

QUALITY OF DECISION VERSUS DEPTH
OF SEARCH ON GAME TREES

by

Dana S. Nau

Department of Computer Science
Duke University

Date: _____

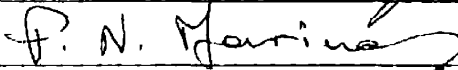
Approved:

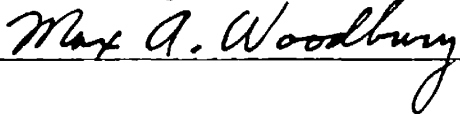


Alan W. Biermann, Supervisor



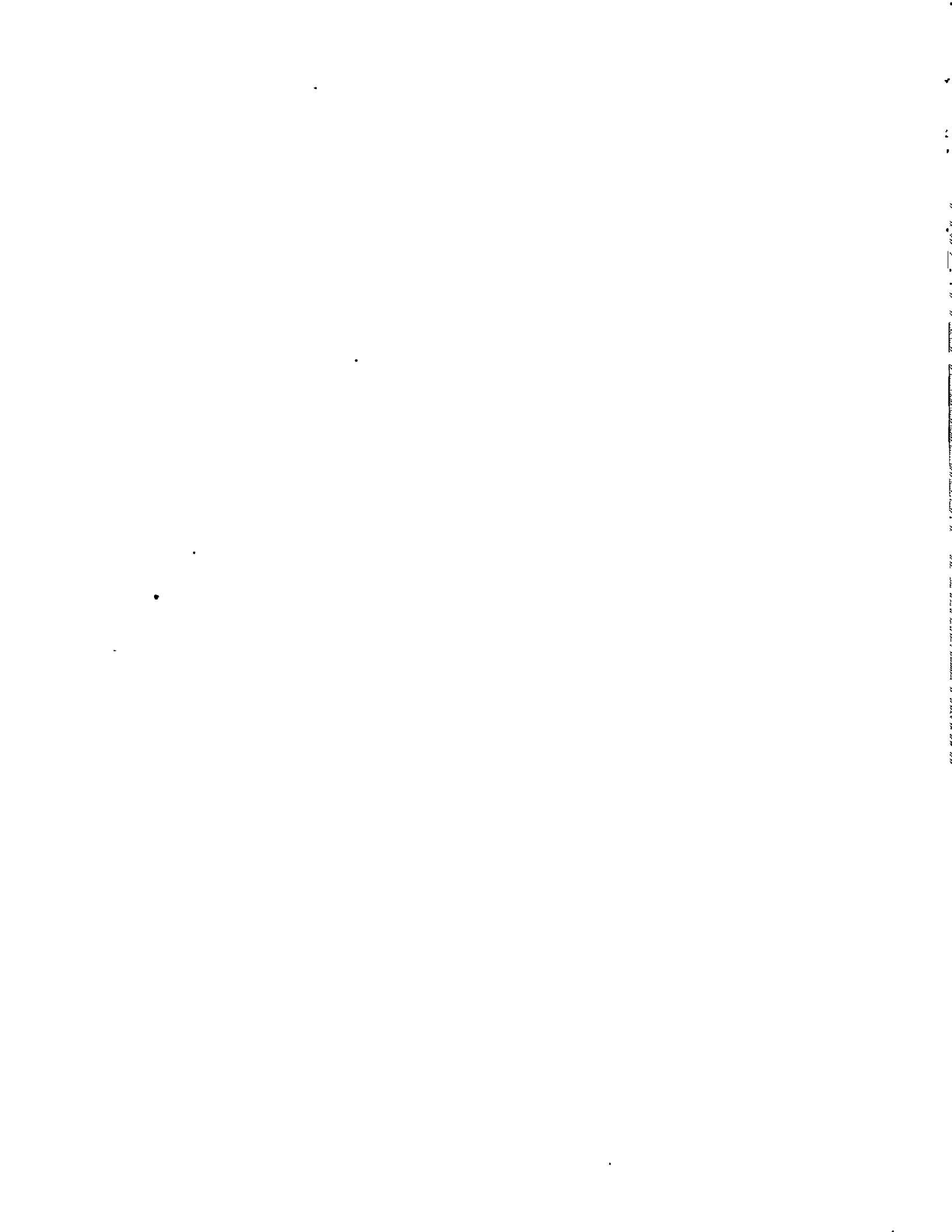






Dissertation submitted in partial fulfillment of
the requirements for the degree of Doctor
of Philosophy in the Department of
Computer Science in the Graduate
School of Duke University

1979



ABSTRACT

(Computer Science)

QUALITY OF DECISION VERSUS DEPTH
OF SEARCH ON GAME TREES

by

Dana S. Nau

Department of Computer Science
Duke University

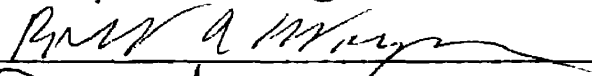
Date: _____

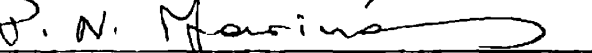
Approved:

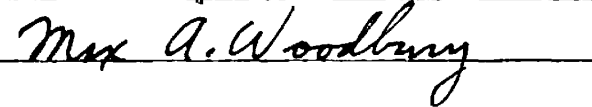


Alan W. Biermann, Supervisor









An abstract of a dissertation submitted in partial
fulfillment of the requirements for the degree
of Doctor of Philosophy in the Department of
Computer Science in the Graduate School of
Duke University

ABSTRACT

There is probably not a single area in computer science in which use is not made of the mathematical structure known as a tree. Many of the properties of trees are little understood, and this dissertation is concerned with some of them. Specifically, the relationships between depth of tree search and quality of behavior in decision making are studied.

The well-known problem reduction approach to problem solving intrinsically involves tree searching. 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. If correct results are to be guaranteed when using problem reduction, substantial portions of this tree must be completely searched, which is physically impossible for very large trees.

In the field of game playing, good results have been obtained by searching the game tree to some arbitrary depth and heuristically estimating its characteristics beyond that depth. For such a heuristic game tree search, there is a consensus that the quality of the decision improves as the search depth increases, but this agreement 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 using a heuristic game tree search. 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 dissertation contains a mathematical proof of this statement, experimental verification of it, and a discussion of its significance for decision making in general.

This work has been supported in part by a National Science Foundation graduate fellowship, and in part by a James B. Duke graduate fellowship.

TABLE OF CONTENTS

ABSTRACT	ii
LIST OF ILLUSTRATIONS	vii
LIST OF TABLES	viii
ACKNOWLEDGEMENTS	x
CHAPTER 1. INTRODUCTION	1
Problem Reduction	
Problem Reduction as Decision Analysis	
The Problem of Search Depth	
CHAPTER 2. A MATHEMATICAL MODEL	8
Introduction	
Game Trees	
Evaluation Functions	
Choosing a Correct Move	
CHAPTER 3. "HOMOGENEOUS" GAMES	31
The Basic Definition	
Truncation	
Independence Results	
CHAPTER 4. A PATHOLOGICAL GAME	39
Introduction	
Theorems about Probability Vectors	
Theorems about Sequences	
The Pathology Theorem for $\Gamma(1,1)$	
Experimental Results	
CHAPTER 5. THE "LAST PLAYER" THEOREM	60
Introduction	
Theorems about Sequences	
The Theorem and a Discussion	

Continued on Next Page

CHAPTER 6. PATHOLOGY ON $\Gamma(m,n)$	71
Introduction	
Theorems about the Probability of Correct Decision	
Double Sequences	
The Main Theorem	
CHAPTER 7. EXPERIMENTAL RESULTS	82
The Importance of a Good Evaluation Function	
The Shape of the Nonpathological Region	
The Values of the Limiting Vectors	
Rates of Convergence	
CHAPTER 8. SUMMARY, CONCLUSIONS AND SPECULATIONS	101
Summary of Results	
Conclusions and Speculations	
APPENDIX A. AN EXTENTION OF TABLE 5.1	107
APPENDIX B. $D(P,d)$ FOR PERFECT P	109
APPENDIX C. A GENERALIZATION OF w_n	111
APPENDIX D. MORE THEOREMS ABOUT DOUBLE SEQUENCES	113
APPENDIX E. PATHOLOGY FOR UNNATURAL PROBABILITY VECTORS	115
APPENDIX F. ALTERNATING PATHOLOGICAL AND NONPATHOLOGICAL BEHAVIOR	117
APPENDIX G. A COMPUTER PROGRAM	120
APPENDIX H. INVESTIGATING PATHOLOGY IN PRACTICAL SITUATIONS	127
LIST OF REFERENCES	130

LIST OF ILLUSTRATIONS

1.1.	Probability of correct decision as a function of search depth on the game tree $\Gamma(1,1)$, for five different classes of evaluation functions	7
2.1.	A portion of an arbitrary game tree	10
2.2.	Examples of S, T, U, and V nodes	11
2.3.	Minimax values computed using a depth 3 minimax search on the game tree of Figure 2.1	25
2.4.	The set of all probability vectors (p_1, p_2, p_3) forms a triangular plane segment	29
3.1.	The tree $\Gamma_S(m, n)$	32
4.1.	Probability of correct decision as a function of search depth, for five different probability vectors	59
5.1.	A binary tree of depth 4 with a forced win for Max	61
5.2.	Sketches of $y = (1-x)^2$, $y = (1 - (1-x)^2)^2$, and $y = x$	64
5.3.	The probability $W_{n,d}$ of the last player having a forced win on a complete n-ary game tree of depth d, as a function of d	69
6.1.	The set of all unnatural $\hat{2}$ -vectors is a proper subset of the union of the two line segments	73
6.2.	The set of all $\hat{2}$ -vectors forms a triangular plane segment	73
6.3.	Pathological and nonpathological behavior of $\Gamma(m, n)$ for a natural \hat{f} -vector P, as a function of m and n	74
7.1.	Pathological and nonpathological behavior of $\Gamma(m, n)$ for a natural probability vector P, as a function of m and n	91

LIST OF TABLES

2.1.	Characteristics of game tree nodes	12
2.2.	The computation of \bar{Q}_e and Q_e for an arbitrary evaluation function \bar{e}	16
4.1.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.1,.2,.3,.4)$	56
4.2.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.4,.3,.2,.1)$	56
4.3.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (1,5,25,125,625)/781$	57
4.4.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (625,125,25,5,1)/781$	57
4.5.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.2,.2,.2,.2,.2)$	58
5.1.	Approximate values of w_n , for $n = 1,2,\dots,50$	64
5.2.	The probability $w_{n,d}$ of the last player having a forced win on a complete n -ary game tree of depth d , for various values of n and d	69
7.1.	Probabilities of correct decision on $\Gamma(1,1)$ at various search depths for each of P_1, P_2, P_3, P_4 , and P_5	84
7.2.	The number of trees $\Gamma(m,n)$ which Theorem 6.4 says need not be pathological, for each of P_1, P_2, P_3, P_4 , and P_5	84
7.3.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_1 = (1,2,3,4,5,6,7,8,9,10,11,12)/78$	86

Continued on Next Page

7.4.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_2 = (3,7,11,15,19,23)/78$	87
7.5.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_3 = (6,15,24,33)/78$	88
7.6.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_4 = (10,26,42)/78$	89
7.7.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_5 = (21,57)/78$	90
7.8.	d_0 as a function of m and n , for the probability vector $P_1 = (1,2,3,4,5,6,7,8,9,10,11,12)/78$	96
7.9.	d_0 as a function of m and n , for the probability vector $P_2 = (3,7,11,15,19,23)/78$	97
7.10.	d_0 as a function of m and n , for the probability vector $P_3 = (6,15,24,33)/78$	98
7.11.	d_0 as a function of m and n , for the probability vector $P_4 = (10,26,42)/78$	99
7.12.	d_0 as a function of m and n , for the probability vector $P_5 = (21,57)/78$	100
A.1.	Values of w_n to six decimal places, for $n = 1,2,\dots,n,350$	107
B.1.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.0,.0,.5,.5)$	109
B.2.	Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.5,.5,.0,.0)$	110
F.1.	Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P = (.1108721768, .0184577789, .0278285401, .6653850732, .0358106695, .0223749103, .1192708512)$	118
F.2.	d_0 as a function of m and n , for the probability vector $P = (.1108721768, .0184577789, .0278285401, .6653850732, .0358106695, .0223749103, .1192708512)$	119

ACKNOWLEDGEMENTS

I owe a special debt of gratitude to my faculty advisor, Alan Biermann, for his enthusiasm and advice, and for his unfailing willingness to listen. His encouragement and his confidence in my work kept me going at times when I otherwise would have given up.

Appreciation goes to B. Chandrasekaran of Ohio State University, who gave me the idea for this dissertation when he suggested investigating "when $n+1$ level search is better than n ."

For items of information which turned out to be useful, I thank Doug Smith and Tom Truscott. For assistance with typesetting the dissertation, I thank Tim Sigmon and David Lupo. Thanks also to the members of my Ph.D. committee for their interest in my work.

This work could not have been done without the computer science department's "Unix" computer system. Many of the mathematical intuitions which led to the theorems presented in this dissertation came from hours of sitting at computer terminals computing probability vectors.

This work was supported in part by National Science Foundation and James B. Duke graduate fellowships.

CHAPTER 1

INTRODUCTION

Problem Reduction

There is probably not a single area in computer science in which use is not made of the mathematical structure known as a tree. Trees have many unusual and surprising properties. For example, infinite trees often have uncountably many branches, and trees usually have as many or more nodes at level n than at all previous levels combined. Many of the properties of trees are little understood, and this dissertation is concerned with some of them. Specifically, the relationships between depth of tree search and quality of behavior in decision making are studied.

One of the more common uses of trees in computer science is in problem reduction. This is the process of solving a complex problem by dividing it up into smaller problems. Every time a problem is divided into subproblems, the situation can be modeled by a tree of one level in which the root node represents the original problem and its children represent the subproblems. As the subproblems are in turn divided into sub-subproblems, and so forth, a tree of many levels is generated. When subproblems are found which are simple enough to solve directly, their solutions are used to construct a solution for the original problem.

Problem reduction trees are studied in decision analysis under the name of decision trees [LA1, ST1, TU1]. They are of

interest in many different areas of computer science, including theoretical computer science [AH1, CH1], pattern recognition [ST2, TO1], program synthesis [BI1], theorem proving [NI1, LO1, SL1, VA1], game playing [BA3, BI2, FU1, GR1, HA1, JA1, KN1, LI1, NA1, NI1, RO1, SA1, SL1, TR1, TR2], and other areas [DA1, JA1, NI1, SH1].¹ Every problem reduction tree has an equivalent canonical representation in which the nodes are partitioned into two classes, which are known variously as "and" nodes and "or" nodes, min nodes and max nodes, and so forth. What this representation means is discussed in more detail later.

Problem Reduction as Decision Analysis

Problem reduction can be described as solving a problem by recursive decision-making. For example, when problem reduction is used in theorem proving, each step in the reduction corresponds to asking whether some statement in first order logic can be proved using various combinations of simpler statements, and this is decided by determining whether the simpler statements are themselves provable. When problem reduction is used in program synthesis, the decisions are concerned with whether a program specification can be represented as a series of simpler specifications, and this is decided by deciding how the simpler specifications must be represented.

In decision analysis, it is standard to model problem reduction as the playing of a game between the decision maker and an opponent, whom we will call Max and Min, respectively. Min

¹Fault trees [BA2] are also problem reduction trees, but we are not concerned with them here.

may either be a malicious opponent, as in the case of a real game, or may be a representation of the unpredictable responses of Max's environment to his actions. The problem reduction tree is called a decision tree or game tree, and the canonical tree mentioned at the end of the previous section corresponds to a game in which Max and Min make moves in strict alternation. The two types of nodes in this game tree are those at which it is Max's move and those at which it is Min's move.

In this game-playing model, problem reduction corresponds to looking ahead. The problem of winning the game from the current position is attacked by reducing it to the problems of winning the game from each of the possible next positions. These problems are further reduced in the same way, and so on, until the end of the tree is reached. At this point, the payoffs corresponding to each leaf node are determined, and are used to compute for the other nodes of the tree what are known in decision analysis and mathematical game theory as utility values. Max's action in making the decision then consists of moving from the root node of the game tree to whichever of its children has the largest utility value.

Depending on the problem domain, the utility values may be computed according to any of a number of different criteria [LAI, TU1]. In a real game situation, where Min is assumed to be malicious, the maximin criterion is normally used. This is also the criterion most often used in computer science applications, although it is not called by that name.

In computer science applications, the usual situation is that each leaf of the tree has the value one or zero, depending

on whether or not the corresponding primitive problem can be used to solve its parent problem. In this case, the use of the maximin criterion corresponds to recursive "and" and "or" operations. This is why computer scientists often refer to problem reduction trees as "and-or" trees.

In computer game playing, a wider range of utility values is used, and here the values are referred to as minimax values. Luckily, computer scientists have used this term only in situations where the maximin and minimax criteria are equivalent according to von Neumann's minimax theorem [LA1, OW1, SI1, TU1]. "Maximin values" would actually be a more proper term.

The Problem of Search Depth

Since the number of nodes in a tree generally increases exponentially with the depth of the tree, searching a tree of any large depth is usually a time-consuming task. In fact, many of the problems which are normally attacked by searching trees have been shown to be NP-hard [AH1, CH1, KA1].

When a problem reduction tree is large, it is usually infeasible to search the entire tree. For this reason, much of the computer science research on problem reduction techniques has been directed at procedures which can be guaranteed to find a correct solution without looking at every node in the tree (for example, the MESON system in theorem proving [LO1] and the alpha-beta procedure in game playing [KN1, NI1]). But the guarantee of correct results still requires complete search of some portions of the tree, and problem reduction trees are often so large that this is physically impossible.

In game playing computer programs, good results have been obtained by the heuristic procedure of searching the tree only to some severely limited depth, estimating the utility values of the nodes at that depth, and then proceeding, in the usual manner, to compute utility values for the shallower nodes in the tree as if the estimated utility values were in fact correct. [BI1, GR1, HA1, JA1, KN1, NI1, RO1, SA1, SL1, TR1, TR2]. This technique, which we shall 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. To what extent it has been used or can be used in some applications is unclear. In theorem proving, for example, there has not been much interest in using the computer to find probable proofs of theorems until quite recently [DE1, RA1].

When heuristic game tree searching is used, it is almost universally agreed that increasing the search depth improves the quality of the decision. There have been some dramatic demonstrations of this with game playing computer programs [BI2, RO1, TR1, TR2]. However, 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 dissertation is concerned with a mathematical theory modeling the effects of searching deeper on the probability that a decision is correct. The results of the research are stated assuming the use of "minimaxing," i.e., the maximin criterion mentioned earlier. As discussed in Chapter 8, the results should also extend to situations in which other decision criteria are used. The major result of this study is that contrary to the

conventional belief, there is a large class of game trees such that as long as the search does not reach the end of the tree (in which case the real utility values could be computed), deeper search will consistently cause the decision to become more and more random, rather than increasing the probability that it is correct. As an example, Figure 1.1 duplicates a figure from Chapter 4 which shows how the probability of correct decision varies.

This idea may seem counter-intuitive. One might suppose that since searching deeper gives more information, it cannot fail to improve the decision quality. But the reader should keep in mind that the evaluation function produces estimates of the utility values, 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.

Chapter 2 is a presentation of the mathematical model which is used in the dissertation. Chapter 3 is a description of an infinite class of game trees having a regular enough structure that useful theorems can be stated about them. Chapters 4, 5, and 6 provide the central mathematical results of the dissertation. These results are experimentally verified at the end of Chapter 4 and in Chapter 7. Concluding remarks appear in Chapter 8.

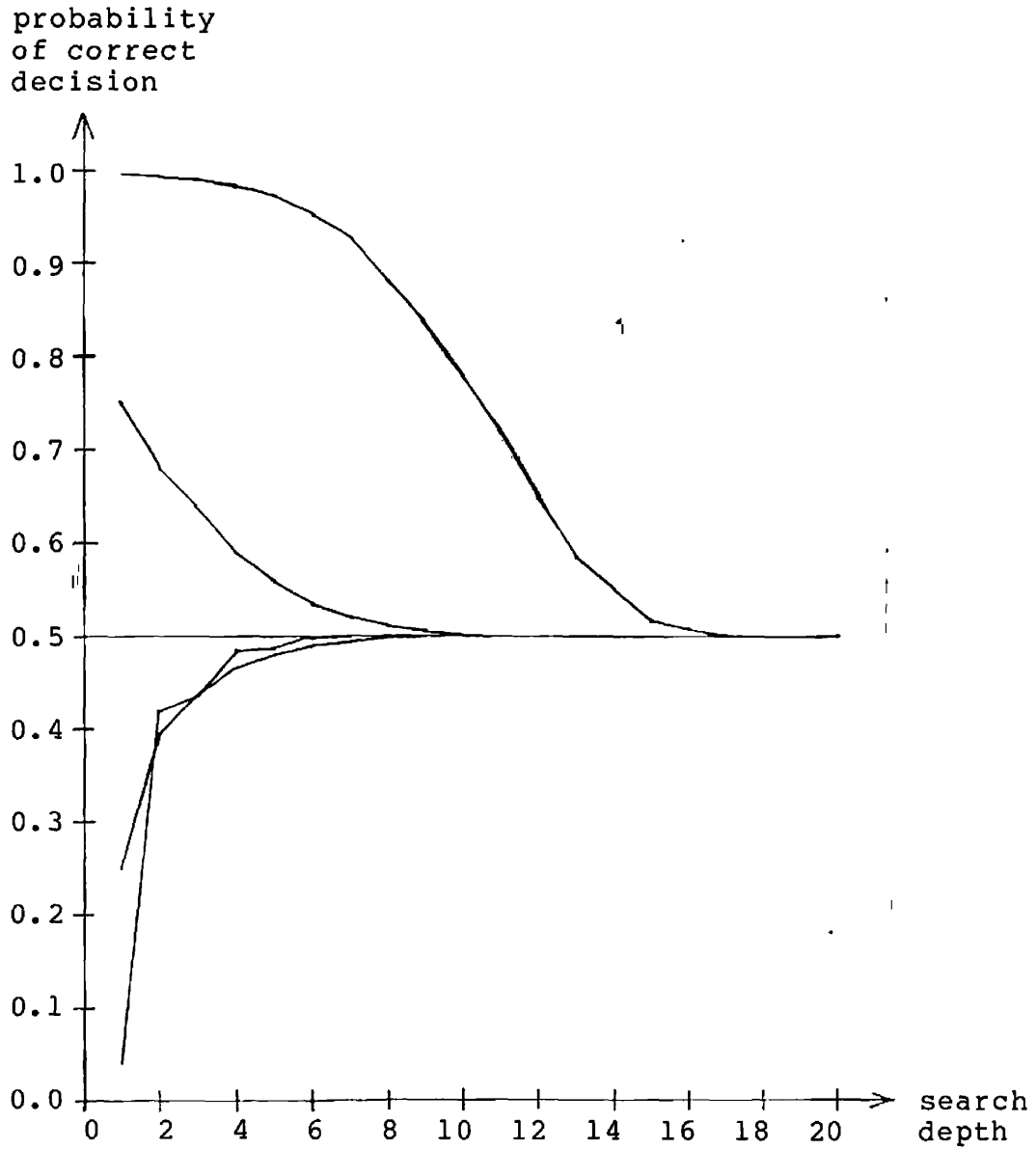


FIGURE 1.1.--Probability of correct decision as a function of search depth on the game tree $\Gamma(1,1)$, for five different classes of evaluation functions. On $\Gamma(1,1)$, the probability of correct decision is 0.5 if the choice is made at random. For each of the five classes, this value is approached as the search depth increases.

CHAPTER 2

A MATHEMATICAL MODEL

Introduction

In order to set up a mathematical model for heuristic game tree searching, several formal definitions are needed. This chapter contains most of them, along with explanations and examples.

In this dissertation, the word "game" is used to mean a zero sum, two person, perfect information game between two players, in which the outcome is not determined even partially by chance (as would occur in dice games and most card games). In addition, the number of possible moves among which a player can choose at each position in a game is restricted to be finite (although it may be very large). However, we do not require that every game end in a finite number of moves.

The above restrictions yield a class of games including such parlor games as chess, checkers, othello, go, nim, tic-tac-toe, and so forth. However, the model is certainly not restricted to parlor games!

In addition to the above restrictions, we shall only consider games in which the play alternates strictly between the two players, and in which there are no draws. These two restrictions can easily be removed, and are stipulated mainly to simplify the mathematics. Any game G_0 in which a player can make several moves in a row may easily be transformed into a game G_1

with strict alternation of play, simply by inserting null moves for one player or the other wherever necessary. If G_1 contains draws, it may easily be transformed into two games G_2 and G_3 without draws, simply by mapping draws into wins and losses, respectively. The properties of G_0 are easily determined from the properties of G_2 and G_3 .

With a little more work, it may also be possible to remove several of the other restrictions, but the author has not paid much attention to this question. This provides a topic for future research.

Game Trees

Since all games under consideration are between two players, it is convenient to call those players Max and Min. Game positions in which Max has won or has a forced win we label "+", and game positions in which Min has won or has a forced win we label "-". Since the games we are considering do not have draws, it is simple to prove inductively that every game position is a forced win for either Max or Min. Thus, every position is labeled either "+" or "-" (but not both).

The set of all possible courses a game might take can be represented by a game tree in which the root node corresponds to the starting position of the game, and the children of each node correspond to the positions which can be reached in one move from that node. Game trees are defined formally in Definition 2.2, and an example game tree is shown in Figure 2.1.

Definition 2.1. If G is a tree, then $N(G)$ denotes the set of nodes of G . A leaf of G is a node $g \in N(G)$ which has no

depth 0 (root node):

Max's move

depth 1:

Min's
move

depth 2:

Max's
move

depth 3:

Min's
move

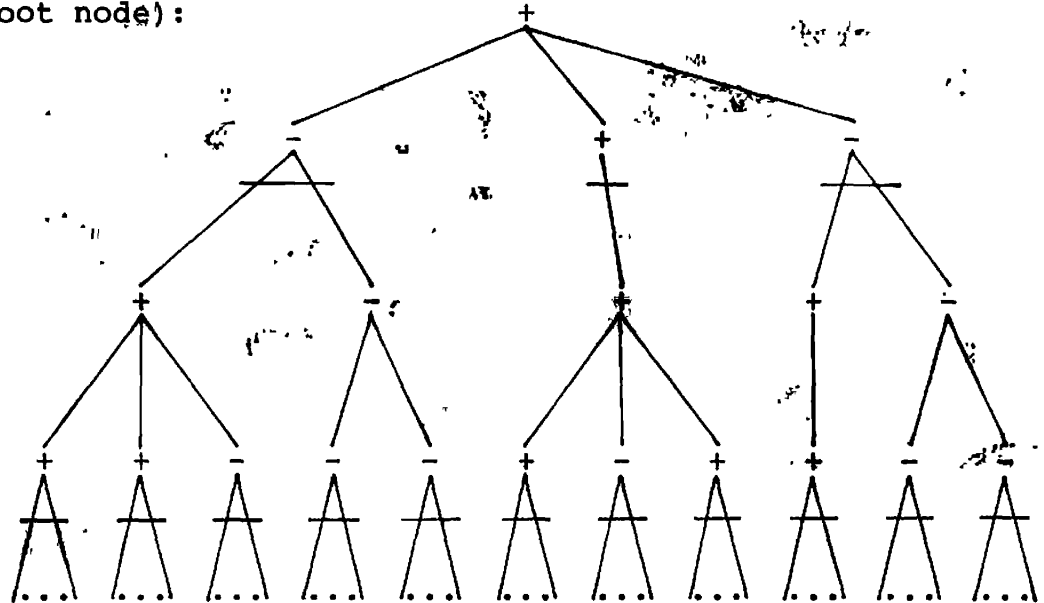


FIGURE 2.1.--A portion of an arbitrary game tree. Min nodes are indicated by the horizontal line segments drawn beneath them.

children.

Definition 2.2. A game tree is a tree G such that--

1. $N(G)$ is partitioned into two sets, called max nodes and min nodes, in such a way that every child of a max node is a min node, and vice versa;
2. $N(G)$ is partitioned into two sets, called "+" nodes and "-" nodes, in such a way that each min node is a "+" node if and only if all of its children are "+" nodes, and each max node is a "-" node if and only if all of its children are "-" nodes;
3. every node of G has a finite number of children.

Max nodes correspond to game positions where it is Max's move, and min nodes correspond to positions where it is Min's

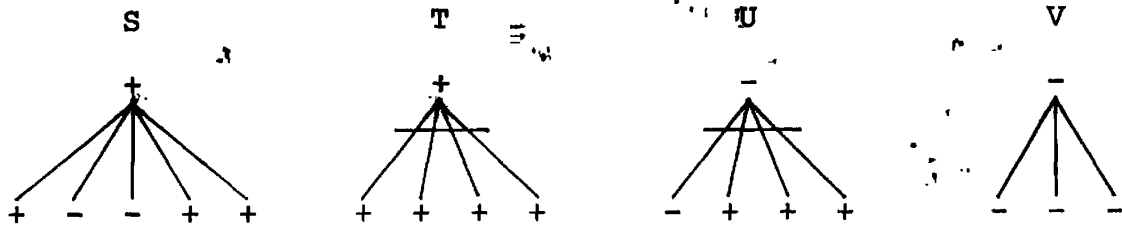


FIGURE 2.2.--Examples of S, T, U, and V nodes. Min nodes are indicated by the horizontal line segments drawn beneath them.

move. For finite game trees, "+" nodes and "-" nodes correspond to forced wins for Max and Min, respectively. This is not necessarily true on infinite game trees, since they represent games which may not always end. However, it is useful to require "+" and "-" labels on all nodes of all game trees.

According to Definition 2.2, every game tree node is classified according to whose move it is and who has a forced win. Thus there are four different types of nodes, as defined in Definition 2.3 below. Examples of these nodes are given in Figure 2.2.

Definition 2.3. Let G be a game tree. S is the set of all "+" max nodes of G. T is the set of all "+" min nodes of G. U is the set of all "-" min nodes of G. V is the set of all "-" max nodes of G. Members of S, T, U, and V are called S nodes, T nodes, U nodes, and V nodes, respectively.

Definition 2.4. S and U nodes are called critical nodes, for it is only at these nodes that it makes a difference which move is chosen. T nodes and V nodes are called noncritical nodes.

From Definition 2.3, it is obvious that--

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

A U node is to Min what an S node is to Max, so S and U nodes have the same kinds of properties. A similar statement holds for T and V nodes. Several of the classifications and properties of game tree nodes are summarized in Table 2.1.

TABLE 2.1.--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

Evaluation Functions

In game playing and other decision-making situations, people may reason in a manner similar to the following:

If I move to position A, then my opponent will probably move to position B, which will leave me in a bad position. However, if I move to position C, then my opponent will not have any good moves. Therefore, I'll move to position C.

Such reasoning implicitly involves estimating how good various game positions are. Although it is not understood how human beings make such estimates, game playing computer programs generally make them by using heuristic evaluation functions. These are functions which map game positions into numbers indicating how good the positions are estimated to be. A high

number means that the position is estimated to be good, say, for Max (and therefore bad for Min); a low number means the reverse. For example, an evaluation function for chess might take into consideration which pieces are on the board, where they are and how mobile they are, who has control of the center of the board, whether either player's king is exposed, and so forth. The estimations made by an evaluation function are usually somewhat (and sometimes drastically) inaccurate. For the evaluation functions used in today's best chess playing computer programs, there are situations under which obvious forced loss positions can be made to look as good as desired, and vice versa [TR1].

Formally, an evaluation function is any mapping from the set of nodes of a game tree into a set of numbers. If the evaluation function is to be one which can be implemented on a computer, then the set of numbers must be finite (although perhaps very large). For convenience, we take this set to be $\{0, 1, \dots, r\}$, where r is an arbitrary integer. Then evaluation functions may formally be defined as follows.

Notation 2.1. Except where otherwise stated, G denotes a game tree, g denotes a node of G , and r denotes a nonnegative integer.

Definition 2.5. \hat{r} is the set $\{0, 1, \dots, r\}$.

Definition 2.6. An evaluation function on G is any function $e: N(G) \rightarrow \hat{r}$. $E_{G,r}$ is the set of all such functions. (Thus if G is finite, then $|E_{G,r}| = |N(G)|^{r+1}$.)

Note that Definition 2.6 places no restriction on how accurate or inaccurate an evaluation function must be. Every function from $N(G)$ into \hat{r} is considered to be an evaluation

function, including functions that return inaccurate values on leaf nodes of G . At each leaf node, one or the other of the players has won the game, and some readers may prefer that no matter how inaccurate an evaluation function is elsewhere, it should at least be able to detect who has won. Given such a restriction, the definition of an evaluation function would instead be as follows.

Let $L(G)$ be the set of leaf nodes of G , and let $M(G)$ be the set of non-leaf nodes. Let $E'_{G,r}$ be the set of all functions $e: M(G) \rightarrow \hat{r}$, and let $f: L(G) \rightarrow \hat{r}$ be defined by

$$\begin{aligned} f(g) &= 0 \text{ if } g \text{ is a "-" leaf,} \\ &= r \text{ if } g \text{ is a "+" leaf.} \end{aligned}$$

Then $E_{G,r}$ would be defined as $\{e \cup f: e \in E'_{G,r}\}$.

Definition 2.6 was chosen because it is both simpler and more general than the above definition. However, which definition is used is of no great importance, and the reader is welcome to use either one.

If e is a good evaluation function, then $e(g)$ will generally be high if g is a "+" node and low if g is a "-" node. However, e will sometimes make mistakes; i.e., there will be some "-" nodes to which e gives high evaluations and some "+" nodes to which e gives low evaluations. For example, having more pieces than one's opponent correlates with having a good position in chess, but it does not always mean that the position is a good one. With an error-free evaluation function, tree search would not be necessary, for perfect play could be obtained by directly evaluating each possible successor to the current position, and choosing the one of highest value.

Since an evaluation function can be any function from $N(G)$ into \mathbb{R} , it follows that for almost any property one might want to state about the behavior of evaluation functions in terms of increasing search depth, a function can be constructed which fails to have that property. By increasing or decreasing its accuracy at positions far down in the tree, an evaluation function can be made to give either good or bad results at large search depths.

A convenient way to avoid this difficulty is to develop a model which makes probabilistic rather than universal predictions about the behavior of evaluation functions. This is the course chosen here.

When setting up a probabilistic model of a set of deterministic objects, the usual procedure is to categorize the objects in terms of whatever features are considered to be relevant, and to consider each object X as a random element from the equivalence class of all objects having the same features as X . The feature which we shall choose to consider important for an evaluation function is how "good" it is, averaged over all nodes in the game tree.

Let $e \in E_{G,r}$ be an evaluation function. If G is a finite game (such as chess or checkers), one way to characterize e 's average behavior is to tabulate the number of nodes for which e returns each of the numbers in its range. More specifically, suppose we let $\bar{Q}_e = (\bar{q}_0, \bar{q}_1, \dots, \bar{q}_r)$, where each \bar{q}_i is the number of nodes g of G such that $e(g) = i$. (For an example, see the first part of Table 2.2.) Then for each Q , the set $\bar{E}_{G,r,Q}$ of all evaluation functions e such that $\bar{Q}_e = Q$ is a set of evaluation

functions which in some sense behave similarly.

TABLE 2.2.--The computation of \bar{Q}_e and Q_e for an arbitrary evaluation function e .

i	0	1	2	3	4
number of nodes g such that $e(g) = i$	21	14	7	11	27
\bar{q}_i	21	14	7	11	27
number of "+" nodes g such that $e(g) = i$	2	1	3	10	27
number of "-" nodes g such that $e(g) = i$	19	13	4	1	0
number of "-" nodes g such that $4-e(g) = i$	0	1	4	13	19
q_i	2	2	7	23	46

The problem with the above characterization is that in general, both good evaluation functions and bad ones may have the same \bar{q}_i values, and thus $\bar{E}_{G,r,Q}$ may contain evaluation functions of widely varying accuracy. We want to put good and bad evaluation functions into different equivalence classes.

If e is a good evaluation function, then $e(g)$ will usually be high if g is a "+" node and low if g is a "-" node, and if e is a bad evaluation function, then $e(g)$ will usually be low if g is a "+" node and high if g is a "-" node. For the rest of this chapter only, let us define the operator "'" by

$$\begin{aligned}
 e'(g) &= e(g) && \text{if } g \text{ is a "+" node,} \\
 &= r-e(g) && \text{if } g \text{ is a "-" node.}
 \end{aligned}$$

If e is good, then $e'(g)$ will usually be high, regardless of whether g is a "+" node or a "-" node. If e is bad, then $e'(g)$

will usually be low.

For G finite, let $Q_e = (q_0, q_1, \dots, q_r)$, where for each $i \in \hat{r}$, q_i is the number of nodes g of G such that $e'(g) = i$ (for example, see Table 2.2). If e is good, then q_i will usually be small if i is small and large if i is large, and vice versa if e is bad. If we let $E_{G,r,Q}$ be the set of all evaluation functions e for which $Q_e = Q$, the evaluation functions in $E_{G,r,Q}$ can be said to have the same average accuracy.

From the above, it appears that the sets $E_{G,r,Q}$ may be a reasonable way to characterize the performance of evaluation functions. But to have equivalence classes of evaluation functions based on this characterization, we must know that the sets $E_{G,r,Q}$ are a partition of $E_{G,r}$. This is shown in Theorem 2.1 below. The proof of this theorem makes implicit use of the following propositions.¹

Notation 2.2. For the rest of this chapter only, Q denotes a vector (q_0, q_1, \dots, q_r) , where each q_i is a nonnegative integer, and where $q_0 + q_1 + \dots + q_r = |N(G)|$.

Proposition 2.1. If G is finite, then--

1. the operator "' maps $E_{G,r}$ one-to-one onto $E_{G,r}$;
2. $e'' = e$ for every $e \in E_{G,r}$;
3. the operator "' maps $E_{G,r,Q}$ one-to-one onto $\bar{E}_{G,r,Q}$, and vice versa.

Statement 3 of Proposition 2.1 tells us that $E_{G,r,Q} = \{e: e' \in \bar{E}_{G,r,Q}\}$ and $\bar{E}_{G,r,Q} = \{e: e' \in E_{G,r,Q}\}$.

¹In this dissertation, a proposition is a statement whose proof is obvious. Thus all propositions are stated without formal proofs.

Proposition 2.2. Let G be finite, and let X be the set of all vectors $Q = (q_0, q_1, \dots, q_r)$ satisfying Notation 2.2. Then $\{E_{G,r,Q} : Q \in X\}$ is a partition of $E_{G,r}$.

Theorem 2.1. Let G be finite, and let X be the set of all vectors $Q = (q_0, q_1, \dots, q_r)$ satisfying Notation 2.2. Then $\{E_{G,r,Q} : Q \in X\}$ is a partition of $E_{G,r}$.

Proof. Propositions 2.1 and 2.2 are used implicitly throughout this proof.

$$\bigcup_{Q \in X} E_{G,r,Q} = \{e : e' \in \bigcup_{Q \in X} \bar{E}_{G,r,Q}\} = \{e : e' \in E_{G,r}\} = E_{G,r}.$$

Furthermore, if $Q \in X$, then

$$E_{G,r,Q} = \{e' : e \in \bar{E}_{G,r,Q}\} \neq \emptyset.$$

Suppose $P \neq Q$. Then

$$\begin{aligned} E_{G,r,P} \cap E_{G,r,Q} &= \{e : e' \in \bar{E}_{G,r,P}\} \cap \{e : e' \in \bar{E}_{G,r,Q}\} \\ &= \{e : e' \in \bar{E}_{G,r,P} \cap \bar{E}_{G,r,Q}\} \\ &= \emptyset. \end{aligned}$$

Thus $\{E_{G,r,Q} : Q \in X\}$ is a partition of $E_{G,r}$.

Δ^2

If G is finite, then $E_{G,r,Q}$ is a finite set which is totally determined by G , r , and Q . If we consider e to be chosen at random from a uniform distribution on this set, then Q determines the probability density function (or p.d.f.) for the values returned by e . In particular, the following result is true.

Theorem 2.2. Let G be finite. If e is taken from a uniform distribution on $E_{G,r,Q}$, then for every $g \in G$ and $i \in \hat{r}$,

²The symbol " Δ " is used to mark the ends of proofs.

$$\begin{aligned} \Pr [e(g) = i \mid t \text{ is a "+" node}] \\ &= \Pr [e(g) = r-i \mid t \text{ is a "-" node}] \\ &= \frac{q_i}{|N(G)|}. \end{aligned}$$

In order to prove Theorem 2.2, the following lemma is needed.

Lemma 2.1. Let G be finite and $g \in N(G)$. If e is taken from a uniform distribution on $\bar{E}_{G,r,Q}$, then for each $i \in \hat{r}$, $\Pr [e(g) = i] = q_i/|N(G)|$, independent of g .

Proof. $|\bar{E}_{G,r,Q}|$ is equal to the multinomial coefficient

$$\binom{|N(G)|}{q_1 \ q_2 \ \dots \ q_r} = \frac{|N(G)|!}{q_0! \ q_1! \ \dots \ q_r!}.$$

Let $g \in N(G)$. Then

$$|\{e \in \bar{E}_{G,r,Q} : e(g) = i\}| = \frac{(|N(G)|-1)!}{q_0! \ \dots \ q_{i-1}! \ (q_i-1)! \ q_{i+1}! \ \dots \ q_r!},$$

$$\begin{aligned} \text{so } \Pr [e(g) = i \mid e \in \bar{E}_{G,r,Q}] &= \frac{|\{e \in \bar{E}_{G,r,Q} : e(g) = i\}|}{|\bar{E}_{G,r,Q}|} \\ &= \frac{q_i}{|N(G)|}. \end{aligned}$$

△

Proof of Theorem 2.2. According to Proposition 2.1 and Lemma 2.1, if e is taken from a uniform distribution on from $E_{G,r,Q}$, then for every $i \in \hat{r}$, $\Pr [e'(g) = i] = q_i/|N(G)|$, independent of g . But therefore,

$$\begin{aligned}
 \Pr [e(g) = i \mid g \text{ is a "+" node}] \\
 &= \Pr [e'(g) = i \mid g \text{ is a "+" node}] \\
 &= \Pr [e'(g) = i] \\
 &= \frac{q_i}{|N(G)|},
 \end{aligned}$$

and

$$\begin{aligned}
 \Pr [e(g) = r-i \mid g \text{ is a "-" node}] \\
 &= \Pr [e'(g) = i \mid g \text{ is a "-" node}] \\
 &= \Pr [e'(g) = i] \\
 &= \frac{q_i}{|N(G)|}.
 \end{aligned}$$

△

According to Theorem 2.2, the value of $e(g)$ if g is a "+" node has the discrete p.d.f.

$$\begin{aligned}
 f(x) &= \frac{q_x}{|N(G)|} && \text{if } x \in \mathfrak{f}, \\
 &= 0 && \text{elsewhere.}
 \end{aligned}$$

and the value of $e(g)$ if g is a "-" node has the discrete p.d.f.

$$\begin{aligned}
 \bar{f}(x) &= \frac{q_{r-x}}{|N(G)|} && \text{if } x \in \mathfrak{f}, \\
 &= 0 && \text{elsewhere.}
 \end{aligned}$$

The nonzero values of f and \bar{f} may be represented by the vector

$$\begin{aligned}
 P &= (f(0), f(1), \dots, f(r)) \\
 &= (\bar{f}(r), \bar{f}(r-1), \dots, \bar{f}(0)) \\
 &= Q/|N(G)|.
 \end{aligned}$$

For example, let $E_{G,r,Q}$ be as in Table 2.2, and let e be a randomly chosen member of $E_{G,r,Q}$. If $g \in N(G)$ is a "+" node, then $\Pr[e(g)=0] = 2/80$, $\Pr[e(g)=1] = 2/80$, $\Pr[e(g)=2] = 7/80$, $\Pr[e(g)=3] = 23/80$, and $\Pr[e(g)=4] = 46/80$. If g is a "-" node, then $\Pr[e(g)=0] = 46/80$, $\Pr[e(g)=1] = 23/80$, $\Pr[e(g)=2] = 7/80$,

$\Pr[e(g)=3] = 2/80$, and $\Pr[e(g)=4] = 2/80$. Thus

$$P = (2/80, 2/80, 7/80, 23/80, 46/80).$$

The point of the last several pages is that our endeavor to find a natural way to classify evaluation functions in terms of their "goodness" has led to a set of equivalence classes, each of which is characterized by a distinct vector $P = (p_0, p_1, \dots, p_r)$ in such a way that if e is a random member of the class, then

$$(2.1) \quad \Pr [e(g) = i] = p_i \quad \text{if } i \text{ is a "+" node,} \\ = p_{r-i} \quad \text{if } i \text{ is a "-" node.}$$

If p_i is large for i large and small for i small, then the equivalence class contains "good" evaluation functions, and if p_i is small for i large and large for i small, then it contains "bad" functions.

So far, we have only considered finite games, because the equivalence classes $E_{G,r,Q}$ cannot be constructed when G is infinite. However, the property of equation (2.1) can easily be extended to infinite games, by means of the following definitions.

Definition 2.7. A probability vector is a vector $P = (p_0, p_1, \dots, p_r)$, where $0 \leq p_i \leq 1$ for each i , and where $p_0 + p_1 + \dots + p_r = 1$.

Definition 2.8. If P is a probability vector, the p.d.f. determined by P is the discrete p.d.f.

$$f(i) = p_i \quad \text{if } i \in \hat{P}, \\ = 0 \quad \text{otherwise.}$$

If f is the p.d.f. for some random variable X , then P is called the probability vector for X .

Definition 2.9. If $P = (p_0, p_1, \dots, p_r)$, then $\text{rev } P = (p_r, p_{r-1}, \dots, p_0)$. Note that $P = \text{rev rev } P$.

Definition 2.10. An \hat{P} -vector is a probability vector $P = (p_0, p_1, \dots, p_r)$.

Notation 2.3. Except where stated otherwise, P denotes an \hat{P} -vector.

Definition 2.11. Let P be an \hat{P} -vector. A P -function on a game G is an evaluation function e which associates with each $g \in N(G)$ an independent random variable $e(g)$ from the p.d.f. determined by P if g is a "+" node, or from the p.d.f. determined by $\text{rev } P$ if g is a "-" node. If e is a P -function, then P is the probability vector for e .

Readers may wonder what advantage there is to introducing the definition of a probability vector, when it appears to be identical in its properties to a discrete p.d.f. The answer is that a probability vector carries an additional piece of information: the number of entries in the vector. For example, if we are given a probability vector $P = (p_0, p_1, \dots, p_r)$ for an evaluation function e , then we know that the probability vector for $e(g)$ is P if g is a "+" node and $\text{rev } P$ if g is a "-" node. If we are given only the p.d.f. f determined by P , then we know that the p.d.f. for $e(g)$ is f if g is a "+" node, but we do not know the p.d.f. for $e(g)$ when g is a "-" node unless we are also given the value of r .

Choosing a Correct Move

A typical technique for attempting to minimize the effects of errors in evaluation functions is to try to predict

what may happen in the future as a result of each of the possible moves one can make. It is not well understood how humans do this, but in the case of computer programs, it is done by searching a game tree to some arbitrary depth, applying the evaluation function to the nodes at this depth, and propagating the values back to the children of the node representing the current position, by means of a procedure called minimaxing.

As pointed out in Chapter 1, the the term "minimaxing" is used here in a somewhat different sense than in the minimax theorem of mathematical game theory. Perhaps the most significant difference is that here it refers to operations that are performed on estimated consequences of various decisions, rather than on known payoffs. As an illustration, let us return to the example used at the beginning of the previous section:

If I move to position A, then my opponent will probably move to position B, which will leave me in a bad position. However, if I move to position C, then my opponent will not have any good moves. Therefore, I'll move to position C.

The player who did this reasoning made his decision using what is known in computer science as a depth 2 minimax search (see Chapter 1).

Suppose it was Max who did the above reasoning. If Max had done a depth 0 search, he would simply have chosen a move at random. If he had done a depth 1 search, he would have looked only at positions A and C, and chosen whichever one looked better to him. But instead, he looked one level deeper, in an attempt to predict what Min's response would be. Max expected Min to make whichever reply was worst for Max, according to Max's estimates of the values of the positions. Therefore, he chose a

move which minimized the damage he thought could be done by Min. Had he been explicitly using an evaluation function, the value Max would have assigned to position A would have been the minimum of the evaluation function values for each of Min's possible moves from position A. This number is called the depth 1 minimax value for A.

Minimax values are defined formally in Definition 2.13 below. For $d > 0$, 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 highest value. 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.

Definition 2.12. If g is a node with n children, then the children are $c_1(g)$, $c_2(g)$, ..., $c_n(g)$.

Notation 2.4. Except where otherwise stated, d is a nonnegative integer which denotes a search depth.

Definition 2.13. If $e \in E_{G,r}$ and $g \in N(G)$, then the depth d minimax value for g using e is

$$\begin{aligned}
 e_d(g) &= e(g) && \text{if } d = 0 \text{ or } g \text{ is a leaf,} \\
 &= \max \{e_{d-1}(g') : g' \text{ is a child of } g\} && \text{if } g \text{ is a max node which is not a leaf,} \\
 &= \min \{e_{d-1}(g') : g' \text{ is a child of } g\} && \text{if } g \text{ is a min node which is not a leaf.}
 \end{aligned}$$

As an example, suppose Max is choosing a move using a depth 3 minimax search on the tree of Figure 2.3. Suppose that the evaluation function returns the values 10, 2, 3, 6, 11, 9, 0, 14, 8, 5, and 6 on the nodes at depth 3 of the tree. Then by

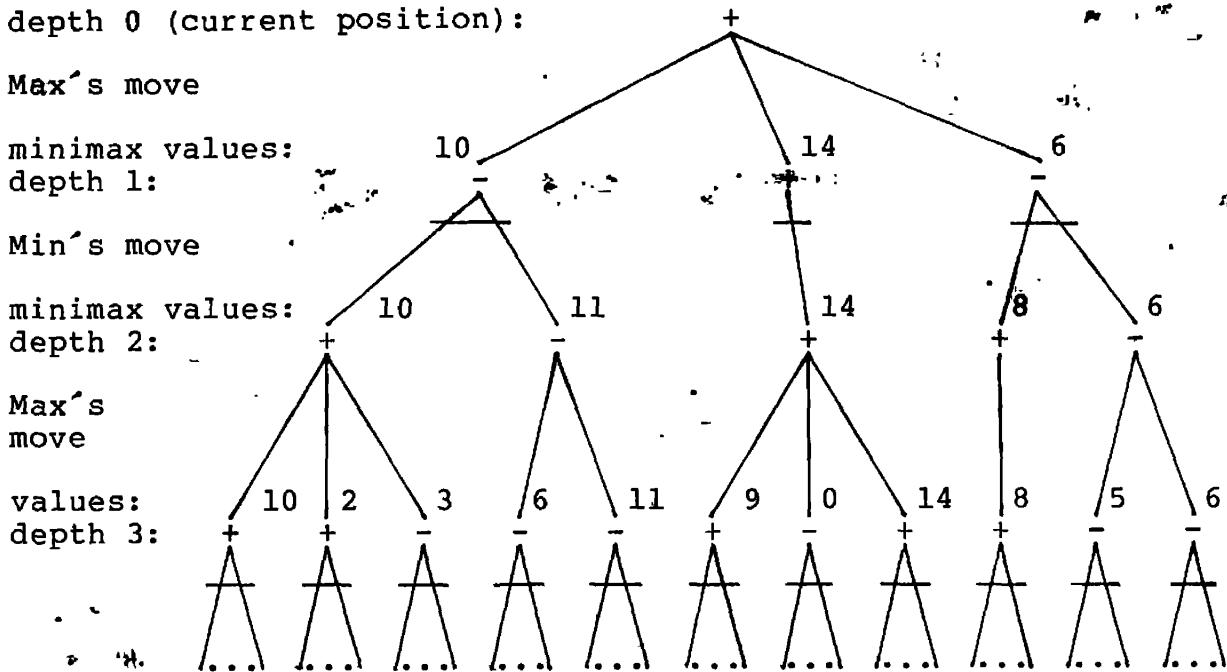


FIGURE 2.3.--Minimax values computed using a depth 3 minimax search on the game tree of Figure 2.1. The values are computed as described in the text.

Definition 2.13, the depth 0 minimax values for these nodes are these same values, as indicated in the figure. The figure also gives the depth 1 minimax values for the nodes at depth 2 and the depth 2 minimax values for the nodes at depth 1, as computed according to Definition 2.13. Based on these values, Max would choose to move to the depth 1 node having the value 14.

If G is a game tree, then the only nodes of G for which it matters which move is chosen are the critical nodes; i.e., the S and U nodes. Obviously, each player is trying to choose a node leading to a forced win for himself. Thus, if g is a critical node, a correct move from g is a move to a "+" child of g if g is a max node, or to a "-" child of g if g is a min node. For

example, in the game tree of Figure 2.3, the only correct choice for Max is the node having the value 14.

In the game tree of Figure 2.3, suppose that the third depth 1 node had received the value 15 rather than 6. Then Max would have chosen this node instead. Since the node is a "-" node, this decision would have been incorrect. If this node had received the value 14, then Max would have chosen at random between it and the second depth 1 node, and thus would have made a correct decision with probability 1/2.

If e is a P-function, then the errors it makes are assumed by Definition 2.11 to be stochastically independent, and the probability of correct decision depends on the probability vectors for the random variables $\{e_{d-1}(g') : g' \text{ is a child of } g\}$. Based on the following definitions, Theorem 2.3 gives a formula for the probability of choosing the correct move at a critical node in a game.

Definition 2.14. $D(G, g, P, d)$ denotes the probability of making a correct decision at a critical node $g \in G$ using a depth d minimax search and a P-function e .

Definition 2.15. If e is a P-function on a game G and $g \in N(G)$, then the probability vector for $e_d(g)$ is called $P_{g,d}^G$.

Definition 2.16. Let X_1, X_2, \dots, X_n be random variables from probability distribution functions determined by the \hat{f} -vectors P_1, P_2, \dots, P_n , respectively. Then the probability vectors for

$\max(X_1, X_2, \dots, X_n)$ and $\min(X_1, X_2, \dots, X_n)$
are called

$\maxp(P_1, P_2, \dots, P_n)$ and $\minp(P_1, P_2, \dots, P_n)$,

respectively.

Definitions 2.13, 2.15, and 2.16 immediately give the following result.

Proposition 2.3. If $g \in N(G)$ is not a leaf, e is a P-function, and $d \geq 0$ is an integer, then

$$\begin{aligned} P_{g,d+1}^G &= \max_p \{P_{g',d}^G : g' \text{ is a child of } g\} && \text{if } g \text{ is a "+" node,} \\ &= \min_p \{P_{g',d}^G : g' \text{ is a child of } g\} && \text{if } g \text{ is a "-" node.} \end{aligned}$$

Theorem 2.3. Let g be a critical nonleaf node of G . Let m be the number of children of g having the same sign ("+" or "-") as g , and n be the number of children of opposite sign. Let e be a P-function on G , and let

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

Then

$$\begin{aligned} D(G,g,P,d) &= \frac{m}{m+n} && \text{if } d = 0, \\ &= \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r \text{Pr} [I=i \text{ and } J=j \mid H=k] && \text{if } d > 0, \end{aligned}$$

where I is the number of children g' of g having the same sign as g such that $e_{d-1}(g') = H$, and J is the number of children g'' of g having the opposite sign from g such that $e_{d-1}(g'') = H$.

Proof. Suppose that g is a max node (i.e., that Max is choosing the move). If $d = 0$, then Max is not searching at all, so his choice is at random from among the children of g . Thus since there are $m+n$ children of g and m of them are "+" nodes,

$$D(G,g,P,0) = \frac{m}{m+n}.$$

If $d \geq 1$, then choosing a move using a depth d search means computing the values $e_{d-1}(c_i(g))$ for $i = 1, 2, \dots, m+n$, and

choosing whichever of $c_1(g), c_2(g), \dots, c_{m+n}(g)$ receives the value H . If $I+J$ of the nodes get this value, the move must be chosen at random from among them. The choice will be a correct one if and only if the move is to a "+" node. Thus, given I and J , the probability of correct decision is $\frac{I}{I+J}$. But H varies randomly over \hat{r} , and I and J vary randomly from 0 to m and 0 to n , respectively. Thus

$$\begin{aligned} D(G,g,P,d) &= \sum_{i=0}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r \Pr [I=i \text{ and } J=j \mid H=k] \\ &= \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r \Pr [I=i \text{ and } J=j \mid H=k]. \end{aligned}$$

The proof is similar if g is a min node.

△

Obviously, determining $D(G,g,P,d)$ for an arbitrary node g of an arbitrary game G can be very complicated. However, for certain kinds of games, the task is more reasonable. Some of these games are discussed in the next chapter.

There is one situation in which $D(G,g,P,d)$ is quite simply determined. That is when $P = (p_0, p_1, \dots, p_r)$ is such that $p_i = 0$ for $i \leq r/2$ (i.e., all entries in the left-hand half of P are zero). In that case, $e(g_1) > e(g_2)$ whenever g_1 is a "+" node and g_2 is a "-" node, so $D(G,g,P,d) = 1$, independent of d .

Definition 2.17. If the probability vector P is as in the above paragraph, then P is perfect. Otherwise, P is imperfect.

Since a perfect probability vector defines a class of perfect evaluation functions, one might suspect that such vectors are not very numerous. This is indeed the case. From

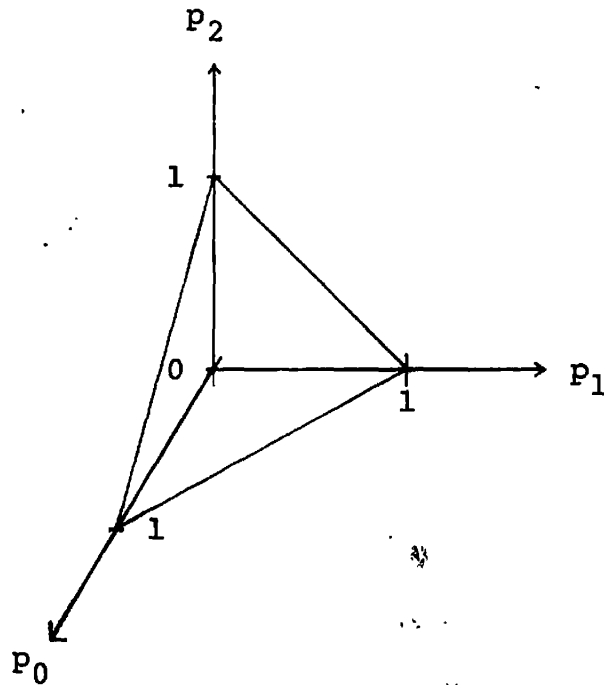


FIGURE 2.4.--The set of all probability vectors (p_0, p_1, p_2) forms a triangular plane segment. The set of all perfect probability vectors (p_0, p_1, p_2) consists of the single point $(0, 0, 1)$.

Definitions 2.7 and 2.10, it follows that the set of all \hat{f} -vectors forms an r -dimensional simplex (for example, see Figure 2.4). Since the $[r/2]+1$ leftmost entries of a perfect \hat{f} -vector must be zero,³ the set of perfect \hat{f} -vectors forms a $[(r-1)/2]$ -dimensional simplex. But the second simplex is of measure zero in the first one. This proves the following.

Proposition 2.4. Almost all \hat{f} -vectors are imperfect.⁴

³The square brackets are used here to denote the "greatest integer" or "floor" function.

⁴For those unfamiliar with this term, a property is said to hold for almost all points in a set if it holds everywhere but on a subset of measure zero [JA2].

This means that the probability of choosing a perfect \hat{f} -vector from any continuous p.d.f. over the set of all \hat{f} -vectors is 0. This result partially explains why it is so difficult to find perfect evaluation functions for games that are too large to analyze exhaustively!

CHAPTER 3

"HOMOGENEOUS" GAMES

The Basic Definition

According to the proof of Theorem 2.3, the probability of correct decision at any critical node g depends on the probability vectors for the minimax values of g 's children. On most game trees, these vectors depend heavily on what the game tree looks like below g , and so it is very difficult to make any general statements about how the probability of correct decision varies with search depth. 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 $\Gamma_x(m,n)$ is the unique infinite game tree such that--

1. the root of $\Gamma_x(m,n)$ is an x node;
2. every critical node in $\Gamma_x(m,n)$ has m children of the same sign and n children of opposite sign;
3. every node of $\Gamma_x(m,n)$ has $m+n$ children.

$\Gamma_S(m,n)$ is illustrated in Figure 3.1.

Notation 3.1. Most of the results to follow are independent of which of $\Gamma_S(m,n), \Gamma_T(m,n), \Gamma_U(m,n),$ and $\Gamma_V(m,n)$ is being considered. Therefore, $\Gamma(m,n)$ denotes any one of these four trees.

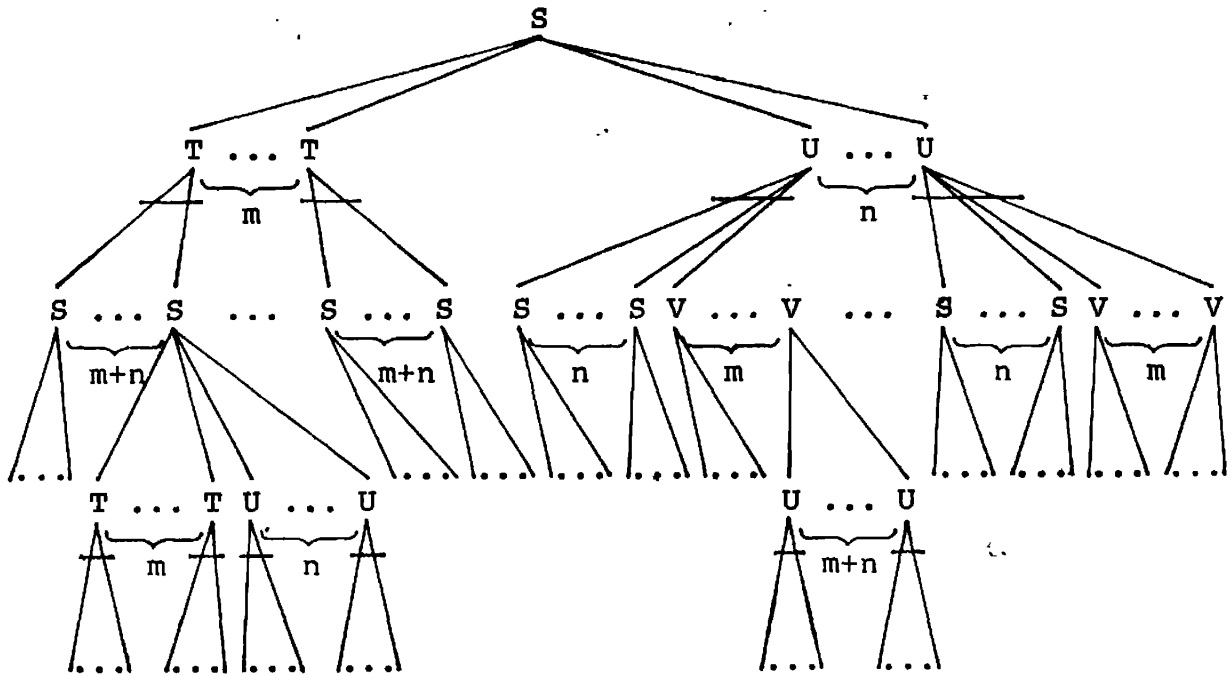


FIGURE 3.1.--The tree $\Gamma_S(m,n)$.

Truncation

$\Gamma(m,n)$ has a number of interesting properties which are explored in the next few chapters. The relevance of these properties to finite games is that infinitely many finite games may be formed from $\Gamma(m,n)$ simply by truncating it in whatever way is desired. This is formalized below.

Definition 3.2. The truncation of a tree G is any tree G' such that--

1. every node of G' is a node of G ;
2. the root of G' is the root of G ;
3. for every nonleaf node g of G' , the children of g in G' are precisely the children of g in G .

Definition 3.3. An (m,n) -game is a finite truncation of $\Gamma(m,n)$.

Proposition 3.1. Let $d \geq 0$ be an integer, G be an (m,n) -game, and $g \in N(G)$. If the subtree of G having g as its root is complete to at least depth d , then the probability vector for $e_d(g)$ in G is the same as the probability vector for $e_d(g)$ in $\Gamma(m,n)$, and $D(G,g,P,d) = D(\Gamma(m,n),g,P,d)$.

Proposition 3.1 allows us to prove results about $\Gamma(m,n)$ with the assurance that they also apply to all sufficiently large (m,n) -games.

Independence Results

Some of the nicer properties of $\Gamma(m,n)$ are--

1. the probability vector for any S node g of $\Gamma(m,n)$ is independent of g , and similarly for the T, U, and V nodes;
2. the probability of correct decision using any P-function and searching to depth d depends solely on P and d .

These properties are proved in Theorem 3.1 and Corollary 3.2.1 below.

Definition 3.4. Let P and Q be \hat{f} -vectors. Then

$$\max_{P,Q} = \max(P, \dots, P, Q, \dots, Q)$$

and
$$\min_{P,Q} = \min(P, \dots, P, Q, \dots, Q),$$

where in each case there are m occurrences of P and n occurrences of Q .

Lemma 3.1. Let P_1, P_2, \dots, P_n be \hat{f} -vectors. Then

$$(3.1) \quad \max(P_1, P_2, \dots, P_n) = \text{rev} \min(\text{rev } P_1, \text{rev } P_2, \dots, \text{rev } P_n)$$

and

$$(3.2) \quad \min(P_1, P_2, \dots, P_n) = \text{rev} \max(\text{rev } P_1, \text{rev } P_2, \dots, \text{rev } P_n).$$

Proof. We prove equation (3.1); the proof of equation (3.2) is similar.

Let X_1, X_2, \dots, X_n be random variables from the discrete probability density functions f_1, f_2, \dots, f_n determined by P_1, P_2, \dots, P_n , respectively, and let $X = \max(X_1, X_2, \dots, X_n)$. Then from Definition 2.16, the probability vector for X is $\maxp(P_1, P_2, \dots, P_n)$.

For $i = 1, 2, \dots, n$, let $Y_i = r - X_i$. Then the p.d.f. for Y_i is $g_i(x) = f_i(r - x)$, so the probability vector for Y_i is $\text{rev } P_i$. Let $Y = \min(Y_1, Y_2, \dots, Y_n)$. Then from Definition 2.16, the probability vector for Y is $\text{minp}(\text{rev } P_1, \text{rev } P_2, \dots, \text{rev } P_n)$, so the probability vector for $r - Y$ is

$$\text{rev minp}(\text{rev } P_1, \text{rev } P_2, \dots, \text{rev } P_n).$$

But for all real x ,

$$\begin{aligned} \Pr [X \leq x] &= \Pr [\max(X_1, X_2, \dots, X_n) \leq x] \\ &= \Pr [\min(Y_1, Y_2, \dots, Y_n) \geq r - x] \\ &= \Pr [Y \geq r - x] \\ &= \Pr [r - Y \leq x]. \end{aligned}$$

Thus X and $r - Y$ have the same p.d.f., and hence the same probability vector.

△

Theorem 3.1. Let s and s' , t and t' , u and u' , and v and v' be distinct $S, T, U,$ and V nodes, respectively, in $\Gamma(m, n)$, and let $m \geq 1$ and $n \geq 1$ be integers. Then for every integer $d \geq 0$,

$$(3.3) \quad P_{s,d}^{\Gamma(m,n)} = P_{s',d}^{\Gamma(m,n)} = \text{rev } P_{u,d}^{\Gamma(m,n)} = \text{rev } P_{u',d}^{\Gamma(m,n)}$$

and

$$(3.4) \quad P_{t,d}^{\Gamma(m,n)} = P_{t',d}^{\Gamma(m,n)} = \text{rev } P_{v,d}^{\Gamma(m,n)} = \text{rev } P_{v',d}^{\Gamma(m,n)}.$$

Proof (by induction on d). From Definitions 2.11, 2.13, and 2.15,

$$\begin{aligned} P_{s,0}^{\Gamma(m,n)} &= P_{s',0}^{\Gamma(m,n)} = P, & P_{t,0}^{\Gamma(m,n)} &= P_{t',0}^{\Gamma(m,n)} = P, \\ P_{u,0}^{\Gamma(m,n)} &= P_{u',0}^{\Gamma(m,n)} = \text{rev } P, & P_{v,0}^{\Gamma(m,n)} &= P_{v',0}^{\Gamma(m,n)} = \text{rev } P. \end{aligned}$$

Thus the theorem holds for $d = 0$.

Suppose the theorem holds for $d = k$. Then for every T child g of s or s' , $P_{g,k}^{\Gamma(m,n)} = P_{t,k}^{\Gamma(m,n)}$, and for every U child g' of s or s' , $P_{g',k}^{\Gamma(m,n)} = P_{u,k}^{\Gamma(m,n)}$. But s and s' each have m T children and n U children, so from Proposition 2.3 and Definition 3.4,

$$P_{s,k+1}^{\Gamma(m,n)} = P_{s',k+1}^{\Gamma(m,n)} = \max_{P_{m,n}} (P_{t,k}^{\Gamma(m,n)}, P_{u,k}^{\Gamma(m,n)}).$$

Similarly (but also using Definition 2.9),

$$P_{u,k+1}^{\Gamma(m,n)} = P_{u',k+1}^{\Gamma(m,n)} = \min_{P_{m,n}} (\text{rev } P_{t,k}^{\Gamma(m,n)}, \text{rev } P_{u,k}^{\Gamma(m,n)}),$$

whence from Definitions 2.9 and 3.4 and Lemma 3.1,

$$\begin{aligned} \text{rev } P_{u,k+1}^{\Gamma(m,n)} &= \text{rev } P_{u',k+1}^{\Gamma(m,n)} = \max_{P_{m,n}} (P_{t,k}^{\Gamma(m,n)}, P_{u,k}^{\Gamma(m,n)}) \\ &= P_{s,k+1}^{\Gamma(m,n)} = P_{s',k+1}^{\Gamma(m,n)}. \end{aligned}$$

Thus equation (3.3) holds for $d = k+1$. The proof for equation (3.4) is similar.

△

Notation 3.2. Let $s, t, u,$ and v be any S, T, U, and V nodes of $\Gamma(m,n)$, respectively. Then Theorem 3.1 allows us to define

$$\begin{aligned} S_d^{m,n} &= P_{s,d}^{\Gamma(m,n)}, & T_d^{m,n} &= P_{t,d}^{\Gamma(m,n)}, \\ U_d^{m,n} &= P_{u,d}^{\Gamma(m,n)}, & \text{and } V_d^{m,n} &= P_{v,d}^{\Gamma(m,n)}. \end{aligned}$$

When the superscripts m and n are obvious, they are omitted.

Corollary 3.1.1.

$S_0 = T_0 = P$ and $U_0 = V_0 = \text{rev } P$,
and for every integer $d \geq 0$,

$$S_d = \text{rev } U_d \quad \text{and} \quad T_d = \text{rev } V_d.$$

Proof. This is merely a restatement of Theorem 3.1 and the first few lines of its proof using Notation 3.2.

△

Corollary 3.1.2. For every integer $d \geq 0$,

$$S_{d+1} = \max_{P_{m,n}}(T_d, U_d), \quad T_{d+1} = \min_{P_{m,n}}(S_d, S_d),$$

$$U_{d+1} = \min_{P_{m,n}}(V_d, S_d), \quad \text{and} \quad V_{d+1} = \max_{P_{m,n}}(U_d, U_d).$$

Proof. Immediate from Notation 3.2 and the proof of Theorem 3.1.

△

Definition 3.5. If $P = (p_0, p_1, \dots, p_r)$ is an \hat{r} -vector and $i \in \hat{r}$, then $(P)_i = p_i$ and $(P)_{i,j} = \sum_{k=i}^j p_k$. (Thus if $i > j$ then $(P)_{i,j} = 0$.)

The probability of correct decision at a critical node g of a game tree using a depth d search depends on the probability vectors for the depth $d-1$ minimax values of the children of g . Theorem 3.2 gives a formula for computing the probability of correct decision on $\overline{\Gamma}(m,n)$ in terms of these values.

Theorem 3.2. Let g be a critical node of $\overline{\Gamma}(m,n)$. If P is an \hat{r} -vector for $\overline{\Gamma}(m,n)$ and $d > 0$ is an integer, then

$$D(\overline{\Gamma}(m,n), g, P, d) = \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r A(m, i, k, T_{d-1}) A(n, j, k, U_{d-1})$$

$$= \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r B(m, i, k, V_{d-1}) B(n, j, k, S_{d-1}),$$

where $A(h,i,k,Q) = \binom{h}{i} ((Q)_k)^i ((Q)_{0,k-1})^{h-i}$
 and $B(h,i,k,Q) = \binom{h}{i} ((Q)_k)^i ((Q)_{k+1,r})^{h-i}$,
 and where 0^0 is taken to be 1.

Proof. Suppose g is an S node, and let H be as in Theorem 2.3. If g' is a "+" child of g , then g' is a T node, so from Notation 3.2 and Definition 3.5,

$$\Pr [e_{d-1}(g') = k] = (T_{d-1})_k$$

and $\Pr [e_{d-1}(g') < k] = (T_{d-1})_{0,k-1}$

for every $k \in \hat{r}$. Let I be the number of "+" children g' of g such that $e_{d-1}(g') = H$. For all other "+" children g'' of g , $e_{d-1}(g'') < H$. But g has m "+" children. Therefore, for each $i \in \hat{r}$,

$$\begin{aligned} \Pr [I=i \mid H=k] &= \binom{m}{i} ((T_{d-1})_k)^i ((T_{d-1})_{0,k-1})^{m-i} \\ &= A(m,i,k,T_{d-1}), \end{aligned}$$

where 0^0 is taken to be 1. Similarly, for each $j \in \hat{r}$,

$$\begin{aligned} \Pr [J=j \mid H=k] &= \binom{n}{j} ((U_{d-1})_k)^j ((U_{d-1})_{0,k-1})^{n-j} \\ &= A(n,j,k,U_{d-1}), \end{aligned}$$

where J is the number of "-" children g' of g such that $e_{d-1}(g') = H$ and 0^0 is taken to be 1. But according to Definition 2.11, I and J are stochastically independent. Therefore, for each $i \in \hat{r}$ and $j \in \hat{r}$,

$$\Pr [I=i \text{ and } J=j \mid H=k] = A(m,i,k,T_{d-1}) A(n,j,k,U_{d-1}),$$

so by Theorem 2.3,

$$\begin{aligned} (3.5) \quad D(\Gamma^*(m,n), g, P, d) &= \sum_{i=1}^m \sum_{j=0}^n \frac{1}{i+j} \sum_{k=0}^r A(m,i,k,T_{d-1}) A(n,j,k,U_{d-1}). \end{aligned}$$

Suppose g is a U node. Then by a similar argument, we get

$$(3.6) \quad D(\Gamma^*(m,n), g, P, d)$$

$$= \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r B(m, i, k, V_{d-1}) B(n, j, k, S_{d-1}).$$

But from Corollary 3.1.1 and Definitions 2.9 and 3.5,

$$\begin{aligned} A(m, i, k, T_{d-1}) &= \binom{m}{i} ((T_{d-1})_k)^i ((T_{d-1})_{0, k-1})^{m-i} \\ &= \binom{m}{i} ((\text{rev } T_{d-1})_{r-k})^i ((\text{rev } T_{d-1})_{r-k+1, r})^{m-i} \\ &= \binom{m}{i} ((V_{d-1})_{r-k})^i ((V_{d-1})_{r-k+1, r})^{m-i} \\ &= B(m, i, r-k, V_{d-1}), \end{aligned}$$

and similarly, $A(n, j, k, U_{d-1}) = B(n, j, r-k, S_{d-1})$. Thus

$$\sum_{k=0}^r A(m, i, k, T_{d-1}) A(n, j, k, U_{d-1}) = \sum_{k=0}^r B(m, i, k, V_{d-1}) B(n, j, k, S_{d-1}),$$

so the quantities in equations (3.5) and (3.6) are equal.

△

Corollary 3.2.1. Let g and g' be any two critical nodes of $\Gamma^*(m, n)$. Then $D(\Gamma^*(m, n), g, P, d) = D(\Gamma^*(m, n), g', P, d)$.

Proof. For $d = 0$, the proof is immediate from Theorem 2.3. For $d > 0$, the proof is immediate from Theorem 3.2.

△

Notation 3.3. Corollary 3.2.1 allows us to denote $D(\Gamma^*(m, n), g, P, d)$ by $D_{m, n}(P, d)$. When the values of m and n are obvious, this quantity is further abbreviated as $D(P, d)$.

CHAPTER 4

A PATHOLOGICAL GAME

Introduction

This chapter is devoted to showing that the game tree $\Gamma(1,1)$ has a property called pathology. The meaning of this term is the following. Let P be imperfect, and suppose someone is using a P -function to choose moves in the game $\Gamma(1,1)$. Contrary to what one might expect, increasing the search depth does not improve the quality of play. Instead, as the search depth is increased, the probability of choosing a correct move approaches 0.5. This is exactly the same probability of correct decision one would obtain in this game by choosing moves totally at random.

In Chapter 6, this result is generalized to all of the $\Gamma(m,n)$. The relevance of this to finite games is the following: since every (m,n) -game is a truncation of $\Gamma(m,n)$, the same kind of result holds for every (m,n) -game as long as the search does not reach the end of the game tree. At the search depths that are used in many real games, this does not occur until very late in the game, at which point one of the players often has a strong advantage.

In order to prove that $\Gamma(1,1)$ is pathological, a number of preliminary theorems are needed. The following two sections contain these results. Following them is a section containing the formal definition of pathology and a proof that $\Gamma(1,1)$ is

pathological.

The final section is a discussion of relevant experimental results.

Theorems About Probability Vectors

The following theorems provide ways to compute the functions \max_p and \min_p . They are stated in a more general form than is necessary for this chapter, so that they will be useful in later chapters as well.

Theorem 4.1. Let P_1, P_2, \dots, P_n be \hat{r} -vectors, and let $P = \max_p (P_0, P_1, \dots, P_n)$ and $Q = \min_p (P_0, P_1, \dots, P_n)$. Then for each $i \in \hat{r}$,

$$(4.1) \quad (P)_{0,i} = \prod_{j=1}^n (P_j)_{0,i} \quad \text{and} \quad (P)_{i,r} = 1 - \prod_{i=1}^n (1 - (P_j)_{i,r}),$$

and

$$(4.2) \quad (Q)_{i,r} = \prod_{j=1}^n (P_j)_{i,r} \quad \text{and} \quad (Q)_{0,i} = 1 - \prod_{j=1}^n (1 - (P_j)_{0,i}).$$

Proof. For $i = 1, 2, \dots, n$, let Z_i be a random variable from the p.d.f. determined by P_i , and let $Z = \max (Z_1, Z_2, \dots, Z_n)$. Then from Definitions 2.16 and 3.5,

$$(P)_{0,i} = \Pr [Z \leq i] = \prod_{j=1}^n \Pr [Z_j \leq i] = \prod_{j=1}^n (P_j)_{0,i}.$$

Thus from Definition 3.5, for each $i \in \hat{r}$,

$$(P)_{i,r} = 1 - (P)_{0,i-1} = 1 - \prod_{j=1}^n (P_j)_{0,i-1} = 1 - \prod_{j=1}^n (1 - (P_j)_{i,r}).$$

This proves statement (4.1); the proof is similar for statement (4.2).

△

As an example, let $P_1 = (0.1, 0.2, 0.3, 0.4)$ and $P_2 = (0.25, 0.25, 0.25, 0.25)$. Then

$$(P)_{0,0} = (.1) (.25) = 0.025,$$

$$(P)_{0,1} = (.1+.2) (.25+.25) = 0.150,$$

$$(P)_{0,2} = (.1+.2+.3) (.25+.25+.25) = 0.450,$$

and $(P)_{0,3} = (.1+.2+.3+.4) (.25+.25+.25+.25) = 1.000$.

Corollary 4.1.1. If P and Q are as in Theorem 4.1, then for each $i \in \hat{r}$,

$$(P)_i = \prod_{j=1}^n (P_j)_{0,i} - \prod_{j=1}^n (P_j)_{0,i-1}$$

and $(Q)_i = \prod_{j=1}^n (P_j)_{i,r} - \prod_{j=1}^n (P_j)_{i+1,r}$.

Proof. Immediate from Theorem 4.1.

△

To continue the previous example,

$$(P)_0 = 0.025 - 0.000 = 0.025,$$

$$(P)_1 = 0.150 - 0.025 = 0.125,$$

$$(P)_2 = 0.450 - 0.150 = 0.300,$$

and $(P)_3 = 1.000 - 0.450 = 0.550$,

so $P = (0.025, 0.125, 0.300, 0.550)$.

Corollary 4.1.2. As a special case of Corollary 4.1.1, if $n = 2$, then

$$(P)_i = (P_1)_i (P_2)_{0,i-1} + (P_2)_i (P_1)_{0,i}$$

and $(Q)_i = (P_1)_i (P_2)_{i+1,r} + (P_2)_i (P_1)_{i,r}$.

Proof. The proof follows by simple algebraic manipulation of the equations of Corollary 4.1.1. The details are left to the reader.

△

To continue the example again,

$$(P)_0 = 0.1 (0) + 0.25 (.1) = 0.25,$$

$$(P)_1 = 0.2 (.25) + 0.25 (.1+.2) = 0.125,$$

$$(P)_2 = 0.3 (.25+.25) + 0.25 (.1+.2+.3) = 0.300,$$

and $(P)_3 = 0.4 (.25+.25+.25) + 0.25 (.1+.2+.3+.4) = 0.550,$

so as before, $P = (0.025, 0.125, 0.300, 0.550).$

Corollary 4.1.3. If $P = \max_{m,n}(P_1, P_2)$, then

$$(P)_{0,i} = ((P_1)_{0,i})^m ((P_2)_{0,i})^n$$

and $(P)_{i,r} = 1 - (1 - ((P_1)_{i,r})^m) (1 - ((P_2)_{i,r})^n).$

If $P = \min_{m,n}(P_1, P_2)$, then

$$(P)_{i,r} = ((P_1)_{i,r})^m ((P_2)_{i,r})^n$$

and $(P)_{0,i} = 1 - (1 - ((P_1)_{0,i})^m) (1 - ((P_2)_{0,i})^n).$

Proof. Immediate from Definition 3.4 and Theorem 4.1.

△

Corollary 4.1.4. Let P be an \hat{f} -vector, $m \geq 1$ and $n \geq 1$

be integers, and $S_d, T_d, U_d,$ and V_d be as in Notation 3.2. If $i \in \hat{f}$, then--

1. $(S_{d+1})_{0,i} = ((T_d)_{0,i})^m ((U_d)_{0,i})^n;$
2. $(S_{d+1})_{i,r} = 1 - (1 - (T_d)_{i,r})^m (1 - (U_d)_{i,r})^n;$
3. $(T_{d+1})_{0,i} = 1 - (1 - (S_d)_{0,i})^{m+n};$
4. $(T_{d+1})_{i,r} = ((S_d)_{i,r})^{m+n};$
5. $(U_{d+1})_{0,i} = 1 - (1 - (V_d)_{0,i})^m (1 - (S_d)_{0,i})^n;$
6. $(U_{d+1})_{i,r} = ((V_d)_{i,r})^m ((S_d)_{i,r})^n;$
7. $(V_{d+1})_{0,i} = ((U_d)_{0,i})^{m+n};$
8. $(V_{d+1})_{i,r} = 1 - (1 - (U_d)_{i,r})^{m+n}.$

Proof. Immediate from Corollary 3.1.2 and Corollary 4.1.3.

△

Corollary 4.1.5. Let P be an \hat{f} -vector, $m \geq 1$ and $n \geq 1$ be integers, and $S_d, T_d, U_d,$ and V_d be as in Notation 3.2. If $i \in \hat{f}$, then--

1. $(S_{d+2})_{0,i} = (1 - (1 - (S_d)_{0,i})^{m+n})^m (1 - (1 - (V_d)_{0,i})^m (1 - (S_d)_{0,i})^n)^n;$
2. $(T_{d+2})_{i,r} = (1 - (1 - (T_d)_{i,r})^m (1 - (U_d)_{i,r})^n)^{m+n};$
3. $(U_{d+2})_{i,r} = (1 - (1 - (U_d)_{i,r})^{m+n})^m (1 - (1 - (T_d)_{i,r})^m (1 - (U_d)_{i,r})^n)^n;$
4. $(V_{d+2})_{0,i} = (1 - (1 - (V_d)_{0,i})^m (1 - (S_d)_{0,i})^n)^{m+n}.$

Proof. Immediate from Corollary 4.1.4.

△

Corollary 4.1.6. As a special case of Corollary 4.1.5, if $m = 1$ and $n = 1$, then

$$(T_{d+2})_{i,r} = (1 - (1 - (T_d)_{i,r}) (1 - (U_d)_{i,r}))^2$$

and $(U_{d+2})_{i,r} = (1 - (1 - (U_d)_{i,r})^2) (1 - (1 - (T_d)_{i,r}) (1 - (U_d)_{i,r})).$

Proof. Immediate from Corollary 4.1.5.

△

Theorems About Sequences

The following theorem about functions and sequences is of major importance in proving the results in the next several chapters.

Notation 4.1. If f is a function, then $f^0(x) = x$, and for every integer $n \geq 0$, $f^{n+1}(x) = f(f^n(x)).$

Theorem 4.2. Let $f(x)$ be continuous over the interval $[a,b]$, with $f(b) = b$. Suppose that for all $x \in [a,b)$, $f(x) > x$ and $f(x) \leq b$. Let $c \in [a,b]$. Then $a \leq f^n(c) \leq b$ for every integer $n \geq 0$, and $\lim_{n \rightarrow \infty} f^n(c) = b$.

Proof. First we prove by induction that $a \leq f^n(c) \leq b$ for every integer $n \geq 0$. The statement holds for $n = 0$, since $a \leq c \leq b$. Suppose it holds for $n = i$. Then either $f^i(c) = b$, whence

$$f^{i+1}(c) = f(f^i(c)) = f(b) = b,$$

or else $f^i(c) \in [a,b)$, whence

$$a \leq f^i(c) < f(f^i(c)) \leq b.$$

Thus, the statement holds for every integer $n \geq 0$.

Suppose $f^i(c) = b$ for some i . Then $f^n(c) = b$ for all $n \geq i$, whence $\lim_{n \rightarrow \infty} f^n(c) = b$. Suppose $f^i(c) \neq b$ for all i . Then $f^i(c) \in [a,b)$ for all i and $f^{i+1}(c) = f(f^i(c)) > f^i(c)$, so $\{f^n(c)\}_{n=0}^{\infty}$ is a bounded monotone increasing sequence, which therefore must have a limit $z \in [a,b]$. But therefore, since f is continuous,

$$z = \lim_{n \rightarrow \infty} f^n(c) = \lim_{n \rightarrow \infty} f^{n+1}(c) = \lim_{n \rightarrow \infty} f(f^n(c)) = f(z).$$

The only $z \in [a,b]$ for which this can be true is $z = b$.

△

Corollary 4.2.1. Let $f(x)$ be continuous over the interval $[a,b]$, with $f(a) = a$. Suppose that for all $x \in (a,b]$, $f(x) < x$ and $f(x) \geq a$. Let $c \in [a,b]$. Then $a \leq f^n(c) \leq b$ for every integer $n \geq 0$, and $\lim_{n \rightarrow \infty} f^n(c) = a$.

Proof. Similar to the proof of Theorem 4.2.

△

Lemmas 4.1 and 4.2 can be proved as corollaries of some more general theorems which are proved in later chapters. However, proving those theorems requires considerable preliminary mathematical work. Since Lemmas 4.1 and 4.2 are easy to prove anyway, it is easier to let the general results wait until later.

Lemma 4.1. Let $0 \leq x_0 \leq 1$ and $0 \leq y_0 \leq 1$, with not both $x_0 = 1$ and $y_0 = 0$. For every integer $n \geq 0$, let

$$x_{n+1} = (1 - (1-x_n)(1-y_n))^2$$

and
$$y_{n+1} = (1 - (1-y_n)^2) (1 - (1-x_n)(1-y_n)).$$

Then for every integer $n \geq 0$,

$$0 \leq x_n \leq 1 \quad \text{and} \quad 0 \leq y_n \leq 1,$$

and not both $x_n = 1$ and $y_n = 0$.

Proof (by induction on n). By its hypotheses, the lemma holds for $n = 0$. Suppose it holds for $n = i$. For $0 \leq x \leq 1$ and $0 \leq y \leq 1$, $(1 - (1-x)(1-y))^2$ is an increasing function of x and y , so

$$x_{i+1} = (1 - (1-x_i)(1-y_i))^2 \geq (1 - (1-0)(1-0))^2 = 0$$

and
$$x_{i+1} = (1 - (1-x_i)(1-y_i))^2 \leq (1 - (1-1)(1-1))^2 = 1.$$

Similarly, $0 \leq y_{i+1} \leq 1$. Suppose that

$$(4.3) \quad x_{i+1} = (1 - (1-x_i)(1-y_i))^2 = 1.$$

Then $(1 - (1-x_i)(1-y_i)) = 1$, whence

$$(4.4) \quad x_i = 1 \quad \text{or} \quad y_i = 1.$$

Suppose further that

$$y_{i+1} = (1 - (1-y_i)^2) (1 - (1-x_i)(1-y_i)) = 0.$$

Then either $1 - (1-y_i)^2 = 0$ or $1 - (1-x_i)(1-y_i) = 0$. The latter alternative contradicts equation (4.3), so it must be that $1 - (1-y_i)^2 = 0$, whence $y_i = 0$. But thus from statement (4.4) it follows that $x_i = 1$, which contradicts the induction hypothesis.

Thus not both $x_{i+1} = 1$ and $y_{i+1} = 0$.

△

Lemma 4.2. For every integer $n \geq 0$, let x_n and y_n be as in Lemma 4.1. Then not both $\lim_{n \rightarrow \infty} x_n = 1$ and $\lim_{n \rightarrow \infty} y_n = 0$.

Proof. Because of Lemma 4.1, only the following three cases need be considered.

Case 1: $x_i = 1$ for some i , and $0 < y_n \leq 1$ for all n .

Then

$$x_{i+1} = (1 - (1-1)(1-y_i)) = 1,$$

so by induction, $x_n = 1$ for every $n \geq i$. Thus for every $n \geq i$,

$$y_{n+1} = (1 - (1-y_n)^2) (1 - (1-1)(1-y_n)) = 1 - (1-y_n)^2.$$

But if we let $a = y_i$ and $b = 1$, then $f(y) = 1 - (1-y)^2$ satisfies the hypotheses of Theorem 4.2. Therefore, from Theorem 4.2,

$$\lim_{n \rightarrow \infty} y_n = 1 \neq 0.$$

Case 2: $0 \leq x_n < 1$ for all n , and $y_i = 0$ for some i .

Then

$$y_{i+1} = (1 - (1-0)^2) (1 - (1-x_i)(1-0)) = 0,$$

so by induction, $y_n = 0$ for every $n \geq i$. Thus for every $n \geq i$,

$$x_{n+1} = (1 - (1-x_n)(1-0))^2 = x_n^2.$$

But if we let $a = 0$ and $b = x_i$, then $f(x) = x^2$ satisfies the hypotheses of Corollary 4.2.1. Therefore, from Corollary 4.2.1,

$$\lim_{n \rightarrow \infty} x_n = 0 \neq 1.$$

Case 3: $0 \leq x_n < 1$ and $0 < y_n \leq 1$ for all n .

Suppose

$$\lim_{n \rightarrow \infty} x_n = 1 \quad \text{and} \quad \lim_{n \rightarrow \infty} y_n = 0.$$

Then there is an N such that for every integer $n > N$,

$$0.9 < x_n < 1 \quad \text{and} \quad 0 < y_n < 0.1.$$

Let $n > N$. Then

$$\begin{aligned}
 y_{n+1} &= (1 - (1-y_n)^2) (1 - (1-x_n)(1-y_n))^2 \\
 &= (2y_n - y_n^2) (x_n + y_n - x_n y_n) \\
 &= (2-y_n)y_n (x_n(1-y_n) + y_n) \\
 &> 1.9 y_n (0.9^2 + 0) = 1.539 y_n,
 \end{aligned}$$

so by induction, $y_{n+i} > 1.539^i y_n$ for every integer $i \geq 0$, whence $y_{n+i} > 1$ for some i , which cannot occur. Thus not both $\lim_{n \rightarrow \infty} x_n = 1$ and $\lim_{n \rightarrow \infty} y_n = 0$.

△

Theorem 4.3. For every integer $n \geq 0$, let x_n and y_n be as in Lemma 4.1. Then $\lim_{n \rightarrow \infty} x_n - y_n = 0$.

Proof. For every integer $n \geq 0$,

$$\begin{aligned}
 x_{n+1} &= (x_n + y_n - x_n y_n)^2 \\
 \text{and} \quad y_{n+1} &= (2y_n - y_n^2) (x_n + y_n - x_n y_n), \\
 \text{so} \quad x_{n+1} - y_{n+1} &= (x_n + y_n - x_n y_n) (x_n - y_n - y_n(x_n - y_n)) \\
 &= (x_n + y_n - x_n y_n) (1 - y_n) (x_n - y_n).
 \end{aligned}$$

Thus by induction,

$$(4.5) \quad x_{n+1} - y_{n+1} = (x_0 - y_0) \prod_{i=0}^n (1 - y_i) (x_i + y_i - x_i y_i).$$

From Lemma 4.1, $0 \leq x_n \leq 1$ and $0 \leq y_n \leq 1$, from which it follows that $0 \leq 1 - y_n \leq 1$ and $0 \leq x_n + y_n - x_n y_n \leq 1$, whence

$$(4.6) \quad 0 \leq \prod_{i=0}^n (1 - y_i) \leq 1$$

and

$$(4.7) \quad 0 \leq \prod_{i=0}^n (x_i + y_i - x_i y_i) \leq 1.$$

Case 1: $\lim_{n \rightarrow \infty} y_n \neq 0$ or $\lim_{n \rightarrow \infty} y_n$ is undefined.

Then there is an ϵ in the interval $(0,1)$ such that for infinitely many integers $i \geq 0$, $y_i > \epsilon$, whence $1-y_i < 1-\epsilon$. Thus

$$\lim_{n \rightarrow \infty} \prod_{i=0}^n (1-y_i) = 0.$$

But from equations (4.5) and (4.7),

$$\begin{aligned} |x_n - y_n| &= |x_0 - y_0| \prod_{i=0}^n (1-y_i) (x_i + y_i - x_i y_i) \\ &\leq |x_0 - y_0| (1) \prod_{i=0}^n (1-y_i), \end{aligned}$$

so $\lim_{n \rightarrow \infty} x_n - y_n = 0$.

Case 2: $\lim_{n \rightarrow \infty} y_n = 0$.

Then by Lemma 4.2, either $\lim_{n \rightarrow \infty} x_n \neq 1$ or $\lim_{n \rightarrow \infty} x_n$ is undefined.

Thus there is an ϵ in the interval $(0,1)$ such that for infinitely many integers $i \geq 0$, $0 \leq x_i < 1-2\epsilon$ and $0 \leq y_i < \epsilon$, whence

$$0 \leq x_i + y_i - x_i y_i < 1-2\epsilon + \epsilon - 0 < 1-\epsilon.$$

Thus $\lim_{n \rightarrow \infty} \prod_{i=0}^n (x_i + y_i - x_i y_i) = 0$.

But from equations (4.5) and (4.6),

$$\begin{aligned} |x_n - y_n| &= |x_0 - y_0| \prod_{i=0}^n (1-y_i) (x_i + y_i - x_i y_i) \\ &\leq |x_0 - y_0| (1) \prod_{i=0}^n (x_i + y_i - x_i y_i), \end{aligned}$$

so $\lim_{n \rightarrow \infty} x_n - y_n = 0$.

△

The Pathology Theorem for $\Gamma(1,1)$

As discussed earlier, the goal of this chapter is to prove that $\Gamma(1,1)$ is pathological. This is done in Theorem 4.4 below. In order to prove this theorem, we first need a formal definition of pathology.

If g is a critical node of G having m children of the same sign and n children of opposite sign, then according to Theorem 2.3, the probability of making a correct decision when choosing moves totally at random is $\frac{m}{m+n} = D(G,g,P,0)$, independent of P . Thus, pathology may be defined as follows.

Definition 4.1. Let P be a probability vector. If G is a game tree such that

$$\lim_{d \rightarrow \infty} D(G,g,P,d) = D(G,g,P,0)$$

for every $g \in N(G)$, then G is P -pathological.

In order to prove Theorem 4.4, one final lemma is needed.

Lemma 4.3. Let P be an \hat{f} -vector and $d \geq 0$ be an integer. Then $D_{1,1}(P,0) = \frac{1}{2}$, and

$$D_{1,1}(P,d+1) = \sum_{k=0}^r (T_d)_k (U_d)_{0,k-1} + \frac{1}{2} \sum_{k=0}^r (T_d)_k (U_d)_k.$$

Proof. From Theorem 2.3, $D_{1,1}(P,0) = \frac{1}{1+1} = \frac{1}{2}$, and from Theorem 3.2,

$$\begin{aligned} D_{1,1}(P,d+1) &= \sum_{i=1}^1 \sum_{j=0}^1 \frac{1}{1+j} \sum_{k=0}^r A(1,i,k,T_d) A(1,j,k,U_d) \\ &= \sum_{j=0}^1 \frac{1}{1+j} \sum_{k=0}^r (T_d)_k A(1,j,k,U_d) \\ &= \sum_{j=0}^1 \frac{1}{1+j} \sum_{k=0}^r (T_d)_k ((U_d)_k)^j ((U_d)_{0,k-1})^{1-j} \end{aligned}$$

$$= \sum_{k=0}^r (T_d)_k (U_d)_{0,k-1} + \frac{1}{2} \sum_{k=0}^r (T_d)_k (U_d)_k.$$

△

Theorem 4.4. $\Gamma(1,1)$ is P-pathological for every imperfect \hat{f} -vector P.

Proof. Let P be an imperfect \hat{f} -vector. Then from Definition 2.17, there is an integer k such that $0 \leq k \leq r/2$ and $(P)_k > 0$. Suppose that $(T_0)_{i,r} = 1$ for some i. Then from Corollary 3.1.1, $(P)_{i,r} = 1$. But then $i \leq k$, so

$$k \leq r/2 = r-r/2 \leq r-k \leq r-i,$$

so from Corollary 3.1.1 and Definitions 2.9 and 3.5,

$$(U_0)_{i,r} = (\text{rev } P)_{i,r} = (P)_{0,r-i} \geq (P)_{0,k} \geq (P)_k > 0.$$

Thus for each $i \in \hat{f}$, not both $(T_0)_{i,r} = 1$ and $(U_0)_{i,r} = 0$, so by Corollary 4.1.6 and Theorem 4.3,

$$(4.8) \quad \lim_{d \rightarrow \infty} (T_{2d})_{i,r} - (U_{2d})_{i,r} = 0.$$

Suppose that $(T_1)_{i,r} = 1$ for some $i \in \hat{f}$. By Corollaries 4.1.4 and 3.1.1,

$$(T_1)_{i,r} = ((S_0)_{i,r})^2 = ((P)_{i,r})^2,$$

so $(P)_{i,r} = 1$. Thus as before, $k \leq r-i$, so from Corollaries 4.1.4 and 3.1.1,

$$\begin{aligned} (U_1)_{i,r} &= (V_0)_{i,r} (S_0)_{i,r} = (V_0)_{i,r} (P)_{i,r} \\ &= (V_0)_{i,r} = (\text{rev } P)_{i,r} = (P)_{0,r-i} \geq (P)_{0,k} \geq (P)_k > 0. \end{aligned}$$

Thus for each $i \in \hat{f}$, not both $(T_1)_{i,r} = 1$ and $(U_1)_{i,r} = 0$, so by Corollary 4.1.6 and Theorem 4.3,

$$(4.9) \quad \lim_{d \rightarrow \infty} (T_{2d+1})_{i,r} - (U_{2d+1})_{i,r} = 0.$$

From equations (4.8) and (4.9), $\lim_{d \rightarrow \infty} (T_d)_{i,r} - (U_d)_{i,r} = 0$. Thus for each $i \in \hat{f}$,

$$\begin{aligned}
 (4.10) \quad & \lim_{d \rightarrow \infty} (T_d)_i - (U_d)_i \\
 &= \lim_{d \rightarrow \infty} ((T_d)_{i,r} - (T_d)_{i+1,r}) - ((U_d)_{i,r} - (U_d)_{i+1,r}) \\
 &= \lim_{d \rightarrow \infty} (T_d)_{i,r} - (U_d)_{i,r} - \lim_{d \rightarrow \infty} (T_d)_{i+1,r} - (U_d)_{i+1,r} \\
 &= 0 - 0 = 0.
 \end{aligned}$$

But by Lemma 4.3,

$$\begin{aligned}
 D_{1,1}(P, d+1) &= \sum_{i=0}^r (T_d)_i (U_d)_{0,i-1} + \frac{1}{2} \sum_{i=0}^r (T_d)_i (U_d)_i \\
 &= \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} ((T_d)_j + (U_d)_j - (T_d)_j) \\
 &\quad + \frac{1}{2} \sum_{i=0}^r ((T_d)_i)^2 + (T_d)_i ((U_d)_i - (T_d)_i) \\
 &= \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} (T_d)_j + \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} ((U_d)_j - (T_d)_j) \\
 &\quad + \frac{1}{2} \sum_{i=0}^r ((T_d)_i)^2 + \frac{1}{2} \sum_{i=0}^r (T_d)_i ((U_d)_i - (T_d)_i) \\
 &= \frac{1}{2} (2) \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} (T_d)_j + \frac{1}{2} \sum_{i=0}^r ((T_d)_i)^2 \\
 &\quad + \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} ((U_d)_j - (T_d)_j) + \frac{1}{2} \sum_{i=0}^r (T_d)_i ((U_d)_i - (T_d)_i) \\
 &= \frac{1}{2} \sum_{i=0}^r (T_d)_i \sum_{j=0}^{i-1} (T_d)_j + \frac{1}{2} \sum_{j=0}^r (T_d)_j \sum_{i=j+1}^r (T_d)_i + \frac{1}{2} \sum_{i=0}^r ((T_d)_i)^2 \\
 &\quad + \sum_{j=0}^r ((U_d)_j - (T_d)_j) \sum_{i=j+1}^r (T_d)_i + \frac{1}{2} \sum_{i=0}^r ((U_d)_i - (T_d)_i) (T_d)_i
 \end{aligned}$$

$$\begin{aligned}
 &= \frac{1}{2} \sum_{i=0}^r (T_d)_i \left(\sum_{j=0}^{i-1} (T_d)_j + \sum_{j=i+1}^r (T_d)_j + (T_d)_i \right) \\
 &\quad + \sum_{j=0}^r ((U_d)_j - (T_d)_j) (T_d)_{j+1,r} + \frac{1}{2} \sum_{i=0}^r ((U_d)_i - (T_d)_i) (T_d)_i \\
 &= \frac{1}{2} (1) (1) \\
 &\quad + \sum_{j=0}^r ((U_d)_j - (T_d)_j) (T_d)_{j+1,r} + \frac{1}{2} \sum_{i=0}^r ((U_d)_i - (T_d)_i) (T_d)_i,
 \end{aligned}$$

so since $0 \leq (T_d)_{i+1,r} \leq 1$ and $0 \leq (T_d)_i \leq 1$ for every $i \in \hat{r}$, it follows from equation (4.10) that

$$\begin{aligned}
 &\lim_{d \rightarrow \infty} D_{1,1}(P, d+1) \\
 &= \frac{1}{2} (1) (1) + \lim_{d \rightarrow \infty} \sum_{j=0}^r ((U_d)_j - (T_d)_j) (T_d)_{j+1,r} \\
 &\quad + \frac{1}{2} \lim_{d \rightarrow \infty} \sum_{i=0}^r ((U_d)_i - (T_d)_i) (T_d)_i \\
 &= \frac{1}{2} + \sum_{i=0}^r \lim_{d \rightarrow \infty} ((U_d)_i - (T_d)_i) (T_d)_{i+1,r} \\
 &\quad + \frac{1}{2} \sum_{i=0}^r \lim_{d \rightarrow \infty} ((U_d)_i - (T_d)_i) (T_d)_i \\
 &= \frac{1}{2} + \sum_{i=0}^r 0 + \frac{1}{2} \sum_{i=0}^r 0 = \frac{1}{2}.
 \end{aligned}$$

△

Experimental Results

Writers of game playing computer programs generally try to select good evaluation functions for use in their programs. As discussed in Chapter 2, good evaluation functions have "good" probability vectors (i.e., probability vectors P such that $(P)_i$ is large for large i and small for small i), and perfect

evaluation functions have perfect probability vectors (see Definition 2.17).

According to Proposition 2.4, almost all probability vectors are imperfect. For such vectors, Theorem 4.4 states that on $\Gamma(1,1)$ it makes no difference whether P is good or bad--the probability of correct decision approaches $1/2$ as the search depth increases. For perfect probability vectors, the probability of correct decision is 1 , independent of the search depth, so searching is useless in this case as well. This latter fact is illustrated experimentally in Appendix B.

The behavior predicted by Theorem 4.4 has been tested numerous times experimentally, and the results have never failed to be in accordance with the theorem. In fact, it was by conducting these experiments that the author gained the insights which led to the proof of the theorem. The results of a representative sample of tests are given in Tables 4.1 through 4.5, and summarized in Figure 4.1.

Suppose e is a P -function and g is a critical node. Then g is either an S node or a U node. Choosing a move from g using a depth d minimax search involves computing the depth $d-1$ minimax values for the children of g . If g is an S node, then the children of g are T and U nodes, and so the probability of correct decision $D(P,d)$ may be computed from the probability vectors T_{d-1} and U_{d-1} using the formula given in Theorem 3.2. According to Theorem 3.2, this same formula works if g is a U node. Tables 4.1 through 4.5 give the values of T_{d-1} , U_{d-1} , and $D(P,d)$ for various values of P and d , as computed using the computer program of Appendix G. The values of P were chosen to

provide a wide variety of examples, and yet keep the presentation reasonably concise.

Table 4.1 is for a probability vector P whose entries increase linearly from left to right: the vector $(1,2,3,4)/10$. In accordance with Corollary 3.1.1, $T_0 = P$ and $U_0 = \text{rev } P$. The probability of correct decision at depth 1 is 0.75; thus, the vector is a fairly good one. But as predicted by Theorem 4.4, T_{d-1} and U_{d-1} become more and more alike as d increases, and $D(P,d)$ decreases monotonically to $1/2$. At depth 11, $D(P,d) = 0.500$, correct to three decimal places.

Table 4.2 is for a fairly bad vector: the reverse of the vector of Table 4.1. For depth 1, the probability of correct decision is as much below $1/2$ as it was above $1/2$ in Table 4.1, but the convergence is slightly faster this time.

Table 4.3 is for a very good probability vector--one whose entries increase exponentially from left to right. This vector is the normalization of the vector $(5^0, 5^1, 5^2, 5^3, 5^4)$. In this case, the probability of correct decision is initially very high, and the convergence is slower than in Table 4.1.

Table 4.4 is for the reverse of the vector of Table 4.3. Interestingly enough, although the probability of correct decision is initially much lower than in Table 4.2, the convergence is considerably faster.

Table 4.5 is for the vector $(.2,.2,.2,.2,.2)$. As one would expect in this case, the probability of correct decision is $1/2$ at all search depths.

According to the proof of Theorem 4.4, $\bar{T}(1,1)$ is pathological because T_d and U_d look more and more alike as d

approaches infinity. As shown in Tables 4.1 through 4.5, the way that this happens is that one entry in each vector converges to 1, and the rest converge to 0. Which entry approaches 1 and which entries approach 0 depends on whether d is odd or even.

The minimax values used to make a decision at odd search depths have the probability vectors T_{2d} and U_{2d} . When P is imperfect, both of these vectors converge to a vector $Q = (0, 0, \dots, 0, 1, 0, 0, \dots, 0)$, which has at least as many "0" entries to the left of the "1" entry as to the right of it, and usually more. This means that for large odd search depths, both minimax values are large, and thus both moves look good. Similarly, the probability vectors governing the decision at even search depths are T_{2d+1} and U_{2d+1} , which both converge to $\text{rev } Q$. This means that for large even search depths, both minimax values are small, and thus both moves look bad. This "manic-depressive" behavior is further discussed in Chapters 5 and 7.

For both odd and even search depths, as the search depth increases, all moves tend more and more strongly to receive exactly the same minimax value. Thus, the choice of what move to make becomes more and more a random one.

TABLE 4.1.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.1,.2,.3,.4)$.

d	T_{d-1}				U_{d-1}				$D(P,d)$
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 4.2.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.4,.3,.2,.1)$.

d	T_{d-1}				U_{d-1}				$D(P,d)$
1	0.40	0.30	0.20	0.10	0.10	0.20	0.30	0.40	0.250
3	0.08	0.30	0.41	0.21	0.05	0.23	0.42	0.29	0.435
5	0.01	0.19	0.60	0.20	0.01	0.17	0.60	0.22	0.480
7	0.00	0.07	0.79	0.14	0.00	0.07	0.79	0.15	0.495
9	0.00	0.01	0.92	0.07	0.00	0.01	0.92	0.07	0.499
11	0.00	0.00	0.98	0.02	0.00	0.00	0.98	0.02	0.500
13	0.00	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.500
2	0.64	0.27	0.08	0.01	0.46	0.33	0.17	0.04	0.396
4	0.50	0.42	0.08	0.00	0.44	0.45	0.10	0.00	0.466
6	0.40	0.57	0.03	0.00	0.38	0.59	0.04	0.00	0.489
8	0.28	0.72	0.00	0.00	0.27	0.73	0.00	0.00	0.497
10	0.14	0.86	0.00	0.00	0.14	0.86	0.00	0.00	0.499
12	0.04	0.96	0.00	0.00	0.04	0.96	0.00	0.00	0.500
14	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500

TABLE 4.3.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (1,5,25,125,625)/781$.

d	T_{d-1}					U_{d-1}					D(P,d)
1	0.00	0.01	0.03	0.16	0.80	0.80	0.16	0.03	0.01	0.00	0.996
3	0.00	0.01	0.06	0.28	0.64	0.64	0.28	0.06	0.01	0.00	0.990
5	0.00	0.02	0.12	0.44	0.41	0.41	0.44	0.12	0.02	0.00	0.972
7	0.00	0.04	0.22	0.56	0.17	0.17	0.56	0.22	0.04	0.00	0.927
9	0.00	0.06	0.38	0.53	0.03	0.03	0.53	0.38	0.07	0.00	0.839
11	0.00	0.07	0.58	0.34	0.00	0.00	0.33	0.59	0.08	0.00	0.711
13	0.00	0.05	0.80	0.16	0.00	0.00	0.13	0.81	0.06	0.00	0.583
15	0.00	0.01	0.95	0.04	0.00	0.00	0.02	0.95	0.02	0.00	0.515
17	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.501
19	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500
21	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500
2	0.00	0.01	0.06	0.28	0.64	0.80	0.16	0.03	0.01	0.00	0.994
4	0.00	0.03	0.12	0.44	0.41	0.64	0.28	0.06	0.01	0.00	0.983
6	0.01	0.05	0.22	0.56	0.17	0.41	0.44	0.12	0.02	0.00	0.953
8	0.00	0.08	0.37	0.51	0.03	0.17	0.58	0.22	0.03	0.00	0.889
10	0.00	0.13	0.56	0.31	0.00	0.03	0.56	0.38	0.04	0.00	0.779
12	0.00	0.15	0.74	0.11	0.00	0.00	0.39	0.58	0.02	0.00	0.648
14	0.00	0.11	0.87	0.02	0.00	0.00	0.21	0.79	0.01	0.00	0.551
16	0.00	0.05	0.95	0.00	0.00	0.00	0.06	0.94	0.00	0.00	0.509
18	0.00	0.01	0.99	0.00	0.00	0.00	0.01	0.99	0.00	0.00	0.501
20	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500
22	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500

TABLE 4.4.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (625,125,25,5,1)/781$.

d	T_{d-1}					U_{d-1}					D(P,d)
1	0.80	0.16	0.03	0.01	0.00	0.00	0.01	0.03	0.16	0.80	0.004
3	0.00	0.01	0.06	0.28	0.64	0.00	0.01	0.03	0.19	0.77	0.435
5	0.00	0.00	0.01	0.15	0.84	0.00	0.00	0.00	0.13	0.87	0.486
7	0.00	0.00	0.00	0.04	0.96	0.00	0.00	0.00	0.04	0.96	0.498
9	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.500
11	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.500
2	0.96	0.04	0.00	0.00	0.00	0.80	0.16	0.03	0.01	0.00	0.420
4	0.95	0.05	0.00	0.00	0.00	0.92	0.08	0.00	0.00	0.00	0.485
6	0.98	0.02	0.00	0.00	0.00	0.98	0.02	0.00	0.00	0.00	0.498
8	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.500
10	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.500

TABLE 4.5.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.2, .2, .2, .2, .2)$.

d	T_{d-1}					U_{d-1}					D(P,d)
1	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.500
3	0.08	0.22	0.30	0.28	0.13	0.08	0.22	0.30	0.28	0.13	0.500
5	0.01	0.15	0.41	0.37	0.06	0.01	0.15	0.41	0.37	0.06	0.500
7	0.00	0.05	0.50	0.43	0.01	0.00	0.05	0.50	0.43	0.01	0.500
9	0.00	0.01	0.51	0.48	0.00	0.00	0.01	0.51	0.48	0.00	0.500
11	0.00	0.00	0.46	0.54	0.00	0.00	0.00	0.46	0.54	0.00	0.500
13	0.00	0.00	0.39	0.61	0.00	0.00	0.00	0.39	0.61	0.00	0.500
15	0.00	0.00	0.27	0.73	0.00	0.00	0.00	0.27	0.73	0.00	0.500
17	0.00	0.00	0.15	0.85	0.00	0.00	0.00	0.15	0.85	0.00	0.500
19	0.00	0.00	0.04	0.96	0.00	0.00	0.00	0.04	0.96	0.00	0.500
21	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.500
2	0.36	0.28	0.20	0.12	0.04	0.36	0.28	0.20	0.12	0.04	0.500
4	0.24	0.41	0.26	0.08	0.01	0.24	0.41	0.26	0.08	0.01	0.500
6	0.11	0.55	0.30	0.03	0.00	0.11	0.55	0.30	0.03	0.00	0.500
8	0.03	0.67	0.30	0.00	0.00	0.03	0.67	0.30	0.00	0.00	0.500
10	0.00	0.73	0.27	0.00	0.00	0.00	0.73	0.27	0.00	0.00	0.500
12	0.00	0.78	0.22	0.00	0.00	0.00	0.78	0.22	0.00	0.00	0.500
14	0.00	0.85	0.15	0.00	0.00	0.00	0.85	0.15	0.00	0.00	0.500
16	0.00	0.92	0.08	0.00	0.00	0.00	0.92	0.08	0.00	0.00	0.500
18	0.00	0.98	0.02	0.00	0.00	0.00	0.98	0.02	0.00	0.00	0.500
20	0.00	1.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.500

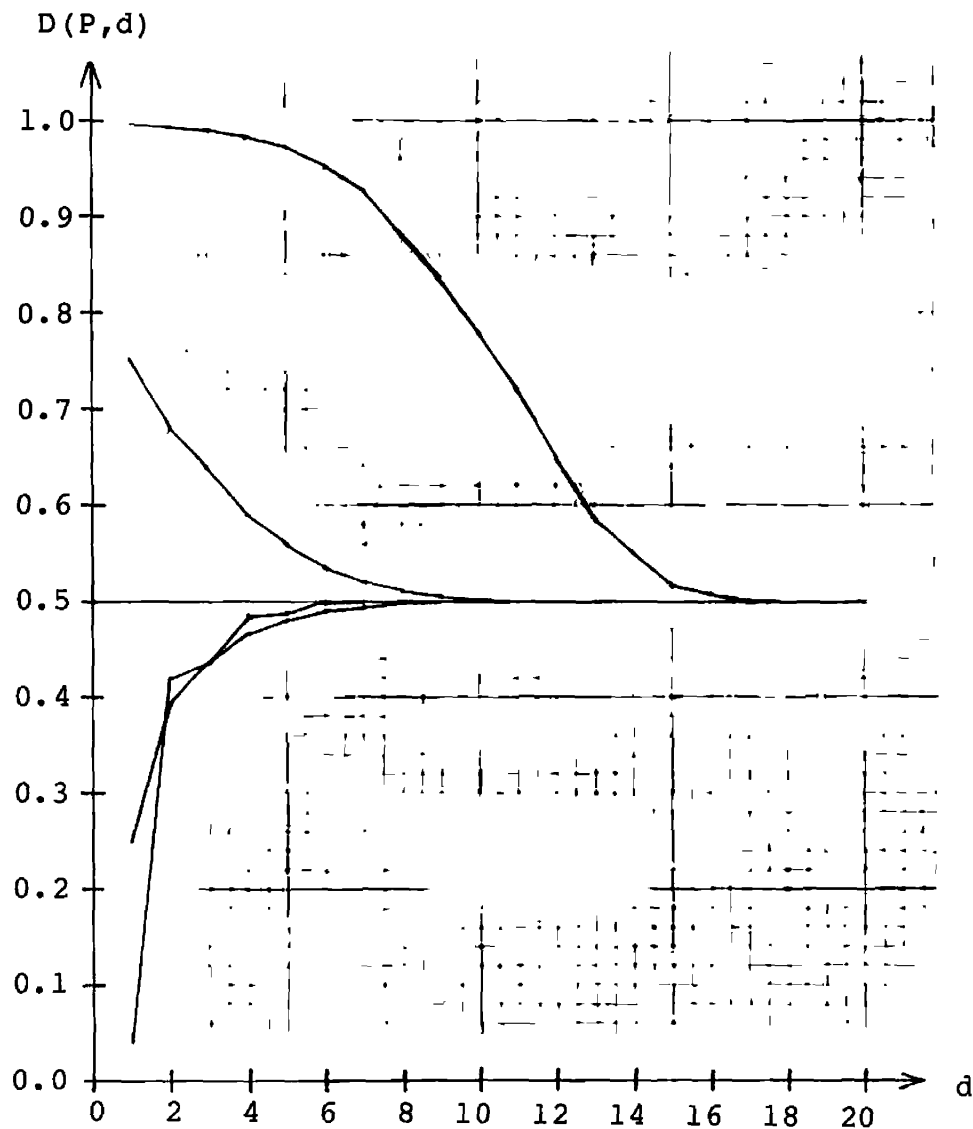


FIGURE 4.1.--Probability of correct decision as a function of search depth for five different probability vectors. From top to bottom, the curves are for the vectors $(1,5,25,125,625)/781$, $(.1,.2,.3,.4)$, $(.2,.2,.2,.2,.2)$, $(.4,.3,.2,.1)$, and $(625,125,25,5,1)/781$.

CHAPTER 5

THE "LAST PLAYER" THEOREM

Introduction

A complete n-ary game tree of depth d is a game tree for which every node has exactly n children, with the exception that the nodes at depth d are all leaves. On such a tree, the same player (say, Max) always has the last move. It is shown in this chapter that for the large majority of such trees, Max has a forced win at the beginning of the game. As discussed at the end of this chapter, this effect provides an intuitive explanation for the "manic-depressive" behavior described at the end of Chapter 4.

Suppose G is a complete n -ary game tree of depth d , whose leaf nodes are independently randomly labeled "+" and "-", with a fixed probability p of any given leaf being labeled "+". Let Max be the player who always has the last move. If d is odd, then Max also has the first move in the game; if d is even, then Min has the first move. A forced win for Max on G requires the existence of a certain set of paths in G leading to "+" leaves. This is illustrated in Figure 5.1 for $n = 2$ and $d = 4$. According to Theorem 5.3, if d is large and p exceeds a cutoff value which depends on n , then the probability of Max having a forced win on G (and on many other trees) approaches 1 as d increases. Since the cutoff value approaches 0 as n increases, this behavior occurs for most values of n and p .

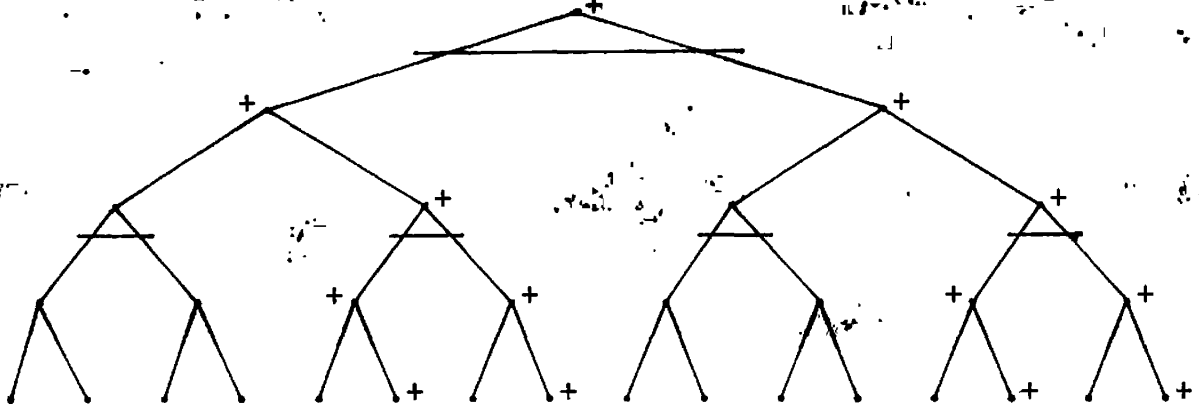


FIGURE 5.1.--A binary tree of depth 4 with a forced win for Max. The forced win requires the existence of a set of "+" nodes such as the ones indicated. The signs of the other nodes do not matter.

Theorem 5.3 is sometimes called the "last player" theorem. Although its proof is different in several respects from the proof of Theorem 4.4, there are also certain similarities, and the preliminary results needed to prove Theorem 5.3 are also important in later chapters. These results are presented in the following two sections, which are followed by a section containing a presentation and discussion of Theorem 5.3.

Theorems about Sequences

In Chapter 4, much use was made of recursively defined sequences whose definitions involved expressions of the general form $(1 - (1-x)^a (1-y)^b)^c$. Such sequences appear quite frequently in the following pages. The following theorems aid in analyzing their properties.

Theorem 5.1. Let $n \geq 1$. Then there is exactly one $x \in [0,1]$ such that $(1-x)^n = x$.

Proof. Let $f_n(x) = (1-x)^n - x$. Then $(1-x)^n = x$ if and only if $f_n(x) = 0$. $f_n(0) = 1$ and $f_n(1) = -1$, so by the intermediate value theorem, there is a point $z_0 \in (0,1)$ for which $f_n(z_0) = 0$.

Suppose there is another point $z_1 \in (0,1)$ such that $f_n(z_1) = 0$. Without loss of generality, we may assume that $z_0 < z_1$. Then by the mean value theorem, there is a $y \in (z_0, z_1)$ such that $f_n'(y) = 0$. But

$$f_n'(x) = n(1-x)^{n-1}(-1) - 1 = -n(1-x)^{n-1} - 1,$$

so if $f_n'(y) = 0$, then $(1-y)^{n-1} = -1/n < 0$. But since $0 < y < 1$, this cannot be. Thus z_1 cannot exist.

△

Notation 5.2. For $n \geq 1$, the unique $x \in (0,1)$ for which $(1-x)^n = x$ is called w_n .

We are only interested in w_n for integers $n \geq 2$. However, most of the properties we need to prove about w_n may be proved just as easily for every real $n > 1$.

Corollary 5.1.1. If $1 \leq m < n$, then $w_m > w_n$.

Proof. Let $1 \leq m < n$, and suppose $w_m \leq w_n$. Then

$$w_n = (1-w_n)^n < (1-w_n)^m \leq (1-w_m)^m = w_m,$$

which is a contradiction.

△

Corollary 5.1.2. Let $n \geq 1$ and $x \in [0,1]$. If $x < w_n$, then $(1-x)^n > w_n$. If $x > w_n$, then $(1-x)^n < w_n$. If $x = w_n$, then $(1-x)^n = w_n$.

Proof. Obviously, $(1-x)^n$ is decreasing for $x \in [0,1]$. Thus if $x < w_n$, then $(1-x)^n > (1-w_n)^n = w_n$, and if $x > w_n$, then $(1-x)^n < (1-w_n)^n = w_n$. It is merely a restatement of Theorem 5.1

to say that $(1-x)^n = w_n$ when $x = w_n$.

△

Corollary 5.1.3. $\lim_{n \rightarrow \infty} w_n = 0$, and the convergence is monotonic.

Proof. Let $0 < \epsilon < 1$. Then $0 < 1-\epsilon < 1$, so there is an N such that for every $n > N$, $0 < (1-\epsilon)^n < \epsilon$. From Corollary 5.1.2, if $\epsilon \leq w_n$ then $(1-\epsilon)^n \geq w_n$, which is a contradiction. Thus $\epsilon > w_n$. Thus, since $w_n > 0$ for every n , $\lim_{n \rightarrow \infty} w_n = 0$. By Corollary 5.1.1, the convergence is monotonic.

△

Obviously $w_1 = \frac{1}{2}$, and it is easy to show using the quadratic formula that $w_2 = \frac{3 - \sqrt{5}}{2}$. Indeed, although this is not germane to the dissertation, it can easily be shown that w_n is irrational for every integer $n \geq 2$. Table 5.1 gives approximate values of w_1 through w_{50} .

Graphically, w_n can be thought of as the intersection of the functions $y = (1-x)^n$ and $y = x$. For $n > 1$, w_n is also the unique intersection in the interval $(0,1)$ of the functions $y = (1 - (1-x)^n)^n$ and $y = x$. This is proved in Lemma 5.1, and is illustrated for w_2 in Figure 5.2. This fact is quite important in later developments. For example, Theorem 5.2, from which the "last player" theorem follows almost as a corollary, states the limit of the sequence $x_{i+1} = (1 - (1-x_i)^n)^n$ in terms of w_n .

Lemma 5.1. Let $n > 1$, and let

$$g_n(x) = (1 - (1-x)^n)^n - x$$

for all real x . Then g_n has exactly one root in the interval $(0,1)$. This is at $x = w_n$.

TABLE 5.1.--Approximate values of w_n , for $n = 1, 2, \dots, 50$.
A larger table appears in Appendix A.

n	w_n	n	w_n	n	w_n	n	w_n	n	w_n
1	.50000	11	.15560	21	.10271	31	.07872	41	.06463
2	.38197	12	.14745	22	.09955	32	.07700	42	.06352
3	.31767	13	.14024	23	.09662	33	.07536	43	.06246
4	.27551	14	.13382	24	.09387	34	.07380	44	.06144
5	.24512	15	.12805	25	.09130	35	.07231	45	.06045
6	.22191	16	.12283	26	.08889	36	.07088	46	.05950
7	.20346	17	.11809	27	.08662	37	.06952	47	.05858
8	.18835	18	.11375	28	.08448	38	.06822	48	.05770
9	.17570	19	.10977	29	.08245	39	.06697	49	.05684
10	.16492	20	.10610	30	.08054	40	.06577	50	.05601

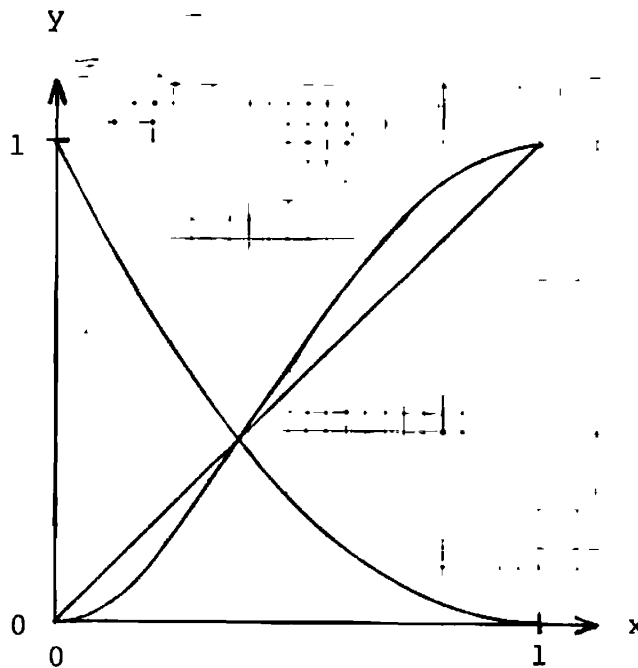


FIGURE 5.2.--Sketches of $y = (1-x)^2$, $y = (1 - (1-x)^2)^2$, and $y = x$.

Proof.

$g_n(w_n) = (1 - (1-w_n)^n)^n - w_n = (1-w_n)^n - w_n = w_n - w_n = 0,$
 so w_n is a root of g_n .

Suppose g_n has roots x_1 and x_2 in $(0,1)$, with $x_1 < x_2$.
 In addition, 0 and 1 are roots of g_n , so from the mean value
 theorem, there are points $y_1, y_2,$ and y_3 such that

$$0 < y_1 < x_1 < y_2 < x_2 < y_3 < 1$$

and $g_n'(y_1) = g_n'(y_2) = g_n'(y_3) = 0.$

Therefore, applying the mean value theorem again, there are
 points z_1 and z_2 such that $0 < y_1 < z_1 < y_2 < z_2 < y_3 < 1$ and
 $g_n''(z_1) = g_n''(z_2) = 0.$ Now,

$$g_n'(x) = n^2(1-(1-x)^n)^{n-1}(1-x)^{n-1} - 1,$$

so $g_n''(x) = n^2(n-1)(1-(1-x)^n)^{n-2}(1-x)^{n-2}((n+1)(1-x)^n - 1).$

For $x \in (0,1),$ $n^2(n-1)(1-(1-x)^n)^{n-2}(1-x)^{n-2} > 0,$ so
 $g_n''(x) = 0$ if and only if $(n+1)(1-x)^n - 1 = 0.$ There is exactly
 one $x \in (0,1)$ for which this is true. Thus not both z_1 and z_2
 can exist, so w_n is the only root of g_n in $(0,1).$

△

Lemma 5.2. Let $n > 1.$ Then--

1. if $0 < x < w_n,$ then $(1 - (1-x)^n)^n < x;$
2. if $w_n < x < 1,$ then $(1 - (1-x)^n)^n > x;$
3. if $x = w_n,$ then $(1 - (1-x)^n)^n = x.$

Proof. Let g_n be as in Lemma 5.1. Then $g_n(0) = 0,$
 $g_n'(0) = -1,$ and as shown in Lemma 5.1, $g_n(x) \neq 0$ for $0 < x < w_n.$
 Therefore, by the intermediate value theorem, $g_n(x) < 0$ for
 $0 < x < w_n.$ Similarly, $g_n(x) > 0$ for $w_n < x < 1.$ Statement 3
 follows directly from Corollary 5.1.2.

△

Theorem 5.2. Let $x_0 \in [0,1]$, and let $n > 1$. For every integer $i \geq 0$, let $x_{i+1} = (1 - (1-x_i)^n)^n$. Then--

1. if $0 \leq x_0 < w_n$, then $\lim_{i \rightarrow \infty} x_i = 0$;
2. if $w_n < x_0 \leq 1$, then $\lim_{i \rightarrow \infty} x_i = 1$;
3. if $x_0 = w_n$, then $x_i = w_n$ for all i .

Proof. Let $f(x) = (1 - (1-x)^n)^n$ for every real x . If $a = 0$ and $0 < b < w_n$, then it follows from Lemma 5.2 that f satisfies the hypotheses of Corollary 4.2.1, whence $\lim_{i \rightarrow \infty} x_i = 0$ if $x_0 \in [0, w_n)$. If $w_n < a < 1$ and $b = 1$, then it follows from Lemma 5.2 that f satisfies the hypotheses of Theorem 4.2, whence $\lim_{i \rightarrow \infty} x_i = 1$ if $x_0 \in (w_n, 1]$. This proves statements 1 and 2. Statement 3 follows immediately from Lemma 5.2.

△

The Theorem and a Discussion

The "last player" theorem is first formally stated and proved, and then its meaning is discussed.

Theorem 5.3. Let $p \in [0,1]$, and let $G_{n,d}$ be a complete n -ary game tree of depth d whose leaves are independently randomly labeled "+" and "-" in such a way that $\Pr [g \text{ is a + node} \mid g \text{ is a leaf of } G_{n,d}] = p$. Let Max be the player who always has the last move in $G_{n,d}$, and let $W_{n,d} = \Pr [\text{Max has a forced win on } G_{n,d}]$. Then--

1. if $p > w_n$, then $\lim_{d \rightarrow \infty} W_{n,d} = 1$;
2. if $p < w_n$, then $\lim_{d \rightarrow \infty} W_{n,d} = 0$;

3. if $p = w_n$, then $W_{n,d} = 1 - w_n$ if d is odd,
 $= w_n$ if d is even.

Proof. Let g be the root of $G_{n,d}$. If $d = 0$, then g is a leaf, so $W_{n,0} = p$. If $d > 0$ is odd, then Max has the first move, so he can force a win on $G_{n,d}$ if he can force a win from any of g 's children. Thus if $d > 0$ is odd, then

$$W_{n,d} = 1 - (1 - W_{n,d-1})^n.$$

If $d > 0$ is even, then Min has the first move, so Max can force a win on $G_{n,d}$ only if he can force a win from each of g 's children. Thus if $d > 0$ is even, then

$$W_{n,d} = (W_{n,d-1})^n = (1 - (1 - W_{n,d-2})^n)^n.$$

Thus for every $d \geq 0$,

$$W_{n,2(d+1)} = (1 - (1 - W_{n,2d})^n)^n,$$

so by Theorem 5.2,

$$\begin{aligned} \lim_{d \rightarrow \infty} W_{n,2d} &= 0 && \text{if } p < w_n, \\ &= 1 && \text{if } p > w_n, \end{aligned}$$

and $W_{n,2d} = w_n$ if $p = w_n$.

Thus $\lim_{d \rightarrow \infty} W_{n,2d+1} = 1 - (1 - \lim_{d \rightarrow \infty} W_{n,2d})^n = 0$ if $p < w_n$,
 $= 1$ if $p > w_n$,

and $W_{n,2d+1} = 1 - (1 - w_n)^n = 1 - w_n$ if $p = w_n$.

△

Theorem 5.3 says that for an n -ary tree, if the probability of a leaf node being a forced win is greater than w_n , the probability of the last player having a forced win approaches 1 as the depth of the tree increases. As can be seen from Table 5.1, w_n converges to 0 fairly quickly. Thus Theorem 5.3 says that the last player has a forced win on the large majority of

complete game trees with a constant branching factor and randomly assigned leaf values.

The probability of the last player having a forced win usually approaches 1 quite rapidly as the depth increases. As an illustration of this, the probabilities of the last player having a forced win on trees of various depths and branching factors are listed in Table 5.2 and graphed in Figure 5.3.

The situation modeled by Theorem 5.3 is rather different from that of Theorem 4.4. In particular, the nodes in the tree of Theorem 4.4 do not get their values in a random manner. However, Theorem 4.4 models a situation in which a player is searching a tree to an arbitrary depth and applying to the nodes at that depth an evaluation function which makes errors in a random way. These nodes are the leaves of the tree that the player sees (although they are not the leaves of the game tree itself), and the errors produced by the evaluation function make the values the player sees somewhat random. Strictly speaking, the "last player" theorem still does not apply, because these values are not stochastically independent of each other. However, one might expect the behavior predicted by the theorem to occur anyway.

A depth d minimax search involves computing the depth $d-1$ minimax values for the children of the current node. If d is odd, then the player doing the search perceives himself as being the last player. If the "last player" phenomenon occurs, each of his possible moves should tend to look like forced wins, and so the depth $d-1$ minimax values should be high. This means that the probability vectors for these values should tend to have their

TABLE 5.2.--The probability $W_{n,d}$ of the last player having a forced win on a complete n -ary game tree of depth d , for various values of n and d . The probability p of a leaf node being a "+" node is $1/2$ in each case.

d	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$
0	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
1	0.7500	0.8750	0.9375	0.9688	0.9844	0.9922	0.9961
2	0.5625	0.6699	0.7725	0.8532	0.9098	0.9466	0.9692
3	0.8086	0.9640	0.9973	0.9999	1.0000	1.0000	1.0000
4	0.6538	0.8959	0.9893	0.9997	1.0000	1.0000	1.0000
5	0.8802	0.9989	1.0000	1.0000	1.0000	1.0000	1.0000
6	0.7747	0.9966	1.0000	1.0000	1.0000	1.0000	1.0000
7	0.9492	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	0.9010	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
9	0.9902	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
10	0.9805	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
11	0.9996	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
12	0.9992	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
13	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
14	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

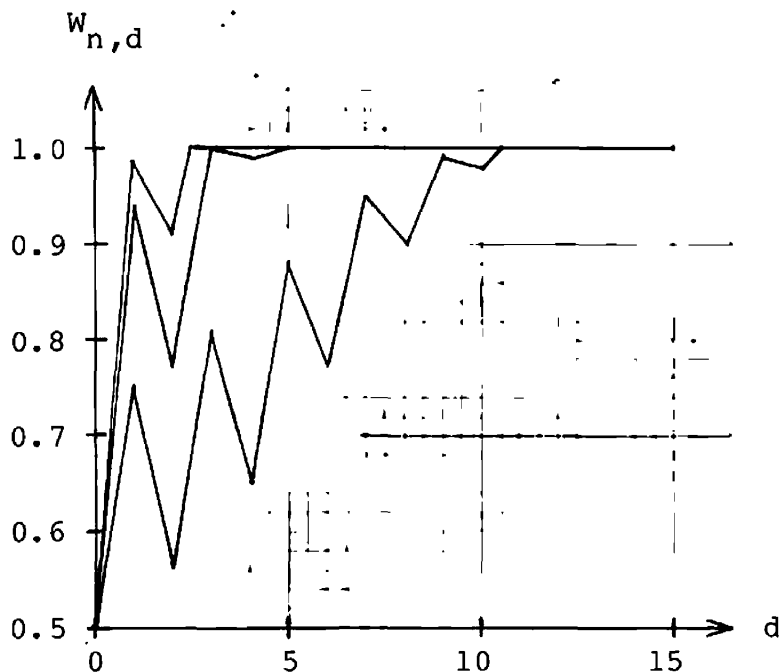


FIGURE 5.3.--The probability $W_{n,d}$ of the last player having a forced win on a complete n -ary game tree of depth d , as a function of d . From bottom to top, the curves are for $n = 2, 4,$ and 6 . The probability p of a leaf node being a "+" node is $1/2$ in each case.

largest entries on their right-hand sides. Similarly, if d is even, then the player doing the search perceives his opponent as being the last player, so the depth $d-1$ probability vectors should tend to have their largest entries on their left-hand sides. As discussed in Chapter 4 and illustrated in Tables 4.1 through 4.5, this is indeed what happens. This phenomenon is again discussed in Chapter 7.

CHAPTER 6

PATHOLOGY ON $\Gamma(m,n)$ Introduction

Theorem 4.4 says that $\Gamma(1,1)$ is P-pathological for every imperfect P. It is reasonable to suspect that similar results hold for other trees as well. It would be ideal to have a theorem saying "given an arbitrary game tree, it is pathological if and only if . . ."

Unfortunately, proving such a theorem would be very difficult, if not impossible. One of the key steps in the proof of Theorem 4.4 was the representation of the probability vectors for the minimax values of the nodes of $\Gamma(1,1)$ by recursive formulas which were independent of the nodes of $\Gamma(1,1)$. According to Corollary 3.1.2, this can be done for each of the $\Gamma(m,n)$, but it cannot be done for most other game trees because of their irregular structure. For this reason, the mathematical development in the current chapter is directed solely at the $\Gamma(m,n)$. The question of how the results apply to other game trees is discussed in Chapter 8.

In this chapter, it is shown that for almost all probability vectors P, all but finitely many of the $\Gamma(m,n)$ are P-pathological. This result, which is stated formally by Theorem 6.4 and its corollaries, is the most important theoretical result of the dissertation.

To discuss this result in more detail, we need the following definition.

Definition 6.1. An \hat{f} -vector P is natural if

$$\min \{i \in \hat{f}: (P)_i > 0\} = r - \max \{i \in \hat{f}: (P)_i > 0\}.$$

If P is not natural, then P is unnatural.

According to this definition, a probability vector is natural if it has the same number of zero entries at each end. For example, $(0.2, 0.5, 0.3)$ and $(0, 0, 0.2, 0.5, 0.3, 0, 0)$ are natural, but $(0, 0.2, 0.5, 0.3, 0, 0)$ is not.

If P is unnatural, then certainly either $(P)_0 = 0$ or $(P)_r = 0$. This means that the set of unnatural \hat{f} -vectors is a proper subset of the union of two simplexes of $r-1$ dimensions each (for example, see Figure 6.1). But the set of all \hat{f} -vectors forms a simplex of r dimensions (Figure 6.2), and the union of the two $r-1$ dimensional simplexes is of measure zero in the r dimensional one. This proves the following.

Proposition 6.1. Almost all \hat{f} -vectors are natural.

The importance of Proposition 6.1 is that Theorem 6.4 holds for every natural \hat{f} -vector. This means that if an \hat{f} -vector is chosen from any continuous p.d.f. over the set of all \hat{f} -vectors, the probability that Theorem 6.4 does not apply to it is 0.

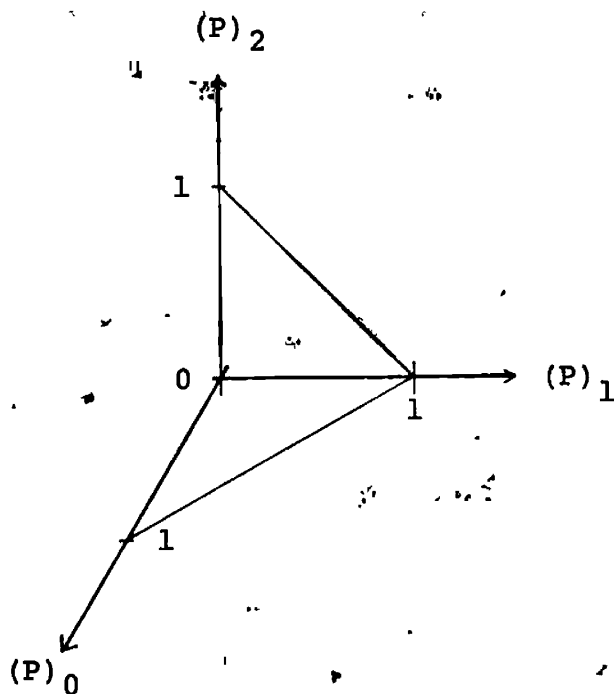


FIGURE 6.1.--The set of all unnatural 2-vectors is a proper subset of the union of the two line segments.

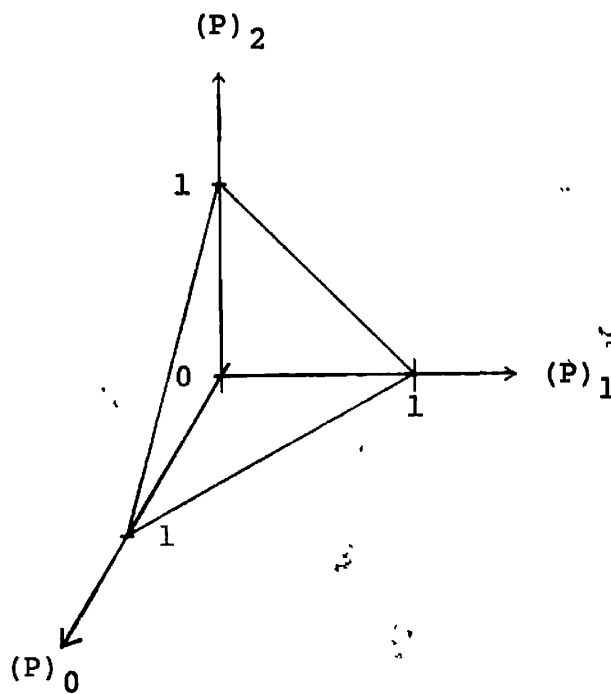


FIGURE 6.2.--The set of all 2-vectors forms a triangular plane segment.

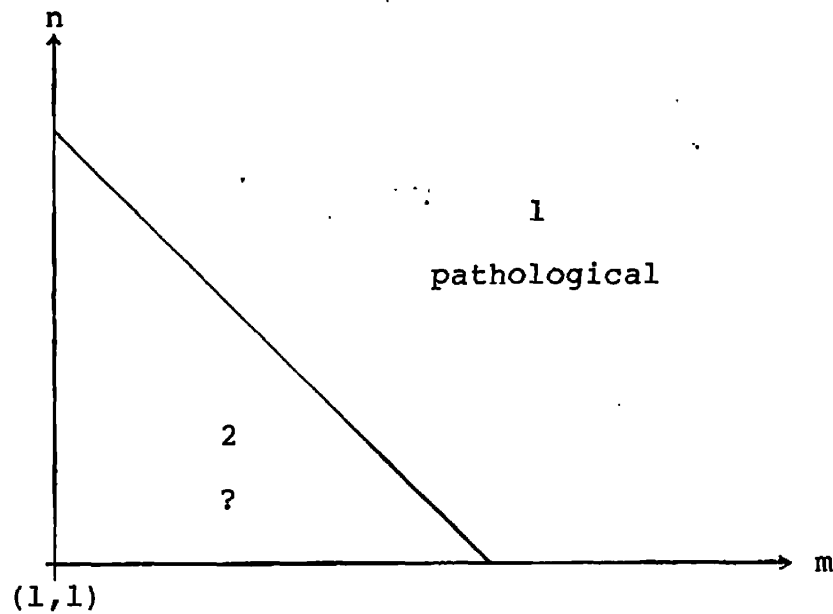


FIGURE 6.3.--Pathological and nonpathological behavior of $\Gamma(m,n)$ for a natural \hat{p} -vector P , as a function of m and n . The numbered areas are for reference in the text.

Theorem 6.4 is illustrated graphically in Figure 6.3, in which every tree $\Gamma(m,n)$ is represented by the pair of integers (m,n) . Area 1 is the area of pathology predicted by the theorem, 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, and the results of this experimentation are discussed in Chapter 7.

Theorem 6.4 requires a number of preliminary results, which appear in the next two sections. The final section of this chapter consists of the proof of Theorem 6.4.

Theorems about the Probability of Correct Decision

The theorems in this section provide a way to prove pathology theorems about $\Gamma(m,n)$, by providing conditions under

which the formula for the probability of correct decision given in Theorem 3.2 converges to $\frac{m}{m+n}$.

Theorem 6.1. Let P be an \hat{r} -vector, and suppose that $(P)_t = 1$ for some $t \in \hat{r}$. Let the function A be as in Theorem 3.2, and let $m \geq 1$ and $n \geq 1$ be integers. Then

$$\sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r A(m,i,k,P) A(n,j,k,P) = \frac{m}{m+n}.$$

Proof. In the definition of A , 0^0 is taken to be 1. This fact is implicitly used throughout the proof.

If $k \in \hat{r}$, then $(P)_k = 0$ if $k \neq t$, and $(P)_{0,k-1} = 0$ if $k = t$. Thus for $i = 1, 2, \dots, m-1$ and $k \in \hat{r}$, if $k \neq t$, then $A(m,i,k,P) = \binom{m}{i} ((P)_k)^i ((P)_{0,k-1})^{m-i} = \binom{m}{i} 0^i ((P)_{0,k-1})^{m-i} = 0$, and if $k = t$, then

$$A(m,i,k,P) = \binom{m}{i} ((P)_k)^i ((P)_{0,k-1})^{m-i} = \binom{m}{i} 1^i 0^{m-i} = 0.$$

Similarly, for $j = 1, 2, \dots, n-1$ and $k \in \hat{r}$,

$$A(n,j,k,P) = \binom{n}{j} ((P)_k)^j ((P)_{0,k-1})^{n-j} = 0.$$

$$\text{Therefore, } \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r A(m,i,k,P) A(n,j,k,P)$$

$$= \sum_{j=0}^n \frac{m}{m+j} \sum_{k=0}^r A(m,m,k,P) A(n,j,k,P)$$

$$= \frac{m}{m+0} \sum_{k=0}^r A(m,m,k,P) A(n,0,k,P) + \frac{m}{m+n} \sum_{k=0}^r A(m,m,k,P) A(n,n,k,P).$$

But for every $k \in \hat{r}$, if $k \neq t$, then

$$A(m,m,k,P) = \binom{m}{m} ((P)_k)^m ((P)_{0,k-1})^0 = (1) 0^m ((P)_{0,k-1})^0 = 0.$$

Therefore,

$$\frac{m}{m+0} \sum_{k=0}^r A(m,m,k,P) A(n,0,k,P) + \frac{m}{m+n} \sum_{k=0}^r A(m,m,k,P) A(n,n,k,P)$$

$$\begin{aligned}
 &= \frac{m}{m+0} A(m, m, t, P) A(n, 0, t, P) + \frac{m}{m+n} A(m, m, t, P) A(n, n, t, P) \\
 &= \frac{m}{m+0} \binom{m}{m} ((P)_t)^m ((P)_{0, t-1})^0 \binom{n}{0} ((P)_t)^0 ((P)_{0, t-1})^n \\
 &\quad + \frac{m}{m+n} \binom{m}{m} ((P)_t)^m ((P)_{0, t-1})^0 \binom{n}{n} ((P)_t)^n ((P)_{0, t-1})^0 \\
 &= \frac{m}{m+0} (1) 1^m 0^0 (1) 1^0 0^n + \frac{m}{m+n} (1) 1^m 0^0 (1) 1^n 0^0 \\
 &= 0 + \frac{m}{m+n} = \frac{m}{m+n}.
 \end{aligned}$$

△

Corollary 6.1.1. Let P be as in Theorem 6.1. If

$\lim_{d \rightarrow \infty} X_d = P$ and $\lim_{d \rightarrow \infty} Y_d = P$, then

$$\lim_{d \rightarrow \infty} \sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r A(m, i, k, X_d) A(n, j, k, Y_d) = \frac{m}{m+n}.$$

Proof.
$$\sum_{i=1}^m \sum_{j=0}^n \frac{i}{i+j} \sum_{k=0}^r A(m, i, k, X) A(n, j, k, Y)$$

is a continuous function of the probability vectors X and Y .

△

Theorem 6.2. Let $m \geq 1$ and $n \geq 1$ be integers, P be an \hat{f} -vector, and $S_d, T_d, U_d,$ and V_d be as in Notation 3.2. Suppose there are numbers $h \in \hat{f}$ and $k \in \hat{f}$ such that

$$(6.1) \quad \lim_{d \rightarrow \infty} (S_{2d})_{0, h} = 1 \quad \text{and} \quad \lim_{d \rightarrow \infty} (S_{2d})_{0, h-1} = 0,$$

$$(6.2) \quad \lim_{d \rightarrow \infty} (T_{2d})_{k, r} = 1 \quad \text{and} \quad \lim_{d \rightarrow \infty} (T_{2d})_{k+1, r} = 0,$$

$$(6.3) \quad \lim_{d \rightarrow \infty} (U_{2d})_{k, r} = 1 \quad \text{and} \quad \lim_{d \rightarrow \infty} (U_{2d})_{k+1, r} = 0,$$

and

$$(6.4) \quad \lim_{d \rightarrow \infty} (V_{2d})_{0, h} = 1 \quad \text{and} \quad \lim_{d \rightarrow \infty} (V_{2d})_{0, h-1} = 0.$$

Then $\Gamma(m, n)$ is P -pathological.

Proof. From Definition 3.5 and statements (6.2) and (6.3),

$$\lim_{d \rightarrow \infty} (T_{2d})_k = \lim_{d \rightarrow \infty} (T_{2d})_{k,r} - (T_{2d})_{k+1,r} = 1$$

and $\lim_{d \rightarrow \infty} (U_{2d})_k = \lim_{d \rightarrow \infty} (U_{2d})_{k,r} - (T_{2d})_{k+1,r} = 1.$

Therefore, from Notation 3.3, Theorem 3.2, and Corollary 6.1.1,

$$(6.5) \quad \lim_{d \rightarrow \infty} D_{m,n}(P, 2d+1) = \frac{m}{m+n}.$$

From Definition 3.5, Corollary 4.1.4, and statement (6.1),

$$\begin{aligned} \lim_{d \rightarrow \infty} (T_{2d+1})_h &= \lim_{d \rightarrow \infty} (T_{2d+1})_{0,h} - \lim_{d \rightarrow \infty} (T_{2d+1})_{0,h-1} \\ &= 1 - (1 - \lim_{d \rightarrow \infty} (S_{2d})_{0,h})^{m+n} - 1 + (1 - \lim_{d \rightarrow \infty} (S_{2d})_{0,h-1})^{m+n} \\ &= 1 - (1-1)^{m+n} - 1 + (1-0)^{m+n} = 1. \end{aligned}$$

From Definition 3.5, Corollary 4.1.4, and statements (6.1) and (6.4),

$$\begin{aligned} \lim_{d \rightarrow \infty} (U_{2d+1})_h &= \lim_{d \rightarrow \infty} (U_{2d+1})_{0,h} - \lim_{d \rightarrow \infty} (U_{2d+1})_{0,h-1} \\ &= 1 - (1 - \lim_{d \rightarrow \infty} (V_{2d})_{0,h})^m (1 - \lim_{d \rightarrow \infty} (S_{2d})_{0,h})^n \\ &\quad - 1 + (1 - \lim_{d \rightarrow \infty} (V_{2d})_{0,h-1})^m (1 - \lim_{d \rightarrow \infty} (S_{2d})_{0,h-1})^n \\ &= 1 - (1-1)^m (1-1)^n - 1 + (1-0)^m (1-0)^n = 1. \end{aligned}$$

Therefore, from Notation 3.3, Theorem 3.2, and Corollary 6.1.1,

$$(6.6) \quad \lim_{d \rightarrow \infty} D_{m,n}(P, 2d) = \frac{m}{m+n}.$$

From equations (6.5) and (6.6), $\lim_{d \rightarrow \infty} D_{m,n}(P, d) = \frac{m}{m+n}.$

△

Double Sequences

Because of Corollary 4.1.5, the proof of Theorem 6.4 depends on the properties of double sequences of the form

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

$$y_{i+1} = (1 - (1-y_i)^{m+n})^m (1 - (1-x_i)^m (1-y_i)^n)^n.$$

This section contains several results which provide tools to deal with these sequences. The results are stated using the following notation.

Notation 6.2. x_0 and y_0 are always real numbers in the interval $[0,1]$. For every $m \geq 1$ and $n \geq 1$, $x_{m,n,0} = x_0$ and $y_{m,n,0} = y_0$. For every $m \geq 1$, $n \geq 1$, and integer $i \geq 0$,

$$x_{m,n,i+1} = (1 - (1-x_{m,n,i})^m (1-y_{m,n,i})^n)^{m+n}$$

and

$$y_{m,n,i+1} = (1 - (1-y_{m,n,i})^{m+n})^m (1 - (1-x_{m,n,i})^m (1-y_{m,n,i})^n)^n.$$

The following proposition has a simple inductive proof.

The theorem following it is a little more complicated.

Proposition 6.2. Let $m \geq 1$ and $n \geq 1$. Then $0 \leq x_0 \leq 1$ and $0 \leq y_0 \leq 1$ for every integer $i \geq 0$. If $y_0 = 0$, then $y_{m,n,i} = 0$ for every integer $i \geq 0$. If $x_0 = y_0 = 0$, then $x_{m,n,i} = y_{m,n,i} = 0$ for every integer $i \geq 0$. If $y_0 = 1$, then $x_{m,n,i} = y_{m,n,i} = 1$ for every integer $i \geq 1$.

Theorem 6.3. Let $m \geq 1$, $n \geq 1$, $x_0 > w_{m+n}$, and $y_0 > w_{m+n}$.

Then

$$\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 1.$$

Proof. Let $z_0 = \min(x_0, y_0)$, and for every integer $i \geq 0$,

let

$$z_{i+1} = (1 - (1-z_i)^{m+n})^{m+n}.$$

$w_{m+n} < z_0 \leq 1$, so from Theorem 5.2, $\lim_{i \rightarrow \infty} z_i = 1$.

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

$$z_i \leq x_{m,n,i} \leq 1 \quad \text{and} \quad z_i \leq y_{m,n,i} \leq 1.$$

This statement holds for $i = 0$, because $z_0 = \min(x_{m,n,0}, y_{m,n,0})$.

Suppose it holds for $i = k$. Then

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

and

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

From Proposition 6.2, $x_{k+1} \leq 1$ and $y_{k+1} \leq 1$. Therefore, the statement holds for $i = k+1$.

From the above, it follows that

$$\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 1.$$

△

Corollary 6.3.1. Let $x_0 > 0$ and $y_0 > 0$. Then there is an L such that whenever $m \geq 1$, $n \geq 1$, and $m+n > L$,

$$\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 1.$$

Proof. From Corollary 5.1.3, there is an L such that for every $i > L$, $w_i < \min(x_0, y_0)$. Let $m \geq 1$ and $n \geq 1$ be such that $m+n > L$. Then from Theorem 6.3,

$$\lim_{i \rightarrow \infty} x_{m,n,i} = y_{m,n,i} = 1.$$

△

The Main Theorem

The results of the last two sections provide the tools necessary to prove Theorem 6.4. The meaning of this theorem was discussed in the introduction to this chapter, and the practical significance of the theorem is discussed in subsequent chapters.

Theorem 6.4. Let P be a natural \hat{f} -vector. Let $m \geq 1$, $n \geq 1$, and

$$h = \min \{i \in \hat{f}: (P)_i > 0\}.$$

If $w_{m+n} < \min ((P)_h, (P)_{r-h})$, then $\Gamma(m, n)$ is P -pathological.

Proof. Suppose $w_{m+n} < \min ((P)_h, (P)_{r-h})$. Then from Corollary 3.1.1 and Definitions 2.9 and 3.5,

$$(U_0^{m,n})_{r-h,r} = (S_0^{m,n})_{0,h} = (P)_{0,h} > w_{m+n},$$

and $(V_0^{m,n})_{0,h} = (T_0^{m,n})_{r-h,r} = (P)_{r-h,r} > w_{m+n}$,

so from Corollary 4.1.5 and Theorem 6.3,

$$\lim_{d \rightarrow \infty} (T_{2d}^{m,n})_{r-h,r} = \lim_{d \rightarrow \infty} (U_{2d}^{m,n})_{r-h,r} = 1$$

and $\lim_{d \rightarrow \infty} (V_{2d}^{m,n})_{0,h} = \lim_{d \rightarrow \infty} (S_{2d}^{m,n})_{0,h} = 1$.

But according to Definition 6.1, $\max \{i \in \hat{f}: (P)_i > 0\} = r-h$, whence

$$(P)_{0,h-1} = (P)_{r-h+1,r} = 0.$$

Thus from Corollary 3.1.1 and Definitions 2.9 and 3.5,

$$(U_0^{m,n})_{r-h+1,r} = (S_0^{m,n})_{0,h-1} = (P)_{0,h-1} = 0$$

and $(V_0^{m,n})_{0,h-1} = (T_0^{m,n})_{r-h+1,r} = (P)_{r-h+1,r} = 0$,

so from Corollary 4.1.5 and Proposition 6.2,

$$(T_{2d}^{m,n})_{r-h+1,r} = (U_{2d}^{m,n})_{r-h+1,r} = 0$$

and $(V_{2d}^{m,n})_{0,h-1} = (S_{2d}^{m,n})_{0,h-1} = 0$

for every integer $d \geq 0$. Thus from Theorem 6.2, $\Gamma(m, n)$ is P -pathological.

△

Corollary 6.4.1. Let P be a natural \hat{f} -vector. Then there is an L such that for all positive integers m and n such

that $m+n > L$, $\bar{\Gamma}(m,n)$ is P -pathological.

Proof. Let h be as in Theorem 6.4. Then from Definition 6.1 and Corollary 5.1.3, there is an L such that for all positive integers m and n such that $m+n > L$, $w_{m+n} < \min((P)_h, (P)_{r-h})$, whence $\bar{\Gamma}(m,n)$ is P -pathological by Theorem 6.4.

△

Corollary 6.4.2. If P is a natural \hat{r} -vector, then all but finitely many of the $\bar{\Gamma}(m,n)$ are P -pathological.

Proof. Immediate from Corollary 6.4.1.

△

CHAPTER 7

EXPERIMENTAL RESULTS

Introduction

As with Theorem 4.4, Theorem 6.4 has been verified in numerous experimental tests. Indeed, it was by conducting such experiments that the author gained the mathematical intuitions which led to the proofs of both of these theorems.

Several tables and figures were presented at the end of Chapter 4 which illustrated the convergence of $D(P,d)$ to $\frac{1}{2}$ on $\Gamma(1,1)$. For $\Gamma(m,n)$, the limiting value is $\frac{m}{m+n}$ rather than $\frac{1}{2}$, but otherwise the behavior is much the same, and it would be redundant to present any more such tables here. We instead discuss which of the $\Gamma(m,n)$ are pathological for a given probability vector, and the various ways in which this occurs.

In addition to serving as a demonstration of Theorem 6.4, the experimental results presented here lend themselves naturally to discussions of (1) the importance of having an evaluation function which is capable of making fine distinctions of quality among different game positions, (2) how the shape of the set of nonpathological $\Gamma(m,n)$ relates to our conception of what an evaluation function does, (3) the limiting values of the probability vectors for minimax values, and (4) the meaning of the rate of convergence of $D(P,d)$. Each of these topics is dealt with in a separate section of this chapter.

The Importance of a Good Evaluation Function

Consider an evaluation function which returns values in the range $\{5,6,\dots,94\}$. If the returned values of this function are rounded to the nearest multiple of 10, the resulting function is equivalent to one which returns values in the range $\{1,2,\dots,10\}$. This function evaluates game positions in the same way as the original one, but in a rougher manner. Therefore, we would expect the performance of the new function to be no better, and possibly poorer, than the performance of the first one. Just how much poorer is dramatically illustrated below.

As an example, consider the probability vector

$$P_1 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12) / 78.$$

If e is a P_1 -function, then repeated rounding operations on the values returned by e produce evaluation functions having probability vectors

$$P_2 = (1+2, 3+4, 5+6, 7+8, 9+10, 11+12) / 78,$$

$$P_3 = (1+2+3, 4+5+6, 7+8+9, 10+11+12) / 78,$$

$$P_4 = (1+2+3+4, 5+6+7+8, 9+10+11+12) / 78,$$

and $P_5 = (1+2+3+4+5+6, 7+8+9+10+11+12) / 78.$

These vectors all give similar performance on $\Gamma(1,1)$, as shown in Table 7.1. Without any further information than this, one might think that they would all perform similarly on each of the $\Gamma(m,n)$. However, this is not the case. The first indication of this comes from Theorem 6.4.

Theorem 6.4 states that for every natural probability vector P , $\Gamma(m,n)$ is P -pathological whenever w_{m+n} is less than the minimum t of the leftmost and rightmost nonzero elements of P . But according to Corollary 5.1.3, w_{m+n} decreases monotonically as

TABLE 7.1.--Probabilities of correct decision on $\Gamma(1,1)$ at various search depths for each of $P_1, P_2, P_3, P_4,$ and P_5 .

d	D(P_1, d)	D(P_2, d)	D(P_3, d)	D(P_4, d)	D(P_5, d)
1	0.806	0.799	0.788	0.774	0.731
2	0.746	0.739	0.727	0.711	0.669
3	0.687	0.680	0.671	0.659	0.635
4	0.635	0.628	0.618	0.607	0.585
5	0.592	0.585	0.576	0.564	0.566
6	0.560	0.554	0.546	0.539	0.535
7	0.538	0.533	0.529	0.515	0.528
8	0.524	0.518	0.513	0.510	0.512
9	0.514	0.511	0.510	0.502	0.510
10	0.508	0.504	0.503	0.501	0.503
11	0.505	0.503	0.503	0.500	0.503
12	0.503	0.501	0.501	0.500	0.501
13	0.501	0.500	0.500	0.500	0.500
14	0.501	0.500	0.500	0.500	0.500
15	0.500	0.500	0.500	0.500	0.500
16	0.500	0.500	0.500	0.500	0.500

TABLE 7.2.--The number of trees $\Gamma(m,n)$ which Theorem 6.4 says need not be pathological, for each of $P_1, P_2, P_3, P_4,$ and P_5 .

Probability vector	Minimum of the leftmost and the rightmost entries	Smallest k for which $w_k < t$ (see Table 5.1)	Number of trees which need not be pathological
P1	t = 0.01282	338	56616
P2	t = 0.03846	84	3403
P3	t = 0.07692	33	496
P4	t = 0.12821	15	91
P5	t = 0.26923	5	6

$m+n$ increases. This means that pathology must occur on every $\Gamma(m,n)$ for which $m+n \geq k$, where k is the smallest integer such that $w_k < t$. If $m+n \leq k-1$, then pathology may or may not occur. Thus the number of trees which need not be pathological is

$$1 + 2 + \dots + k-2 = \frac{(k-1)(k-2)}{2}.$$

For P_1 through P_5 , Table 7.2 gives the number of trees which need

not be pathological. This table indicates that the rounding operations which produced P_2 through P_5 yielded successively poorer probability vectors.

Tables 7.3 through 7.7, which were computed using the computer program of Appendix H, tell which of the $\Gamma(m,n)$ are P_1 -, P_2 -, P_3 -, P_4 -, and P_5 -pathological, for $m = 1, 2, \dots, 22$ and $n = 1, 2, \dots, 20$. As can be seen from these tables, pathology occurs everywhere that Theorem 6.4 says that it must occur. It occurs in many other places as well, but the the predictions made in Table 7.2 are good relative indicators of the numbers of nonpathological trees: the number of non- P_i -pathological $\Gamma(m,n)$ decreases markedly as i increases. At least several hundred and probably several thousand of the $\Gamma(m,n)$ are nonpathological for P_1 , whereas none of them are nonpathological for P_5 . This dramatically demonstrates the importance of using an evaluation function with a large effective range.

TABLE 7.3.--Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_1 = (1,2,3,4,5,6,7,8,9,10,11,12)/78$.

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
m																					
1	5*	4*	4*	4*	4*	4*	4*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*
2	4*	4*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	3*	2*	2*	2*	2*	2*	2*	2*
3	25	34	3*	3*	3*	3*	3*	3*	3*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*
4	25	24	23	23	23	23	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*
5	14	24	24	23	23	23	23	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*
6	14	14	14	13	23	23	23	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*	2*
7	14	14	13	13	13	13	13	13	12	12	12	12	12	12	2*	2*	2*	2*	2*	2*	2*
8	14	14	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
9	14	14	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
10	14	14	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
11	14	13	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
12	04	13	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
13	04	03	13	13	13	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
14	03	03	03	03	03	13	13	13	12	12	12	12	12	12	12	12	12	12	12	12	12
15	03	03	03	03	03	03	03	03	02	12	12	12	12	12	12	12	12	12	12	12	12
16	03	03	03	03	03	03	03	03	02	02	02	02	02	02	12	12	12	12	12	12	12
17	03	03	03	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02
18	03	03	03	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02
19	03	03	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02	02
20	03	03	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02	02
21	03	03	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02	02
22	03	03	03	03	03	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_1, d)$ converges to 1. The additional information contained in this table is explained later in the chapter.

TABLE 7.4.--Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_2 = (3,7,11,15,19,23)/78$.

$n =$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
m	2* 2*	2* 2* 1*	12 12 1*	12 12 1*	12 12 1*	02 12 12 1*	02 02 02 01 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1* 1*	02 02 01	02 02 01	02 02 01	02 01	02 01	02 01	01 01	01 01	01 01	01 01	01 01	01 01	01 01	01 01	

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_2, d)$ converges to 1. The additional information contained in this table is explained later in the chapter.

TABLE 7.5.--Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_3 = (6,15,24,33)/78$.

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
m																					
1	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*
2	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*
3	01	1*	1*	1*	1*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
4	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
5	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
6	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
7	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
8	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
9	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
10	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
11	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
12	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
13	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
14	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
15	01	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
16	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
17	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
18	01	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
19	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
20	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
21	01	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
22	01	01	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_3,d)$ converges to 1. Below the diagonal line is where Theorem 6.4³ says that pathology must occur (see Table 7.2). The additional information contained in this table is explained later in the chapter.

TABLE 7.6.--Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_4 = (10,26,42)/78$.

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
m																					
1	1*	1*	1*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
2	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
3	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
4	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
5	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
6	01	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
7	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
8	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
9	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
10	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
11	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
12	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
13	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
14	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
15	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
16	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
17	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
18	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
19	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
20	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
21	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
22	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_4,d)$ converges to 1. Below the diagonal line is where Theorem 6.4 says that pathology must occur (see Table 7.2). The additional information contained in this table is explained later in the chapter.

TABLE 7.7.--Pathology of $\Gamma(m,n)$ as a function of m and n , for the probability vector $P_5 = (21,57)/78$.

m	n = 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
2	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
3	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
4	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
5	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
6	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
7	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
8	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
9	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
10	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
11	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
12	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
13	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
14	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
15	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
16	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
17	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
18	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
19	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
20	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
21	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
22	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_5, d)$ converges to 1. Below the diagonal line is where Theorem 6.4₅ says that pathology must occur (see Table 7.2). The additional information contained in this table is explained later in the chapter.

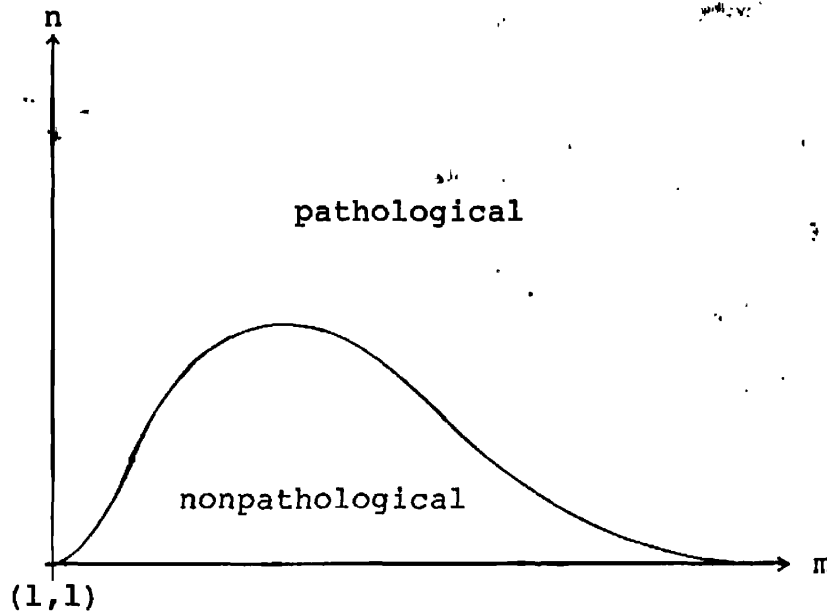


FIGURE 7.1.--Pathological and nonpathological behavior of $\Gamma(m, n)$ for a natural probability vector P , as a function of m and n .

The Shape of the Nonpathological Region

Theorem 6.4 and its corollaries state that for every natural \hat{p} -vector P , $\Gamma(m, n)$ is P -pathological whenever the branching factor $m+n$ is sufficiently large. Tables 7.3 through 7.7 show that pathology occurs elsewhere as well; indeed, the shape of the region where pathology does not occur is roughly as shown in Figure 7.1. Not only does pathology occur when $m+n$ is large, but it also occurs whenever the ratio of m to 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 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 $\Gamma(m,n)$ 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 surprising that pathology would occur.

The Values of the Limiting Vectors

When Theorem 6.4 predicts pathology, the proof of the theorem shows that the pathology occurs because T_{2d} and U_{2d} both converge to the probability vector $(0,0,\dots,0,1)$ and T_{2d+1} and U_{2d+1} both converge to the probability vector $(1,0,0,\dots,0)$. This is the "manic-depressive" behavior described in Chapters 4 and 5 in its most extreme form! In places where Theorem 6.4 does not predict pathology, the limiting vectors may be different, but the same kind of behavior still occurs.

Let g be a critical node. Choosing a move from g using a depth d minimax search involves computing the minimax values $\{e_{d-1}(g') : g' \text{ is a child of } g\}$. For the purposes of the following discussion, we may without loss of generality assume that g is an S node (see Corollary 3.1.1). In this case, each of the minimax values has either T_{d-1} or U_{d-1} as its probability

vector. In practice, as d increases, these vectors converge to vectors of the form $(0, \dots, 0, 1, 0, \dots, 0)$. More specifically, if we let Q_i be the \hat{r} -vector whose i 'th entry is 1, then there are integers $i \leq \frac{r}{2}$ and $j \leq \frac{r}{2}$ such that (1) for d even, T_{d-1} converges to Q_i and U_{d-1} converges to Q_j , and (2) for d odd, T_{d-1} converges to $\text{rev } Q_j$ and U_{d-1} converges to $\text{rev } Q_i$. What this means is that for even search depths, every move tends to look bad, and for odd search depths, every move tends to look good. This "manic-depressive" behavior, which was also discussed in Chapters 4 and 5, has been noticed by writers of chess and checker playing computer programs. Indeed, the term "manic-depressive" was suggested to the author by Thomas Truscott [TR1], a co-author of the prize-winning chess playing computer program Duchess [TR2, RO1].

If $\Gamma(m,n)$ is pathological, then $i = j$. In this case, as the search depth increases, all moves tend to look more and more alike, so the play becomes increasingly random.

All of this is illustrated in Tables 7.3 through 7.7. These tables provide information on the values of the vectors Q_i and Q_j . For trees which are not pathological, the first digit of each table entry is the value of i and the second digit is the value of j . If a tree is pathological, then $i = j$, so i is given and the pathology is indicated with an asterisk.

In Table 7.3, for example, the entry for $\Gamma(3,1)$ is 25. This means that

$$\lim_{d \rightarrow \infty} T_{2d-1}^{(3,1)} = (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),$$

$$\lim_{d \rightarrow \infty} U_{2^{d-1}}^{(3,1)} = (0,0,0,0,0,0,1,0,0,0,0,0,0),$$

$$\lim_{d \rightarrow \infty} T_{2^d}^{(3,1)} = (0,0,0,0,0,0,0,0,0,0,1,0,0),$$

and

$$\lim_{d \rightarrow \infty} U_{2^d}^{(3,1)} = (0,0,0,0,0,0,0,1,0,0,0,0,0).$$

It can easily be seen from the proof of Theorem 6.4 that $i = j = 0$ whenever the theorem predicts pathology. Therefore, the table entry in such a case should be "0*". Inspection of the tables reveals that this is indeed so.

Rates of Convergence

As pointed out earlier in this chapter, the rate of convergence of $D(P_i, d)$ on $\Gamma(1,1)$ is not indicative of the quality of P_i . We now discuss what the rate of convergence does indicate.

Let i and j be as in the previous section. Suppose that $i = i_0$ and $j = j_0$ for some tree $\Gamma(m_0, n_0)$, and that $i = i_0 - 1$ and $j = j_0$ for the tree $\Gamma(m_0 + 1, n_0)$. If we consider the vectors $T_d^{m,n}$ and $U_d^{m,n}$ independent of game trees and extend their definitions to non-integer values of m and n , then there should be some point $m \in (m_0, m_0 + 1)$ where the limiting value of $T_{2^{d-1}}^{m, n_0}$ changes from Q_{i_0} to $Q_{i_0 - 1}$. Since m_0 and $m_0 + 1$ are both quite close to m , we would expect the T and U vectors for $\Gamma(m_0, n_0)$ and $\Gamma(m_0 + 1, n_0)$ to converge more slowly than usual. Similar considerations hold when m, n, i and j are varied in other ways.

The point of the above discussion is that when two adjacent table entries differ, the rates of convergence of the probability vectors (and hence of the probabilities of correct decision) should be slower than elsewhere. This has been

verified experimentally, as discussed below.

Let us define d_0 to be the smallest value of d such that for every $d \geq d_0$, every entry of each of the vectors T_{2d} , T_{2d+1} , U_{2d} , and U_{2d+1} is within 0.0005 of its limiting value. Then d_0 is a measure of the rate of convergence of the T and U vectors, and hence by Corollary 6.1.1 it is a measure of the rate of convergence of $D(P,d)$. Tables 7.8 through 7.12 give the values of d_0 corresponding to the entries in Tables 7.3 through 7.7.

As can be seen in Tables 7.8 through 7.12, d_0 is large whenever the corresponding entry from Tables 7.3 through 7.7 is different from one of its neighbors. This is in accordance with the above discussion, thus providing evidence that the rate of convergence of $D(P,d)$ is not an indication of the quality of a probability vector, but is instead an indication of being near a point where a limit changes.

TABLE 7.8.-- d_0 as a function of m and n , for the probability vector $P_1 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)/78$.

m	$n = 1$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	18	19	9	8	8	8	10	9	7	7	5	5	5	5	5	6	6	6	6	6
2	16	12	19	9	7	7	6	6	6	6	6	6	6	8	11	7	7	5	5	5
3	14	10	9	8	8	8	8	8	8	9	7	7	5	5	5	5	5	5	5	5
4	12	8	11	9	9	9	11	7	7	7	5	5	5	5	5	5	5	5	5	4
5	13	10	16	7	6	8	12	7	7	6	6	5	5	5	5	5	5	5	4	4
6	9	7	12	9	10	8	8	9	7	7	6	6	6	6	6	6	6	6	6	6
7	7	6	17	7	7	7	8	12	7	7	9	9	11	13	10	10	10	10	10	12
8	6	6	9	7	6	6	6	8	7	7	5	5	5	5	5	5	6	6	6	6
9	6	8	7	5	6	6	6	8	7	7	5	5	5	5	5	5	5	6	6	6
10	6	8	7	5	5	6	6	8	9	7	5	5	5	5	5	5	5	5	6	6
11	8	9	7	6	6	6	6	8	9	7	5	5	5	5	5	5	5	4	4	6
12	11	8	6	6	6	6	6	8	7	7	5	5	5	5	5	5	5	4	4	4
13	12	7	12	8	6	6	6	8	7	7	6	5	5	5	5	5	4	4	4	4
14	7	5	7	7	9	8	8	8	7	6	6	6	6	6	6	6	4	4	4	4
15	7	5	5	5	6	7	7	8	9	8	8	6	6	6	6	6	6	6	6	6
16	5	5	5	5	6	6	6	9	7	7	7	7	7	9	10	8	8	6	6	6
17	5	5	5	5	6	6	6	7	5	5	5	5	5	5	5	7	7	7	7	7
18	5	5	5	5	6	6	8	7	5	5	5	5	5	5	5	5	5	5	5	5
19	5	5	5	6	6	6	9	7	5	5	5	5	5	5	5	5	5	5	5	5
20	5	5	4	6	6	8	7	5	5	5	5	5	5	5	5	5	5	5	5	5
21	5	4	4	6	6	10	7	5	5	5	5	5	5	4	5	5	5	5	5	5
22	4	4	6	6	6	7	5	5	5	5	5	5	5	4	4	4	4	4	4	4

TABLE 7.10.-- d_0 as a function of m and n , for the probability vector $P_3 = (6,15,24,33)/78$.

m	$n = 1$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	15	9	7	6	5	5	4	4	4	4	4	4	4	4	4	6	6	6	6	6
2	10	8	6	6	6	6	6	6	6	6	6	6	6	8	11	7	7	5	5	5
3	11	10	8	8	8	8	8	8	8	9	7	7	5	5	5	5	5	5	5	5
4	7	7	7	9	9	9	11	7	7	7	5	5	5	5	5	5	5	5	5	3
5	5	5	6	6	6	8	12	7	7	5	5	5	5	5	5	5	5	5	3	3
6	5	5	5	6	6	6	8	9	7	7	5	5	5	5	5	5	5	3	3	3
7	5	5	4	4	6	6	8	12	7	7	5	5	5	5	5	5	5	3	3	3
8	5	4	4	4	6	6	6	8	7	7	5	5	5	5	5	5	5	3	3	3
9	4	4	4	4	6	6	6	8	7	7	5	5	5	5	5	5	5	3	3	3
10	4	4	4	4	4	6	6	8	9	7	5	5	5	5	5	5	5	3	3	3
11	4	4	4	4	4	6	6	8	9	7	5	5	5	5	5	5	5	3	3	3
12	4	4	4	4	4	6	6	8	7	7	5	5	5	5	5	5	5	3	3	3
13	4	4	4	4	4	6	6	8	7	7	5	5	5	5	5	5	3	3	3	3
14	4	4	4	4	4	6	6	8	7	5	5	5	5	5	5	5	3	3	3	3
15	4	4	4	4	6	6	6	8	7	5	5	5	5	5	5	5	3	3	3	3
16	4	4	4	4	6	6	6	9	7	5	5	5	5	5	5	5	3	3	3	3
17	4	4	4	4	6	6	6	7	5	5	5	5	5	5	5	3	3	3	3	3
18	4	4	4	4	6	6	8	7	5	5	5	5	5	5	5	3	3	3	3	3
19	4	4	4	6	6	6	9	7	5	5	5	5	5	5	3	3	3	3	3	3
20	4	4	4	6	6	8	7	5	5	5	5	5	5	5	3	3	3	3	3	3
21	4	4	4	6	6	10	7	5	5	5	5	5	5	3	3	3	3	3	3	3
22	4	4	6	6	6	7	5	5	5	5	5	5	5	3	3	3	3	3	3	3

CHAPTER 8

SUMMARY, CONCLUSIONS AND SPECULATIONS

Summary of Results

Game trees are a model of the problem reduction approach to decision making. Large game trees are physically impossible to search completely, so sometimes a limited search is done using a heuristic evaluation function to estimate the properties of the tree. The conventional belief about such a heuristic game tree search is "the deeper the search, the better the decision." We have examined this statement mathematically, to discover when and where it holds.

The mathematical model used for this investigation has the following major features: (1) the classification of the nodes of the game tree as critical nodes (nodes where it makes a difference what decision is made) and noncritical nodes (where the decision makes no difference), (2) a stochastic model of evaluation function errors in terms of a probability vector which characterizes the "average goodness" of the function, (3) the use of "minimaxing" (i.e., the maximin decision criterion), and (4) a characterization of the quality of a decision at a critical node as the probability that the decision is correct.

There is an infinite class of game trees (the trees $\Gamma(m,n)$ of Chapter 3) for which the probability of correct decision depends only on m , n , the search depth, and the evaluation function's probability vector. Theorem 6.4 proves

that these trees are pathological in the sense that for almost every probability vector and for all but finitely many of the $\Gamma(m,n)$, increasing the search depth does not improve the quality of the decision, but instead causes the decision's probability of correctness to converge to what it would be if moves were being made totally at random.

Experimental verification of this phenomenon is presented at the end of Chapter 4 and in Chapter 7, where it is pointed out that the decisions become random because of a "manic-depressive" kind of behavior: as the search depth increases, all moves tend to look alternately good and then bad, depending on whether the search depth is odd or even. The mathematical results in Chapter 5 provide intuitive motivation for why this occurs.

Conclusions and Speculations

As used in computer science, the problem reduction approach almost invariably involves computing either maxima and minima or Boolean sums and products. Both of these procedures are cases of the maximin decision criterion, which is one of several criteria which have been studied in decision analysis [LAI, TU1]. Most of these criteria involve maximization or minimization in a form similar enough to the maximin criterion that pathology theorems similar to those of this dissertation should hold for them as well.

To the reader who has studied the mathematical results in the preceding chapters, it will be obvious that they should extend to a considerably larger class of game trees than the $\Gamma(m,n)$. Most other game trees are complicated enough that

considerably more involved proof techniques may be required; this provides a topic for future research. We now discuss what significance the results have for game trees in general.

Judging from the success of chess playing programs such as Belle, Chess 4.7, and Duchess [BI2, RO1, TR1, TR2], it does not appear that pathology generally occurs for the evaluation functions that they use, if it occurs for them at all. But as mentioned in Chapter 7, the "manic-depressive" behavior which occurs on the $\Gamma(m,n)$ has also been observed to occur in computer programs which play chess, checkers, and other games. The author believes that this behavior is indicative of an underlying pathological tendency present in almost all game trees. This tendency appears to be overridden in most games by at least three other factors.

First, the experimental results in Chapter 7 indicate that if an evaluation function can make reasonably fine distinctions among the relative strengths of various game positions, then $\Gamma(m,n)$ is pathological only when the branching factor $m+n$ is quite large, or the ratio m/n is small. Neither of these conditions is likely to occur consistently in most applications of game trees, although they may occur to some extent.

Second, the $\Gamma(m,n)$ are "homogeneous" in the sense that every S node looks like every other S node, every T node looks like every other T node, and so forth. This certainly is not true of most other games: some S nodes, for example, will be "strong" positions in the sense that most of their children will also be forced win nodes, and some will be "weak" positions in

the sense that only a few of their children will be forced win nodes. A good evaluation function will probably tend to evaluate strong positions more highly than weak ones, in which case the player will tend to move into game positions where m/n is large, so that pathology is less likely to occur.

Third, pathology can only occur while neither player is searching to the end of the tree, for seeing the final results of a game allows perfect play. The player with the shallower search will have an advantage throughout most of a pathological game, but generally will not attain a totally unassailable position.¹ Near the end of the game, the player with the deeper search will achieve perfect play several moves before his opponent, and thus may be able to recoup his losses and win the game.

The above three factors are not well understood, and provide a topic for further investigation. Pathological behavior might possibly occur for limited periods of time even in games such as chess or checkers. In a situation where most moves look fairly similar and neither player has a distinct advantage, the game tree might look "homogeneous" enough for pathology to occur for a few moves. This is highly speculative.

There is a fourth significant difference between the $\Gamma(m,n)$ and most other games, but it is not at all clear how it affects the existence of pathology. This is that many games are more properly represented as directed graphs than as trees [LE1],

¹For the $\Gamma(m,n)$, this situation can be modeled by a Markov process, in which for each player there will be a steady state probability of his being at a forced win node when it is his move. The author has done some preliminary research on this topic, but not enough to warrant its presentation here.

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 the evaluation function incorporates a random number generator,² it should return the same value for each of these nodes. But this violates the assumption of stochastic independence of the errors made by the evaluation function. Just what effect this has needs further investigation.

A good example of this is Bachet's game [BA1, DO1, JA1]. This name refers to a class of games similar to nim. These games would be perfectly modeled as (m,n) -games (see Definition 3.3), except that the independence assumption is violated as explained above. Although the proof is not presented here, the author has shown that for Bachet's game, the probability of correct decision is completely independent of the search depth. This itself is an interesting contradiction to the commonly accepted beliefs about searching deeper.

At the beginning of Chapter 2 and throughout the current chapter, topics needing further investigation have been mentioned. Some of them involve generalizing the mathematical theory, and some of them involve investigating the practical significance of the theory. At this point, the latter type of investigation would appear to be a bit more important. Not much work has been done on this topic, and there are several methodological problems which have not yet been resolved. This is discussed at greater length in Appendix H.

²Actually, some writers of game playing 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!

The conclusion to be reached at this time is as follows. Pathology occurs almost universally on the $\Gamma(m,n)$, and almost certainly on many other game trees as well. How often it might occur in common applications of game trees is unclear, and there is reason to believe that it may be hard to observe even where it does occur. However, it is no longer possible blithely to assume (as has been done in the past) that increasing the depth of a heuristic game tree search will improve the quality of a decision.

APPENDIX A

AN EXTENTION OF TABLE 5.1

TABLE A.1.--Values of w_n to six decimal places, for $n = 1, 2, \dots, 350$.

n	w_n	n	w_n	n	w_n	n	w_n
1	0.500000	39	0.066972	77	0.040720	115	0.030025
2	0.381966	40	0.065775	78	0.040327	116	0.029825
3	0.317672	41	0.064626	79	0.039944	117	0.029629
4	0.275508	42	0.063521	80	0.039568	118	0.029436
5	0.245122	43	0.062459	81	0.039200	119	0.029245
6	0.221910	44	0.061436	82	0.038839	120	0.029057
7	0.203456	45	0.060450	83	0.038486	121	0.028872
8	0.188348	46	0.059500	84	0.038140	122	0.028689
9	0.175699	47	0.058582	85	0.037801	123	0.028509
10	0.164921	48	0.057697	86	0.037469	124	0.028331
11	0.155602	49	0.056841	87	0.037143	125	0.028156
12	0.147449	50	0.056013	88	0.036823	126	0.027983
13	0.140243	51	0.055213	89	0.036510	127	0.027813
14	0.133819	52	0.054437	90	0.036202	128	0.027645
15	0.128049	53	0.053686	91	0.035900	129	0.027479
16	0.122833	54	0.052958	92	0.035604	130	0.027315
17	0.118090	55	0.052252	93	0.035313	131	0.027153
18	0.113755	56	0.051567	94	0.035027	132	0.026994
19	0.109774	57	0.050902	95	0.034747	133	0.026836
20	0.106105	58	0.050256	96	0.034471	134	0.026681
21	0.102708	59	0.049628	97	0.034201	135	0.026528
22	0.099555	60	0.049018	98	0.033935	136	0.026376
23	0.096617	61	0.048424	99	0.033673	137	0.026226
24	0.093873	62	0.047846	100	0.033416	138	0.026079
25	0.091303	63	0.047284	101	0.033163	139	0.025933
26	0.088889	64	0.046736	102	0.032915	140	0.025789
27	0.086618	65	0.046202	103	0.032671	141	0.025647
28	0.084477	66	0.045682	104	0.032430	142	0.025506
29	0.082454	67	0.045175	105	0.032194	143	0.025367
30	0.080539	68	0.044680	106	0.031961	144	0.025230
31	0.078723	69	0.044198	107	0.031732	145	0.025094
32	0.076998	70	0.043727	108	0.031507	146	0.024960
33	0.075358	71	0.043267	109	0.031285	147	0.024828
34	0.073796	72	0.042818	110	0.031067	148	0.024697
35	0.072306	73	0.042379	111	0.030852	149	0.024568
36	0.070883	74	0.041950	112	0.030641	150	0.024440
37	0.069522	75	0.041531	113	0.030432	151	0.024314
38	0.068220	76	0.041121	114	0.030227	152	0.024189

TABLE A.1.--Continued.

n	w _n	n	w _n	n	w _n	n	w _n
153	0.024065	203	0.019267	253	0.016170	303	0.013991
154	0.023943	204	0.019192	254	0.016120	304	0.013954
155	0.023822	205	0.019118	255	0.016069	305	0.013917
156	0.023703	206	0.019044	256	0.016019	306	0.013881
157	0.023585	207	0.018971	257	0.015969	307	0.013844
158	0.023468	208	0.018899	258	0.015919	308	0.013808
159	0.023352	209	0.018827	259	0.015870	309	0.013772
160	0.023238	210	0.018756	260	0.015821	310	0.013736
161	0.023125	211	0.018686	261	0.015773	311	0.013701
162	0.023013	212	0.018616	262	0.015724	312	0.013665
163	0.022902	213	0.018546	263	0.015677	313	0.013630
164	0.022793	214	0.018478	264	0.015629	314	0.013595
165	0.022685	215	0.018409	265	0.015582	315	0.013560
166	0.022577	216	0.018342	266	0.015535	316	0.013525
167	0.022471	217	0.018275	267	0.015488	317	0.013491
168	0.022366	218	0.018208	268	0.015442	318	0.013457
169	0.022262	219	0.018142	269	0.015396	319	0.013423
170	0.022160	220	0.018076	270	0.015350	320	0.013389
171	0.022058	221	0.018011	271	0.015305	321	0.013355
172	0.021957	222	0.017947	272	0.015259	322	0.013322
173	0.021857	223	0.017883	273	0.015215	323	0.013288
174	0.021758	224	0.017819	274	0.015170	324	0.013255
175	0.021661	225	0.017756	275	0.015126	325	0.013222
176	0.021564	226	0.017694	276	0.015082	326	0.013189
177	0.021468	227	0.017632	277	0.015038	327	0.013157
178	0.021373	228	0.017570	278	0.014995	328	0.013124
179	0.021279	229	0.017509	279	0.014951	329	0.013092
180	0.021186	230	0.017448	280	0.014909	330	0.013060
181	0.021094	231	0.017388	281	0.014866	331	0.013028
182	0.021002	232	0.017328	282	0.014824	332	0.012996
183	0.020912	233	0.017269	283	0.014781	333	0.012965
184	0.020822	234	0.017210	284	0.014740	334	0.012933
185	0.020733	235	0.017152	285	0.014698	335	0.012902
186	0.020646	236	0.017094	286	0.014657	336	0.012871
187	0.020558	237	0.017036	287	0.014616	337	0.012840
188	0.020472	238	0.016979	288	0.014575	338	0.012809
189	0.020387	239	0.016923	289	0.014534	339	0.012779
190	0.020302	240	0.016866	290	0.014494	340	0.012748
191	0.020218	241	0.016810	291	0.014454	341	0.012718
192	0.020135	242	0.016755	292	0.014414	342	0.012688
193	0.020052	243	0.016700	293	0.014375	343	0.012658
194	0.019971	244	0.016645	294	0.014335	344	0.012628
195	0.019890	245	0.016591	295	0.014296	345	0.012599
196	0.019809	246	0.016537	296	0.014257	346	0.012569
197	0.019730	247	0.016484	297	0.014219	347	0.012540
198	0.019651	248	0.016431	298	0.014180	348	0.012511
199	0.019573	249	0.016378	299	0.014142	349	0.012482
200	0.019495	250	0.016325	300	0.014104	350	0.012453
201	0.019419	251	0.016273	301	0.014066		
202	0.019342	252	0.016222	302	0.014029		

APPENDIX B

D(P,d) FOR PERFECT P

According to the statement of Theorem 4.4, there is an extremely small (see Proposition 2.4) set of probability vectors to which it does not apply: the set of perfect probability vectors. If a probability vector is perfect, searching deeper still does not help, for at every search depth the probability of correct decision is exactly 1. This is illustrated for the vector (0, 0, .5, .5) in Table B.1, which contains the same kinds of information as Tables 4.1 through 4.5.

The reverse of a perfect probability vector is not perfect, and so Theorem 4.4 does apply to the vector (.5, .5, 0, 0). This is illustrated in Table B.2, from which it can be seen that the probability of correct decision approaches 1/2 as the search depth increases.

TABLE B.1.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (0, 0, .5, .5)$.

d	T_{d-1}				U_{d-1}				D(P,d)
1	0.00	0.00	0.50	0.50	0.50	0.50	0.00	0.00	1.000
3	0.00	0.00	0.75	0.25	0.25	0.75	0.00	0.00	1.000
5	0.00	0.00	0.94	0.06	0.06	0.94	0.00	0.00	1.000
7	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	1.000
2	0.00	0.00	0.75	0.25	0.50	0.50	0.00	0.00	1.000
4	0.00	0.00	0.94	0.06	0.25	0.75	0.00	0.00	1.000
6	0.00	0.00	1.00	0.00	0.06	0.94	0.00	0.00	1.000
8	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	1.000

TABLE B.2.--Minimax values and probabilities of correct decision for $\Gamma(1,1)$ at various search depths, using the probability vector $P = (.5, .5, 0, 0)$.

d	T_{d-1}				U_{d-1}				$D(P,d)$
1	0.50	0.50	0.00	0.00	0.00	0.00	0.50	0.50	0.000
3	0.00	0.00	0.75	0.25	0.00	0.00	0.63	0.38	0.438
5	0.00	0.00	0.72	0.28	0.00	0.00	0.68	0.32	0.479
7	0.00	0.00	0.74	0.26	0.00	0.00	0.72	0.28	0.493
9	0.00	0.00	0.78	0.22	0.00	0.00	0.77	0.23	0.498
11	0.00	0.00	0.84	0.16	0.00	0.00	0.84	0.16	0.499
13	0.00	0.00	0.91	0.09	0.00	0.00	0.91	0.09	0.500
15	0.00	0.00	0.97	0.03	0.00	0.00	0.97	0.03	0.500
17	0.00	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.500
19	0.00	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.500
2	0.75	0.25	0.00	0.00	0.50	0.50	0.00	0.00	0.375
4	0.61	0.39	0.00	0.00	0.53	0.47	0.00	0.00	0.461
6	0.54	0.46	0.00	0.00	0.51	0.49	0.00	0.00	0.486
8	0.48	0.52	0.00	0.00	0.47	0.53	0.00	0.00	0.495
10	0.40	0.60	0.00	0.00	0.40	0.60	0.00	0.00	0.498
12	0.29	0.71	0.00	0.00	0.29	0.71	0.00	0.00	0.499
14	0.16	0.84	0.00	0.00	0.16	0.84	0.00	0.00	0.500
16	0.05	0.95	0.00	0.00	0.05	0.95	0.00	0.00	0.500
18	0.01	0.99	0.00	0.00	0.01	0.99	0.00	0.00	0.500
20	0.00	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.500

APPENDIX C

A GENERALIZATION OF w_n

The set $\{w_n: n > 1\}$ of Chapter 5 may be extended to a more general set having analogous properties. The theorems in the preceding pages do not depend on this set in any way, but some of its properties are interesting enough to be stated here. The theorems stated below have proofs analogous to the proofs of the theorems in Chapter 5; the author can supply proofs upon request.

In Chapter 5, we noted that for $n > 1$, w_n is the unique intersection point in the interval $(0,1)$ of the the function $y = (1 - (1-x)^n)^n$ and the line $y = x$. Let $m > 1$, $n \geq 0$, and $c \in (0,1]$, and consider the intersection in the interval $(0,1)$ of the function $y = (1 - (1-x)^m)^{m+n}$ and the line of slope c which passes through the point $(1,1)$. Theorem C.1 states that this intersection point is unique.

Theorem C.1. Let $c \in (0,1]$, $m > 1$, and $n \geq 0$, and let

$$g_{c,m,n}(x) = (1 - (1-x)^m)^{m+n} - cx + c - 1$$

for all real x . Then $g_{c,m,n}$ has exactly one root in the interval $(0,1)$.

Notation C.1. Let $c \in (0,1]$, $m > 1$, and $n \geq 0$. $g_{c,m,n}$ shall be as defined in Theorem C.1 above, and the unique $x \in (0,1)$ for which $g_{c,m,n}(x) = 0$ shall be called $w_{c,m,n}$. Note that according to Lemma 5.1, $w_{1,m,0} = w_m$.

Corollary C.1.1. Let $c \in (0,1]$, $m > 1$, and $n \geq 0$. If $0 < x < w_{c,m,n}$, then $g_{c,m,n}(x) < 0$, and if $w_{c,m,n} < x < 1$, then $g_{c,m,n}(x) > 0$.

Corollary C.1.2. Let $m > 1$ and $n \geq 0$. If $0 < x < w_{1,m,n}$, then $(1 - (1-x)^m)^{m+n} < x$, and if $w_{1,m,n} < x < 1$, then $(1 - (1-x)^m)^{m+n} > x$.

Corollary C.1.3. Let $b \in (0,1]$, $c \in (0,1]$, $m > 1$, and $n \geq 0$. If $b < c$, then $w_{b,m,n} < w_{c,m,n}$.

Theorem C.2. Let $x_0 \in [0,1]$, $m > 1$, and $n \geq 0$. For every integer $i \geq 0$, let $x_{i+1} = (1 - (1-x_i)^m)^{m+n}$. Then--

1. if $0 \leq x_0 < w_{1,m,n}$, then $\lim_{i \rightarrow \infty} x_i = 0$;
2. if $w_{1,m,n} < x_0 \leq 1$, then $\lim_{i \rightarrow \infty} x_i = 1$;
3. if $x_0 = w_{1,m,n}$, then $x_i = w_{1,m,n}$ for all i .

Theorem C.3. Let $c \in (0,1]$ and $m > 1$. Then $\lim_{n \rightarrow \infty} w_{c,m,n} = 1$.

Theorem C.4. Let $c \in (0,1]$ and $n \geq 0$. Then $\lim_{m \rightarrow \infty} w_{c,m,n} = 0$.

APPENDIX D

MORE THEOREMS ABOUT DOUBLE SEQUENCES

Many more theorems about the double sequences of Notation 6.2 can be proved than are needed for this dissertation. Some of them are stated here. As with the material in Appendix C, these theorems are only of peripheral interest, so their proofs are not included. The author can furnish proofs on request.

Theorem D.1. Let $m \geq 1$ and $y_0 = 0$. Then there is an N such that for every $n > N$, $\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 0$.

Theorem D.2. Let $n \geq 1$, $x_0 > 0$, and $y_0 = 0$. Then there is an M such that for every $m > M$,

$$\lim_{i \rightarrow \infty} x_{m,n,i} = 1 \quad \text{and} \quad \lim_{i \rightarrow \infty} y_{m,n,i} = 0.$$

Theorem D.3. Let $m \geq 1$ and $n \geq 1$. If $\lim_{i \rightarrow \infty} y_{m,n,i} = 1$, then $\lim_{i \rightarrow \infty} x_{m,n,i} = 1$.

Theorem D.4. Let $y_0 > 0$ and $m \geq 1$. Then there is an N such that for every $n > N$, $\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 1$.

Theorem D.5. Let $m \geq 1$ and $n \geq 1$. Then--

1. if $0 < x_0 < y_0 < 1$, then for every integer $i \geq 0$,
 $0 < x_{m,n,i} < y_{m,n,i} < 1$;
2. if $0 < y_0 < x_0 < 1$, then for every integer $i \geq 0$,
 $0 < y_{m,n,i} < x_{m,n,i} < 1$;
3. if $0 < y_0 = x_0 < 1$, then for every integer $i \geq 0$,
 $0 < y_{m,n,i} = x_{m,n,i} < 1$.

Theorem D.6. Let $m \geq 1$ and $n \geq 1$. Suppose there is an integer k such that

$$x_{m,n,k+1} > x_{m,n,k} \quad \text{and} \quad y_{m,n,k+1} > y_{m,n,k}.$$

Then for every integer $i \geq k$,

$$x_{m,n,i+1} > x_{m,n,i} \quad \text{and} \quad y_{m,n,i+1} > y_{m,n,i}.$$

Theorem D.7. Let $n > 1$ and $x_0 = 0$. If $y_0 < w_n$, then there is an M such that for every $m > M$,

$$\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 0.$$

If $y_0 > w_n$, then there is an M such that for every $m > M$,

$$\lim_{i \rightarrow \infty} x_{m,n,i} = \lim_{i \rightarrow \infty} y_{m,n,i} = 1.$$

APPENDIX E

PATHOLOGY FOR UNNATURAL PROBABILITY VECTORS

Because of Proposition 6.1, the properties of unnatural probability vectors are of strictly secondary interest. Several theorems concerning pathology for unnatural probability vectors are stated here without proof; the author can supply proofs upon request.

If a probability vector is unnatural, then either it has more zero entries at its left end than its right end, or vice versa. These two types of unnatural probability vectors we call right-skewed and left-skewed, respectively. For example, $(0, 0, 0.2, 0.5, 0.3, 0)$ is right-skewed, and $(0, 0.2, 0.5, 0.3, 0, 0)$ is left-skewed. Intuitively, a right-skewed probability vector is a "very good" one, and a left-skewed probability vector is a "very bad" one.

Theorem E.1. Let P be a right-skewed, imperfect \hat{f} -vector. Then for every integer $m \geq 1$, there is an N such that for every integer $n > N$, $\bar{\Gamma}(m, n)$ is P -pathological.

Theorem E.2. Let P be a right-skewed \hat{f} -vector. Then for every integer $n \geq 1$, there is an M such that for every integer $m > M$, $\lim_{d \rightarrow \infty} D_{m, n}^{m, n}(P, d) = 1$.

Taken together, Theorems E.1 and E.2 state that if P is right-skewed, then $\bar{\Gamma}(m, n)$ can alternately be made nonpathological or pathological simply by increasing m or n , respectively.

Theorem E.3. Let P be a left-skewed \hat{f} -vector. Then there is an N such that for all integers $m \geq 1$ and $n > N$, $\Gamma(m,n)$ is P -pathological.

Theorem E.4. For almost every left-skewed \hat{f} -vector P , for every integer $n > 1$ there is an M such that for every integer $m > M$, $\Gamma(m,n)$ is P -pathological.

Conjecture E.1. For every left-skewed \hat{f} -vector P , for every integer $n \geq 1$ there is an M such that for every integer $m > M$, $\Gamma(m,n)$ is P -pathological.

Taken together, Theorem E.3 and Conjecture E.1 state that for every left-skewed \hat{f} -vector P , only a finite number of the $\Gamma(m,n)$ are nonpathological. If Conjecture E.1 is true, this allows Corollary 6.4.2 to be extended slightly.

APPENDIX F

ALTERNATING PATHOLOGICAL AND
NONPATHOLOGICAL BEHAVIOR

As can be seen from Tables 7.3 through 7.7, increasing m while holding n fixed can cause the limiting values of the T and U vectors to change several times. One might suspect that for an appropriate P , this would cause $\bar{\Gamma}(m,n)$ to be alternately pathological and nonpathological. We shall not discuss this in detail, but there are reasons to believe that some likely candidates for such a P are those P such that

$$\begin{aligned} (P)_0 &= (1+\epsilon) w_k & \text{and} & & (P)_r &= (1+\epsilon) w_{k-1}, \\ (P)_{0,1} &= (1+\epsilon) w_{k-2} & \text{and} & & (P)_{r-1,r} &= (1+\epsilon) w_{k-3}, \\ (P)_{0,(r-2)/2} &= (1+\epsilon) w_{k-r+2} & \text{and} & & (P)_{(r+2)/2,r} &= (1+\epsilon) w_{k-r+1}, \end{aligned}$$

where r is even, $k \geq r+3$ is an integer, and $\epsilon > 0$ is small.

The author has written a computer program to generate probability vectors having the properties specified in the above equations, and several of them have been tested using the computer program of Appendix G. Alternating pathological and nonpathological behavior does indeed occur for several of these vectors. This is illustrated in Tables F.1 and F.2 for the vector $P = (.1108721768, .0184577789, .0278285401, .6653850732, .0358106695, .0223749103, .1192708512)$. These tables give the same kind of information as Tables 7.3 through 7.12.

TABLE F.1.--Pathology of $\bar{\Gamma}(m,n)$ as a function of m and n, for the probability vector $P = (.1108721768, .0184577789, .0278285401, .6653850732, .0358106695, .0223749103, .1192708512)$.

n =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
m																				
1	3*	3*	3*	3*	3*	3*	3*	3*	3*	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*
2	3*	3*	3*	3*	3*	3*	3*	3*	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*
3	3*	3*	3*	3*	3*	3*	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*
4	3*	3*	3*	3*	3*	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*
5	3*	3*	3*	3*	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*
6	3*	3*	3*	3*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*
7	3*	3*	23	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*
8	3*	23	2*	2*	2*	2*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
9	23	2*	2*	2*	1*	1*	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
10	2*	2*	2*	12	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
11	2*	2*	12	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
12	2*	12	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
13	12	1*	1*	1*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
14	1*	1*	1*	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
15	1*	1*	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
16	01	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
17	01	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
18	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
19	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
20	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
21	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*
22	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

NOTE: An asterisk indicates pathology. Its absence indicates that $D(P_5, d)$ converges to 1. The additional information contained in this table is explained in Chapter 7.

APPENDIX G

A COMPUTER PROGRAM

The computer program listed below is the final and most general version of a series of programs which played an important role in the development of this dissertation. Many of the mathematical intuitions leading to the proofs of the theorems in the dissertation were gained by using this program or its predecessors to compute minimax values and probabilities of correct decision at various search depths for many different probability vectors. This program was also used to generate the data appearing in the tables in Chapter 4, Chapter 7, and Appendix F.

The program is written in C, which is the primary programming language for the computer system used by the author. The variable names used in the program are somewhat different than the reader might expect: when the program was written, the author was using the names A_i , B_i , C_i , and D_i for S_d , T_d , U_d , and V_d , respectively.

```
#
/*
** Syntax: level [<options>] <r>,<s> <B0> [; <C0>]
**
** <r> and <s> are the positive and negative branching factors,
** and B0 and C0 are the starting vectors for B and C nodes.
** The program prints Bi and Ci for i = 0 by 2 until
** convergence is achieved.
** If the vector C0 is omitted, the reverse of B0 is assumed.
**
** Options:
```

```

** -o "odd": prints Bi and Ci for i = 1 by 2 instead.
** -b<#> "by <#>": causes only every #'th line to be printed
** (except that the last line is always printed).
** -t<#> "to <#>": causes execution until exactly <#> lines
** have been printed (instead of until convergence)
** -c "count" prints i each time
** -e<#> "epsilon" sets epsilon (convergence test) to <#>.
** -<#> prints <#> digits after the decimal point
** (default is 6).
** -d "decision" prints out the prob. of correct decision.
*/

```

```

#define ELTS      20          /* max no. of elts in a pdf vector */
#define FUZZ      .0000005   /* to test for convergence */

```

```

char      *errname "level";
int       fout;
int       elts;
int       plus;
int       minus;
int       stop;
char      *cntl1;
char      *cntl2;
int       byflag 1;
int       putflag 1;
int       ctflag;
int       oddflag;
int       dflag;
int       toflag;
double    epsilon FUZZ;

```

```

main(argc,argv)

```

```

char **argv;
{
    double    aarev[ELTS]; /* reverse of A pdf vector */
    double    bb[ELTS];   /* B pdf vector */
    double    cc[ELTS];   /* C pdf vector */
    double    ddrev[ELTS]; /* reverse of D pdf vector */
    double    atof();     /* ascii to float converter */
    register ii;
    register char **argv;
    char *p;
    int count;

    /* ignore prog name */
    argv = argv+1;
    --argc;

    cntl1 = cat (" %f", 0);
    cntl2 = cat (" %f", 0);
    /* get dash args */
    while (argc > 0 && **argv == '-') {
        switch (argv[0][1]) {

            case 'b':

```

```
        byflag = atoi(&argv[0][2]);
        putflag = byflag;
        break;

    case 'c':
        ctflag = 1;
        break;

    case 'd':
        dflag = 1;
        free (cntl2);
        cntl2 = cat (" %.", &argv[0][2], "f", 0);
        break;

    case 'e':
        epsilon = atof (&argv[0][2]);
        break;

    case 't':
        toflag = atoi(&(argv[0][2]));
        break;

    case 'o':
        oddflag = 1;
        break;

    default:
        free (cntl1);
        cntl1 = cat (" %.", &argv[0][1], "f", 0);
    }
    --argc;
    argv++;
}

/* must have at least 3 more args */
if(argc<3) error("arg count");

/* find out how many B and C children an A node has */
plus = atoi(*argv);
for(p= *argv++;) {
    if(!*p) error("expecting comma");
    if(*p++ == ',') break;
}
minus = atoi(p);

for(ii=0;;ii++) {
    if(*argv == -1 || **argv=='\n') break;
    bb[ii] = atof(*argv++);
}
elts = --ii;
if (*argv == -1) {
    for(ii=0;ii<=elts;ii++) cc[ii] = bb[elts-ii];
}
else {
    argv++;
}
```

```

    for(ii=0;;ii++) {
        if(*argv == -1 || **argv==';') break;
        cc[ii] = atof(*argv++);
    }
    if(--ii!=elts) error("B0 size != C0 size");
}

if (oddflag) {
/*
* In this case, we want to print B[2i+1] and C[2i+1]
* so we first compute B[1] and C[1] in terms of A[0] and D[0]
*/
    norm (bb,cc);
    for (ii=0; ii<=elts; ii++) {
        aarev[ii] = bb[elts-ii];
        ddrev[ii] = cc[elts-ii];
    }
    revmax(aarev,aarev,bb);
    revmax(ddrev,aarev,cc);
}

for(count=1;;count++) {

    /* to try to compensate for roundoff error: */
    stop = norm(bb, cc);
    if (toflag) stop=0;

    if (count==putflag || stop) {
        if (toflag) {
            --toflag;
            if (!toflag) stop = 1;
        }
        putflag += byflag;
        if (ctflag) printf ("e[%d]: ", 2*(count-1)+oddflag);
        if (dflag) printf ("D[%d]: ", 2*count-1+oddflag);
        for(ii=0;;ii++) {
            printf(cntll+1,bb[ii]);
            if(ii>=elts) break;
            putchar(' ');
        }
        putchar(';');
        for(ii=0;ii<=elts;ii++) printf(cntll,cc[ii]);
        if (dflag) {
            printf(" =>");
            decision(bb,cc);
        }
        putchar('0');
        if(stop) break;
    }

    revmax(bb,cc,aarev);
    revmax(cc,cc,ddrev);
    revmax(aarev,aarev,bb);
    revmax(ddrev,aarev,cc);
}

```

```

}

norm(bb, cc)
double bb[], cc[];
{
    double    sum1,sum2;
    register stop, ii;

    sum1 = sum2 = 0;
    for(ii=0;ii<=elts;ii++) {
        sum1 += bb[ii];
        sum2 += cc[ii];
    }
    stop = 1;
    for(ii=0;ii<=elts;ii++) {
        bb[ii] /= sum1;
        cc[ii] /= sum2;
        stop =& (bb[ii] < epsilon || bb[ii] > 1-epsilon);
        stop =& (cc[ii] < epsilon || cc[ii] > 1-epsilon);
    }
    return stop;
}

revmax(child0,child1,result)
double child0[],child1[],result[];
{
    static double sum0,sum1,oldprod,prod;
    register index;
    double pow();

/*
 * for all i,
 * result[n-i] (since result is reversed) is
 *     "plus"th power of sum from k=0 to i of
 *         k'th elt of first child vector
 *     times "minus"th power of sum from k=0 to i of
 *         k'the elt of second child vector
 *     minus sum from j=0 to i-1 of result[n-j]
 */

    sum0 = sum1 = 0;
    oldprod = 0;
    for(index=0;index<=elts;index++) {
        sum0 += child0[index];
        sum1 += child1[index];
        prod = pow(sum0,plus) * pow(sum1,minus);
        result[elts-index] = prod-oldprod;
        oldprod = prod;
    }
}

/*
 * fast exponentiation routine
 * to get around slowness of floating point simulation routines.
 * calling syntax:  pow (<base>, <exponent>)

```

```

* where <base> is double and <exponent> is integer.
*/
double pow(base,ex)
double base;
{
    if (ex==0) return 1;
    if ((ex/2) * 2 == ex) return pow (base * base, ex/2);
    return base * pow (base, ex-1);
}

decision(bb, cc)
double bb[], cc[];
{
    double choose();
    double pow();
    double sum, result;
    double x,y;
    register ii, jj, kk;
    double cpi, cmj;

    result = 0;
    for(ii=1;ii<=plus;ii++) {
        cpi = choose (plus, ii);
        for(jj=0;jj<=minus;jj++) {
            cmj = choose (minus, jj);
            sum = x = y = 0;
            for(kk=0;kk<=elts;kk++) {
                sum += (
                    pow(bb[kk],ii) *
                    pow(x,plus-ii) *
                    pow(cc[kk],jj) *
                    pow(y,minus-jj)
                );
                x += bb[kk];
                y += cc[kk];
            }
            sum *= ( (cpi * cmj * ii) / (ii+jj+0.0) );
            result += sum;
        }
    }
    printf(cntl2, result);
}

/* n things taken k at a time */
/* = n * (n-1) * ... * (n-k+1) / k! */
double choose (n, k)
{
    double p1; /* numerator */
    double p2; /* denominator */
    int i; /* loop ctr */
    double fmod();

    p1 = 1;
    p2 = 1;

```



```
/*  
 * C(n,k) = C(n,n-k),  
 * so choose the one that involves the lesser number of  
 * multiplications  
 */  
    if (n-k < k) k = n-k;  
  
    for (i=n-k+1; i<=n; i++) p1 *= i;  
    for (i=1; i<=k; i++) p2 *= i;  
    p1 = p1/p2 + 0.5;  
    p1 = p1 - fmod (p1, 1.0);  
    return (p1);  
}
```

APPENDIX H

INVESTIGATING PATHOLOGY IN PRACTICAL SITUATIONS

It is doubtful that pathology occurs very often in common applications of game trees, but to what extent it does occur remains to be seen. Even where pathology occurs, it will probably not be easy to observe, as there are several unsolved methodological problems to be dealt with.

For example, consider the question of whether pathology occurs in any commonly played parlor game. One problem is how to choose an appropriate game. Regardless of what game is chosen, a random number generator will have to be incorporated into the game playing program if it is to choose at random among moves having the same minimax values.

There may be ways to increase the chances of pathology occurring in a game. As pointed out in Chapter 7, pathology should be more likely if the range of the evaluation function is reduced by a procedure such as rounding. Also, pathology might be more likely if a random number generator is used to introduce some additional error into the evaluation function.

If pathology occurs, it may not be easy to detect. One might try playing a game program against itself using various search depths to see which side wins, but since the side with the deeper search will always have an advantage near the end of the game, a win for that side would not necessarily indicate absence of pathology.

Another approach might be to print out tables of minimax values, to see whether they become more alike as the search depth increases. There are some problems with this approach as well. Most evaluation functions compute their returned values by examining a number of features of the playing board, and some of these features may remain unchanged for several moves. Thus, there may be a high correlation between the minimax values at one search depth and the values at other search depths close by, regardless of whether these values are accurate or inaccurate. Since we do not know for most games which positions are forced wins and which ones are forced losses, interpretation of the tables may be very difficult.

The author has attempted some investigation of pathology using a computer program which plays the game of othello [HA2], but little has been accomplished so far. Othello was chosen mainly because the computer program was already available on the computer system being used,¹ and it seemed easier to modify it than to write a program for some other game. This may have been a bad idea. The program turned out to contain undiscovered bugs, and several more were introduced by the author's modifications. Their cumulative effect was to almost totally invalidate the results of several months of work.

One of the major problems was lack of randomness in the random number generator. This caused the percentages of wins for each side to vary wildly during a series of several hundred games which the program played against itself. Finding a suitable

¹This was an early version of the prize-winning othello-playing program written by Dennis Rockwell and Thomas Truscott.

random number generator may require considerable work.

LIST OF REFERENCES

- AH1 Aho, A. V.; Hopcroft, J. E.; and Ullman, J. D. The Design and Analysis of Computer Algorithms. Reading, Mass.: Addison-Wesley Publishing Company, 1975.
- BA1 Banerji, R. B., and Ernst, G. W. "Strategy Construction Using Homomorphisms Between Games." Artificial Intelligence 3 (Winter 1972), 223-249.
- BA2 Barlow, R. E.; Fussell, J. B.; and Singpurwalla, N. D. Reliability and Fault Tree Analysis: Theoretical and Applied Aspects of System Reliability and Safety Assessment. Philadelphia: Society for Industrial and Applied Mathematics, 1975.
- BA3 Baudet, G. M. "On the Branching Factor of the Alpha-Beta Pruning Algorithm." Artificial Intelligence 10 (April 1978), 173-199.
- BI1 Biermann, A. W. "Approaches to Automatic Programming." In Advances in Computers, Vol. 15, M. Rubinoff and M. Yovits, eds. New York: Academic Press, Inc., 1976, pp. 1-63.
- BI2 Biermann, A. W. "Theoretical Issues Related to Computer Game Playing Programs." Personal Computing, September 1978, pp. 86-88.
- CH1 Chandra, A. K., and Stockmeyer, L. J. "Alternation." Seventeenth Annual IEEE Symposium on Foundations of Computer Science (October 1976), 98-108.
- DA1 Davis, R.; Buchanan, B.; and Shortliffe, E. "Production Rules as a Representation for a Knowledge-Based Consultation Program." Artificial Intelligence 8 (February 1977), 15-45.
- DE1 De Millo, R. A.; Lipton, R. J.; and Perlis, A. J. "Social Processes and Proofs of Theorems and Programs." Communications of the ACM 22 (May 1979), 271-280.
- DO1 Domoryad, A. P. Mathematical Games and Pastimes. New York: The MacMillan Company, 1964, pp. 61-62.
- FU1 Fuller, S. H.; Gaschnig, J. G.; and Gillogly, J. J. "Analysis of the Alpha-Beta Pruning Algorithm." Department of Computer Science, Carnegie-Mellon University, July 1973.
- GR1 Greenblatt, R. D.; Eastlake, D. E. III; and Crocker, S. D. "The Greenblatt Chess Program." Proceedings of the AFIPS

Fall Joint Computer Conference 31 (1967), 801-810.

- HA1 Harris, L. R. "The Heuristic Search under Conditions of Error." Artificial Intelligence 5 (Fall 1974), 217-234.
- HA2 Hasegawa, G., and Brady, M. How to Win at Othello. New York: Jove Publications, Inc., 1977.
- JA1 Jackson, P. C. Introduction to Artificial Intelligence. New York: Petrocelli Books, 1974.
- JA2 James, G., and James, R. C., eds. Mathematics Dictionary. Princeton, N. J.: D. Van Nostrand Company, Inc., 1964.
- KA1 Karp, R. M. "The Probabilistic Analysis of Some Combinatorial Search Algorithms." In Algorithms and Complexity: New Directions and Recent Results, J. F. Traub, ed. New York: Academic Press, Inc., 1976, pp. 1-19.
- KN1 Knuth, D. E., and Moore, R. W. "An Analysis of Alpha-Beta Pruning." Artificial Intelligence 6 (Winter 1975), 293-326.
- LA1 LaValle, I. H. Fundamentals of Decision Analysis. New York: Holt, Rinehart and Winston, 1978.
- LE1 Levi, G., and Sirovich, F. "Generalized And/Or Graphs." Artificial Intelligence 7 (Fall 1976), 243-259.
- LI1 Lindstrom, G. "Alpha-Beta Pruning on Evolving Game Trees." Technical Report UUCS 79-101, Department of Computer Science, University of Utah, March 1979.
- LO1 Loveland, D. W. Automated Theorem Proving: A Logical Basis. Amsterdam: North-Holland Publishing Company, 1978, pp. 2, 335-391.
- NA1 Nau, D. S. "Preliminary Results Regarding Quality of Play Versus Depth of Search in Game Playing." First International Symposium on Policy Analysis and Information Systems (June 1979), proceedings to appear.
- NI1 Nilsson, N. J. Problem-Solving Methods in Artificial Intelligence. New York: McGraw-Hill Book Company, 1971.
- OW1 Owen, G. Game Theory. Philadelphia: W. B. Saunders Company, 1968.
- RA1 Rabin, M. O. "Probabilistic Algorithms." In Algorithms and Complexity: New Directions and Recent Results, J. F. Traub, ed. New York: Academic Press, Inc., 1976, pp. 21-39.
- RO1 Robinson, A. L. "Tournament Competition Fuels Computer Chess." Science 204 (29 June 1979), 1396-1398.

- SA1 Samuel, A. L. "Some Studies in Machine Learning Using the Game of Checkers. II--Recent Progress." IBM Journal of Research and Development 11 (November 1967), 601-617.
- SH1 Shortliffe, E. H., and Buchanan, B. G. "A Model of Inexact Reasoning in Medicine." Mathematical Biosciences 23 (April 1975), 351-379.
- SI1 Singleton, R. R., and Tyndall, W. F. Games and Programs: Mathematics for Modeling. San Francisco: W. H. Freeman and Company, 1974.
- SL1 Slagle, J. R. Artificial Intelligence: The Heuristic Programming Approach. New York: McGraw-Hill Book Company, 1971.
- ST1 Stark, R. M., and Nicholls, R. L. Mathematical Foundations for Design: Civil Engineering Systems. New York: McGraw-Hill Book Company, 1972.
- ST2 Stockman, G. C. "A Problem-Reduction Approach to the Linguistic Analysis of Waveforms." Ph.D. dissertation, Technical Report TR-538, University of Maryland, May 1977.
- TO1 Tou, J. T., and Gonzalez, R. C. Pattern Recognition Principles. Reading, Mass.: Addison-Wesley Publishing Company, 1974.
- TR1 Truscott, T. R. Personal communication (1979).
- TR2 Truscott, T. R. "Minimum Variance Tree Searching." First International Symposium on Policy Analysis and Information Systems (June 1979), proceedings to appear.
- TU1 Tummala, V. M. R. Decision Analysis with Business Applications. New York: Intext Educational Publishers, 1973.
- VA1 VanderBrug, G. J., and Minker, J. "State-Space, Problem-Reduction, and Theorem Proving--Some Relationships." Communications of the ACM 18 (February 1975), 107-115.

A BRIEF BIOGRAPHY OF THE AUTHOR

Dana S. Nau was born in Urbana, Illinois on December 29, 1951. He graduated summa cum laude from the University of Missouri at Rolla in May 1974 with a B.S. in Applied Mathematics, and received an A.M. in Computer Science from Duke University in September 1976.

In reverse chronological order, papers by Mr. Nau are as follows:

1. Dana S. Nau. "Preliminary Results Regarding Quality of Play Versus Depth of Search in Game Playing." First International Symposium on Policy Analysis and Information Systems (June 1979), proceedings to appear.
2. Dana S. Nau, George Markowsky, Max A. Woodbury, and D. Bernard Amos. "A Mathematical Analysis of HLA Serology." Mathematical Biosciences 40 (1978), 243-270. Also available as Research Report RC6803, IBM T. J. Watson Research Center, 1977.
3. Dana S. Nau and Max A. Woodbury. "A Command Processor for the Determination of Specificities from Matrices of Reactions between Blood Cells and Antisera." Computers and Biomedical Research 10 (1977), 259-269.
4. Alan Biermann and Dana S. Nau. "Finite-State Solutions to the Tower of Hanoi Problem." Technical Report CS-1976-18, Computer Science Department, Duke University, 1976.
5. Dana S. Nau. "The Specificity Problem: Immunological Applications, Computational Complexity, and a Computer Program." A.M. thesis, Duke University, 1976. Available as Technical Report CS-1976-7, Computer Science Department, Duke University, 1976.
6. Dana S. Nau. "A Note on the Correctness of Kugel's Theorem." Sigart Newsletter (October 1975).
7. Dana S. Nau. "A Theorem Proving Computer Program." First place award at the 1974 Kappa Mu Epsilon Region 5 convention.
8. Dana S. Nau. "A Lattice Point Problem." First place award at the 1972 Kappa Mu Epsilon Region 5 convention. Subsequently published in The Pentagon (Fall 1972), 25-28.

9. Dana S. Nau. "An Original Proof That the Integral from 1 to x of dt/t Is the Inverse of e^x ." Third place award at the Twenty-First International Science Fair, 1970.

In chronological order, the appointments and positions held by Mr. Nau since receiving the bachelor's degree include--

1. a summer research position with Max Woodbury at Duke University;
2. a summer research position with Alan Biermann at Duke University;
3. research positions for two summers at the I.B.M. Thomas J. Watson Research Center in Yorktown Heights, N.Y.;
4. independent consulting with the I.B.M. Thomas J. Watson Research Center in Yorktown Heights, N.Y.;
5. a part-time instructorship in computer science for two semesters at Duke University;
6. an assistant professorship in computer science at the University of Maryland, to begin in August of 1979.

In addition to numerous awards and honors received before getting his Bachelor's degree, Mr. Nau has received both National Science Foundation and James B. Duke graduate fellowships, and is a member of Sigma Xi research honorary. Extra-curricular activities in which he has been involved while in graduate school include several musical activities on the clarinet, sound design for dramatic productions, and bicycling.

