon" effect, in which the search strategy is totally insensitive to the proximity of a totally correct value. An example of this has recently been observed by the author [15, 16] in the investigation of a simple board-splitting game invented by Pearl [20]. This game has a complete game tree of constant branching factor, with independent random terminal node values. In [15] it is shown that when an obvious evaluation function for this game is used and when the depth of the game tree is larger than about 7 or 8, the probability of correct decision decreases with increasing search depth up to a point one move before the end of the game—at which point the probability of correct decision abruptly leaps to 1.

7.3. OVERCOMING PATHOLOGY. If a game is pathological, how should one go about making a decision? One possibility would be to use a very shallow search and a very accurate evaluation function for much of the game, not searching deeply until late in the game when one might be able to search to some of the terminal nodes. Another possibility would be to use a decision criterion other than minimaxing. One such decision criterion is investigated further in [16].

7.4. THE CAUSES AND NATURE OF PATHOLOGY. Although pathology occurs on almost all of the $G(m, n)$ (and also occurs in games such as the board-splitting game investigated in [15, 16]), it does not usually seem to occur on games such as chess or checkers. The most successful chess-playing computer programs have achieved their success by searching the game tree as deeply as possible, even at the expense of using a faster but less accurate evaluation function [3, 22, 28, 29]. We now discuss some possible reasons for this notable lack of pathology.

7.4.1. *Evaluation Function Accuracy.* Some readers have speculated that perhaps evaluation functions for games such as chess increase in accuracy toward the end of a game sufficiently rapidly that any underlying pathological tendency is overcome. However, experts consulted by this author [27, 30] expressed substantial disagreement with such speculations. Indeed, it seems that evaluation functions for chess are notoriously inaccurate in the endgame [30]. In this regard, it is interesting to note that in a game recently investigated by the author, pathology occurs even though the evaluation function increases markedly in accuracy toward the end of the game [15]. In another recent extension of the current paper, Pearl [21] has shown that for a certain class of game trees, the evaluation function must be at least twice as accurate at successive levels of the game tree if pathology is to be avoided.

7.4.2. *Correlation of Evaluation Function Values.* In games such as chess and checkers, the evaluation function value of a node is usually correlated with the evaluation function value of its parent. This also occurs on the $G(m, n)$ when $m > n$. However, games such as chess and checkers exhibit another kind of correlation

among evaluation function values which does not occur on the $G(m, n)$. As an example, consider the following game tree model, which was inspired by a similar model proposed by Newborn [17, p. 157] and later used by Lindstrom [11, p. 41].

Let G be a complete game tree of constant branching factor b and depth d. To the arcs of G we assign independent, identically distributed random values from some distribution which is symmetric around 0. We define the *strength* of a node in G to be the sum of the arc values on the path from that node back to the root, and we make a terminal node a win for Max if and only if it has positive strength. If a node g in G is a strong node, then the parent of g is likely to be strong, whence the siblings of g are likely to be strong. Similarly, if g is a weak node, then its siblings are likely to be weak. This is in marked contrast to any of the $G(m, n)$, where under any reasonable definition of strength, all nodes of the same type (S, T, U, or V) would be equally strong. It seems reasonable to assume that games such as chess or checkers are more similar to G than to any of the $G(m, n)$.

Recent experiments by the author [15, 16] show that when the obvious evaluation function for G is used, pathology does not occur, whereas in a closely related game where sibling nodes are independent of each other, pathology does occur. In [16], these experimental results are corroborated by theoretical results which generalize the results of the current paper. Thus pathology appears less likely if the game tree contains certain interdependencies which are similar to those that exist among games such as chess and checkers.

7.4.3. *Graphs versus Trees.* Many games are more properly represented as directed graphs than as trees, for if a game position can result from several different sequences of moves, then it is represented by more than one game tree node. Unless an evaluation function incorporates a random number generator,[5] it should return the same value for each of these nodes. But this violates the assumption of stochastic independence among the errors made by the evaluation function.

A good example of this is a class of nim-like games known as Bachet's game [1, 4, 8]. These games would be perfectly modeled as truncations of $G(m, n)$, except that the independence assumption is violated as explained above. It can be shown [12] that if an evaluation function is used on Bachet's game which makes random errors (but which returns the same value for a node every time it is applied to that node), the probability of correct decision is completely *independent* of the search depth. This is another interesting contradiction to the commonly accepted beliefs about searching deeper.

7.5. *Conclusion.* Pathology occurs almost universally on the $G(m, n)$. It also occurs on other game trees, although it would be difficult to say how common it is. The essential nature and causes of pathology are not yet thoroughly understood and are currently being investigated [16]. However, it is no longer possible to assume (as has been done in the past) that increasing the depth of search on a game tree will in general improve the quality of a decision.

Glossary

c.d.f.:	cumulative distribution function (Section 3.2.2).
correct move:	move to a "+" node if Max is making the move, or to a "–" node if Min is making the move (Section 3.1).

[5] Actually, some writers of game-playing computer programs like to do this, on the grounds that if the program always makes the same move in the same situation, a human opponent who notices this will have an "unfair" advantage [28]!

critical node:	node which has both "+" and "−" children (Section 2.1).
d:	depth of a minimax search (Section 2.2).
$D(G(m, n), g, e, d)$:	probability of correct decision at a critical node g of $G(m, n)$ (eqs. (3.5) and (3.6)).
e:	evaluation function (Section 2.2).
$e_d(g)$:	depth d minimax value of the node g, using the evaluation function e (eq. 2.1).
$\mathrm{err}_e(g)$:	error made by e at the node g (eq. 2.2).
evaluation function:	function which returns an estimate of the minimax value of a node (Section 1).
f_e^+:	p.f. for the values returned by e on "+" nodes (Section 2.2).
f_e^-:	p.f. for the values returned by e on "−" nodes (Section 2.2).
$f_{e(g)}$:	p.f. for $e(g)$ (eq. (2.4)).
$f_{e_d(g)}$:	p.f. for the depth d minimax value of g (Section 3.2.2).
$F_{e_d(g)}$:	c.d.f. for the depth d minimax value of g (Section 3.2.2).
f_{err_e}:	p.f. for the errors made by e (eq. 2.3).
g:	arbitrary game tree node (Section 2.2).
G:	arbitrary game tree (Section 2.2).
game tree:	tree whose nodes and arcs represent the positions and moves, respectively, in a perfect information game (Section 2.1).
$G(m, n)$:	infinite game tree in which each critical node has m children of the same sign ("+" or "−") as itself, and n children of opposite sign (Definition 3.1).
h and h':	smallest and largest possible values e returns with nonzero probability on "+" nodes (Section 2.3).
H:	maximum of the depth $d - 1$ minimax values of the children of an S node g (Section 3.2.3).
heuristic game tree searching:	making a decision by searching a game tree to some limited depth and using an evaluation function (Section 1).
I:	number of "+" children of g having the depth $d - 1$ minimax value H (Section 3.2.3).
J:	number of "−" children of g having the depth $d - 1$ minimax value H (Section 3.2.3).
m:	in $G(m, n)$, the number of "+" children of each critical "+" node and the number of "−" children of each critical "−" node (Definition 3.1).
Max and Min:	two players of a zero-sum game (Section 2.1).
minimax value:	defined in eq. (2.1).
n:	in $G(m, n)$, the number of "−" children of each critical "+" node and the number of "+" children of each critical "−" node (Definition 3.1).
noncritical node:	node for which all children are "+" nodes or all children are "−" nodes (Section 2.1).
pathological game tree:	game tree on which deeper search consistently increases the likelihood that decisions will be made totally at random (Section 5.3).
p.f.:	probability function (for a discrete random variable) (Section 2.3).

probability of correct decision:	probability that a correct move is chosen at a critical node of a game tree (eq. (3.5) and (3.6)).
r:	integer used to indicate the range $\{0, 1, \ldots, r\}$ of an evaluation function (Section 2.2).
S node:	"+" node where it is Max's move (Section 2.1).
s_d:	probability function for $e_d(s)$, where s is any S node (Section 3.2.2).
S_d:	c.d.f. for s_d (Section 3.2.2).
sign (of a node):	"+" (forced win) or "–" (forced loss) (Section 3.1).
symmetric bound for e:	number h such that the highest number e returns with nonzero probability is h, and the lowest number e returns with nonzero probability is $r - h$ (Section 2.3).
T node:	"+" node where it is Min's move (Section 2.1).
t_d:	probability function for $e_d(t)$, where t is any T node (Section 3.2.2).
T_d:	c.d.f. for t_d (Section 3.2.2).
U node:	"–" node where it is Min's move (Section 2.1).
u_d:	probability function for $e_d(u)$, where u is any U node (Section 3.2.2).
U_d:	c.d.f. for u_d (Section 3.2.2).
V node:	"–" node where it is Max's move (Section 2.1).
v_d:	probability function for $e_d(v)$ where v is any V node (Section 3.2.2).
V_d:	c.d.f. for v_d (Section 3.2.2).
$w(n)$:	unique root in the interval (0, 1) of the equation $(1 - (1 - x)^n)^n = x$ (Section 4).
x_i and y_i:	two sequences jointly defined by eq. (4.1) and (4.2).
z_i:	sequence defined in the statement of Theorem 4.1.
"+" node:	node where Max has a forced win (Section 2.1).
"–" node:	node where Min has a forced win (Section 2.1).

REFERENCES

1. BANERJI, R.B., AND ERNST, G.W. Strategy construction using homomorphisms between games. *Artif. Intell. 3* (1972), 223–249.
2. BAUDET, G.M. On the branching factor of the alpha–beta pruning algorithm. *Artif. Intell. 10* (1978), 173–199.
3. BIERMANN, A.W. Theoretical issues related to computer game playing programs. *Personal Comput.* (Sept. 1978), 86–88.
4. DOMORYAD, A.P. *Mathematical Games and Pastimes.* MacMillan, New York, 1964.
5. FULLER, S.H., GASCHNIG, J.G., AND GILLOGLY, J.J. Analysis of the alpha–beta pruning algorithm. Tech. Rep., Dept. of Computer Science, Carnegie-Mellon Univ., July 1973.
6. GREENBLATT, R.D., III, EASTLAKE, D.E., AND CROCKER, S.D. The Greenblatt chess program. *Proc. AFIPS Fall Joint Computer Conference 31,* AFIPS Press, Arlington, Va., (1967), 801–810.
7. HARRIS, L.R. The heuristic search under conditions of error. *Artif. Intell. 5* (1974), 217–234.
8. JACKSON, P.C. *Introduction to Artificial Intelligence,* Petrocelli, New York, 1974.
9. KNUTH, D.E., AND MOORE, R.W. An analysis of alpha–beta pruning. *Artif. Intell. 6* (1975), 293–326.
10. LAVALLE, I.H. *Fundamentals of Decision Analysis.* Holt, Rinehart and Winston, New York, 1978.
11. LINDSTROM, G. Alpha–beta pruning on evolving game trees. Tech. Rep. UUCS 79-101, Dept. of Computer Science, Univ. of Utah, March 1979.
12. NAU, D.S. Unpublished notes, 1979.
13. NAU, D.S. Quality of decision versus depth of search on game tress. Ph.D. Dissertation, Duke Univ., Aug. 1979.

14. NAU, D.S. The last player theorem. *Artif. Intell. 18* (1982), 53–65.
15. NAU, D.S. An investigation of the causes of pathology in games. *Artif. Intell. 19* (1982), 257–278.
16. NAU, D.S. Pathology on game trees revisited, and an alternative to minimaxing. *Artif. Intell. 21*, 1/2 (1983), 222–244.
17. NEWBORN, M.M. The efficiency of the alpha–beta search on trees with branch-dependent terminal node scores. *Artif. Intell. 8* (1977), 137–153.
18. NILSSON, N.J. *Problem-Solving Methods in Artificial Intelligence.* McGraw-Hill, New York, 1971.
19. PEARL, J. Asymptotic properties of minimax trees and game-tree searching procedures. *Artif. Intell. 14* (1980), 113–138.
20. PEARL, J. Colloquium talk, Univ. of Maryland, Aug. 1980.
21. PEARL, J. On the nature of pathology in game searching. Tech. Rep. UCLA-ENG-CSL-82-17, School of Engineering and Applied Science, Univ. of California, Los Angeles, Calif., Jan. 1982.
22. ROBINSON, A.L. Tournament competition fuels computer chess. *Science 204* (1979), 1396–1398.
23. SAMUEL, A.L. Some studies in machine learning using the game of checkers, II. Recent progress. *IBM J. Res. Devel. 2* (1967), 601–617.
24. SHANNON, C. Programming a computer for playing chess. *Phil. Mag. 7*, 14 (1950), 256–275.
25. SLAGLE, J.R. *Artificial Intelligence: The Heuristic Programming Approach.* McGraw-Hill, New York, 1971.
26. STOCKMAN, G.C. A minimax algorithm better than alpha–beta? *Artif. Intell. 12* (1979), 179–196.
27. THOMPSON, K. Personal communication, Jan. 1981.
28. TRUSCOTT, T.R. Personal communication, 1979.
29. TRUSCOTT, T.R. Minimum variance tree searching. Proc. 1st Int. Symp. on Policy Analysis and Information Systems, Durham, N.C., June 1979, pp. 203–209.
30. TRUSCOTT, T.R. Personal communication, Jan, 1981.
31. TUMMALA, V.M.R. *Decision Analysis with Business Applications.* Intext, New York, 1973.

RECEIVED FEBRUARY 1980; REVISED DECEMBER 1982; ACCEPTED JANUARY 1983