

UNIVERSITY OF CALIFORNIA

Los Angeles

Recursive Random Games:

A Probabilistic Model for Perfect Information Games

A dissertation submitted in partial satisfaction of the  
requirements for the degree Doctor of Philosophy  
in Computer Science

by

Gerard Philippe Michon

1983

**MASTER COPY**

**840029**

900 511111

80008

© Copyright by  
Gerard Philippe Michon


1983

The dissertation of Gerard Philippe Michon is approved

  
Jack W. Carlyle

  
Thomas S. Ferguson

  
Sheila A. Greibach

  
E. Burton Swanson

  
Judea Pearl, Committee Chair

University of California, Los Angeles

1983

## Table of contents

	Page
Table of contents	iii
Index to definitions, theorems and figures	iv
Abstract	vii
0.Preface and Summary	1
1.Introduction	11
2.Impartial recursive random games	15
3.Matched statistics and inert structures	29
4.Analyzing the complexity of game solving	41
5.Partisan games	57
6.Error propagation and pathology analysis	67
7.Quiescence analysis	77
8.Estimating the winning probability after a truncated search	94
Appendix	
A.Strange non-inert internal structures	101
B.Necessary and sufficient conditions for inertness	105
C.Standard pruning transforms N and M	107
References	113

INDEX to Definitions, Theorems and Figures.

Branching Factor $r$	.....	Def.	2.4
- of SOLVE	.....	Th.	4.1
- of SOLVE <sub><math>i</math></sub>	.....	Th.	4.3
Bushy Inert Structures	.....	Th 4.2 Fig.	3.2
External Structure	.....	Def.	3.2
Games of Finite Height	.....	Def.	2.3
Game Structure (G)	.....	Def.	2.6
HSOLVE	.....	Fig.	7.2
Inert Games	.....	Def.	3.4
Internal Structure	.....	Def.	3.1
Length of a Game	.....	Def.	2.5
Matched statistics	.....	Th 3.1 Def.	3.3
Minimal Search Tree	.....	Def.	7.1
Misère Play Rule	.....	Fig.	1.1
Normal Play Rule	.....	Fig.	1.1
Partisan Games	.....	Def.	5.1
Partisan Statistics	.....	Def.	5.2
Performance of SOLVE	.....	Fig.	4.1
Performance of SOLVE <sub><math>i</math></sub>	.....	Fig.	4.2
Probability of finiteness	.....	Th 2.1 Fig.	2.2
Probability of a game	.....	Fig.	2.1
Probability of losses	.....	Def.	2.3
Probability of ties	.....	Fig.	2.3
Probability of wins	.....	Def.	2.3

QSOLVE	.....	Fig.	7.2
Quiescence	.....	Def.	7.2
Quiescence Consistency	.....	Th.	7.1
Quiescence Threshold	.....	Fig.	7.1
Recursive Random Games $G$	.....	Def.	2.1
Stability	.....	Fig.	5.2
Static Evaluation	.....	Fig.	6.1
Statistical Structure $F$	.....	Def.	2.2
Status of Games	.....	Fig.	1.1
Status Diagram	.....	Fig.	3.1
Survival Rate (b)	.....	Def.	2.2
Transition Statistics	.....	Fig.	2.4
Visible Nodes	.....	Fig.	7.2
Winning probability $p$	.....	Def.	2.2

## VITA

March 29, 1956--Born, Talence (Gironde), France

1976-1979-- Ecole Polytechnique, Paris, France

1979-1980-- Ecole Nationale Superieure des Telecommunications

1980-1981-- M.S., University of California, Los Angeles

1981-1983-- Research Assistant, Department of Computer Science

University of California, Los Angeles



## ABSTRACT OF THE DISSERTATION

Recursive Random Games:

A Probabilistic Model for Perfect Information Games

by

Gerard Philippe Michon

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 1983

Professor Judea Pearl, Chair

A simple probabilistic model for game trees is described which exhibits features likely to be found in realistic games. The model allows any node to have  $n$  offsprings (including  $n=0$ ) with probability  $f_n$  and assigns each terminal node a WIN status with probability  $p$  and a LOSS status with probability  $q = 1-p$ . Our model may include infinite game trees and/or games that never end when played perfectly. The statistical properties of games and the computational complexities of various game solving approaches are quantified and compared. A simple analysis of game pathology and quiescence is also given. The model provides a theoretical justification for the observed good behavior of game-playing programs whose search horizon is not rigid. Pathological features that were recently found to be inherent in some former game models are put in a new perspective.

Recursive Random Games:  
A Probabilistic Model for Perfect Information  
Games

0. Preface and Summary

Traditionally, game-playing has been regarded as one of the key activities a machine would have to perform well before the question of its intelligence could be addressed, but all is not well with this idea. The power of fascination of a working program is such that people often see this as an end in itself, yet since computer science is still largely an experimental field, we have the possibility of trying out ideas before questioning their scope or validity. The result is that if some ideas work reasonably well, they are unlikely ever to be questioned. For example, the reasons why a simple idea like minimaxing works are still quite obscure once one recognizes that an initial intuitive appeal and/or a decent performance in actual tests do not qualify as rational explanations. Moreover, before we could even wonder why any given methodology works, we should wonder what it means that such a methodology does work. Do we want a procedure to perform well against opponents whose abilities are inherently limited, or do we request some kind of

absolute measure ?

Game-playing is a perfect laboratory for studying a pure form of deductive reasoning uncorrupted by previous knowledge. We are not interested here in the form of expertise that is gained through long and tedious compilation of recipes: our object is the process by which those recipes are synthesized efficiently from scratch. What are the mechanisms that contribute to turn a rough and inaccurate understanding of a situation into a decent level of expertise without resorting to externally compiled knowledge ? A better understanding of such mechanisms would undoubtedly be gained if their effects could be measured with respect to some universal yardstick.

The staggering combinatorial complexity of usual games makes a probabilistic yardstick an obvious choice. This is similar to the problem of statistical mechanics, where the number of molecules in a typical real sample is so large that deterministic laws can only be used as tools for deriving the probabilistic laws that govern the behavior of the sample as a whole. Compare this to the size of the game tree for chess, traditionally estimated to be around  $10^{120}$ , and the number of different board positions roughly  $10^{40}$ . Moreover, one can even design

games that are theoretically undecidable. No algorithm could be designed that would be guaranteed to find whether a typical position of such a game can be settled in finitely many moves when played perfectly.

[\*]

In spite of this staggering complexity, simple practical ideas like the minimax evaluation of a truncated game tree have proved reasonably effective. However, if the playing performance of commercially available microprocessor-based machines is no longer ridiculous, the play-level of even the best programs running on the most powerful machines available still does not compare with that of the better human experts. We conjecture that this is due, in part, to the lack of a reliable measure that could accurately monitor progress in game-playing technology. We still lack an efficient way of discriminating between good and bad new ideas, and are often too shy in questioning the validity of old ones.

Normally, we hope that a few general concepts combined with a lot of wasted computations will enable a machine to reach a decent level of expertise. This works to some extent. When this does not measure up to our expectations, the usual remedy is to introduce still more computing power and hope that the result will somehow be better. The basic processes through which common sense principles are turned into expertise are hardly ever questioned. Undoubtedly, current approaches are

---

\* Petri Nets can be viewed as games where the transition which is fired next is up the player whose turn it is. The outcome of such games is determined according to the normal play rule: whoever cannot fire a transition loses the game. 'Petri Net Games' are undecidable in the sense described here.

able to perform this magic but there is also very little doubt that they are suboptimal. It is unlikely that we just need a few more tricks and a little more speed to let our machines compete against the best experts and win. The success that brute force approach has enjoyed needs explaining. A probabilistic quantification of game concepts should help in such a clarification and contribute to a better design of future procedures. To achieve this goal, our probabilistic yardstick needs to be both simple to use and universal enough.

Our exploration of these problems in the present study is modest, restricted to an effort to explain how some procedures effectively take advantage of some probabilistic features of certain games to improve on the quality of any decision procedure, including random guessing, at the expense of more computational time. For this we need a meta-domain of discourse that would allow us to examine infinitely many game positions at once. For if there is a form of intelligence that works surprisingly well without any previous experience in the domain of discourse, the very mechanisms that make such a form of intelligence work at all cannot be captured by the study of any single game. Such a discourse is given to us by the theory of probability.

The use of probabilistic methods in the study of a totally deterministic domain like perfect information games may appear superficially surprising, but in the world of games like in many others, determining the likelihood of a given feature is often much easier than recognizing its occurrence in a particular instance. For example, it may well be easier to find the probability of winning from a typical position than

it is to find the correct winning strategy from a given situation. In other words, it may be much easier to know how well we are expected to do than how well we are doing. The best we can do, then, is choose a procedure of good expected performance according to some probabilistic model of the situation.

This paper presents such a probabilistic model for two-player perfect information games. Every game position allows a random number of legal moves such that each, if enacted, leads to a similar situation. The model both conveys nontrivial general properties of games and accounts for some aspects of games that were overlooked in former models discussed in the literature. This includes game models based on a uniform tree structure [15,17,23,26], that is, games that always allow the same number of legal options irrespective of the particular situation. By placing those models in a larger perspective, we explain some of the paradoxical artifacts with which they were recently found to be plagued.

Section 1 introduces the fundamental concepts generally used in the world of games in a perspective suitable for the rest of this exposition. We emphasize that games can be reduced to game trees, that games whose graphs contain cycles will unfold into infinite game trees, and that some but not all, of these infinite game trees correspond to games that cannot be settled in a finite number of moves if both players are playing perfectly. We find it useful to view those games as 'dynamic ties'.

Once section 2 introduces our model quantitatively, the natural appearance of dynamic ties, appears as a justification a posteriori of our methodology. Our probabilistic assumptions do not rule out a counterpart of the cyclic games that may appear with games played on graphs. The catch is that the kind of ties that appear in game trees are essentially impossible to recognize as such by any finite procedure. So even though our model does not rule out ties, we must consider that whenever the model's parameter are such that ties appear with nonzero probability the games may well be impossible to solve even for players with unbounded look-ahead capabilities. This can also be seen as an intrinsic limitation of game-playing procedures that discard the cycle structure of the game graph entirely. [\*]

Section 3 separates the model's parameters into two basic groups. The first group, called the internal structure  $K$  governs the behavior of games before their actual termination by determining the probability distribution of the number of legal options. The second, called the external structure consists of a probability of termination  $(1-b)$  and a probability  $p$  that a terminated game is a win for whoever turn it is to play. The external structure may be matched to the internal structure when the parameter  $p$  is equal to a special value  $\xi_K$ . When this happens, internal and external nodes have the same probability of being first player wins, and our analysis remains valid even if external nodes ap-

\* Hash coding techniques are sometimes used to detect nodes that have been previously examined. While this technique is intended primarily to eliminate duplicate expansions of a node's successors in the acyclic case, it has the potential of avoiding cyclic expansions. Therefore, game-playing procedures that incorporate hash-coding may recognize ties in a finite number of steps.

pear according to some unspecified scheme. This scheme does not need to obey our general probabilistic assumption, and this remark extends the scope of our discussion. For example, external nodes could appear predominantly as children of bushy nodes, or they could all be located at a given fixed depth, as in most former models. In the matched case, the exact conditions for termination influence only the game's length, not its outcome. Furthermore, we show that some internal structures, called inert, have the property that they do not allow ties, games of infinite length, even if the probability of termination is arbitrarily low. The key to inertness is shown to be the fact that those games allow a large standard deviation in the number of legal options available from a typical game position. This can happen for arbitrarily bushy game structures. Inert structures are appropriate for describing long and bushy games in which ties almost never occur. The geometric distribution is shown to be a convenient special case appropriate for such a description.

The use of the model in the comparative analysis of the complexity of game-solving procedures is exemplified in section 4. An expression for the branching factor of the standard depth-first search procedure SOLVE is given. Various procedures that improve on SOLVE are also discussed. The lowest probability of termination that makes a game-solving procedure almost surely terminate is suggested as a good measure of the limitations inherent in that particular procedure.

Section 5 extends the model to include asymmetrical partisan games. In our probabilistic perspective, partisan games are games that



allow the probability distribution of the number of legal options to depend explicitly on whose turn it is to play. The winning probability for tip nodes may also depend on who played last. The discussion is not significantly more complex than that for the impartial case which is used in the rest of the paper mostly for convenience. The world of partisan games, however, is significantly richer than that of impartial ones. Most real games have partisan rules and this often translates into a partisan probabilistic model. The partisan version of our model may exhibit an amusing phenomenon, illustrated in section 5. One player may have a decisive advantage in a game lasting many moves, while the other player only stands a chance if the game lasts very few moves. Partisan games can also be totally unfair. If one of the players cannot win a terminal position while the other player cannot lose one, the best the former player can hope for is a tie. In this very special case, virtually all internal structures will yield a nonzero probability for ties if the probability of termination is low enough. [\*]

Section 6 outlines some ideas relating to the propagation of errors in the evaluation of the status of a game tree. This issue was the initial motivation for the model presented here. Program designers have been assuming for decades that the quality of a game-playing decision based on minimaxing could only improve with a deeper search. Yet it was recently discovered [20] that the standard theoretical game model that is traditionally [15,17,23,26] used supports the opposite view, in spite

---

\* One of the rare exceptions is the geometric distribution (see Section 3). If the game's internal structure  $K$  is the same for both players, then exceptions to this statement can only be obtained if  $1-K$  is its own inverse. At this writing, the geometric distribution is the only such example we know of.

of the fact that minimax programs have been performing reasonably well for decades. Theoretically, the deeper a minimax search, the worse the decision. Nau [20] termed this phenomenon pathological. Minimax search performs much better in practice than it has any right to in theory ! Explanations have been attempted [24,25] , but the question is still essentially open.

Our model supports the view that a tractable game will not be pathological, while an intractable game, a game in which dynamic ties appear naturally, normally will be pathological. In this context, pathology appears as a property of the internal structure of a game. A minimax estimate can be expected to discriminate between good and bad moves better than the static evaluation function it is based on, provided the internal structure of the game is inert. To put it bluntly, pathology was observed on former models that used a constant branching degree  $d$  because the internal structure of these models ( $K(z) = z^d$ ) is not inert. This remark could also be seen as an indication that the games people actually play are somehow better represented by an inert structure than by a non-inert one. [\*]

In actual game-playing procedures, it would be foolish to hope for an exhaustive search of the game tree. For all situations that are not end-game positions, the search has to be truncated at some point, yet truncating the search tree at a fixed depth turns out to be a poor idea

---

\* It is not our intention to say that chess, say, can be represented accurately by a recursive random game. However, we believe that the concepts presented here are general enough to have a counterpart in the domain of real games. Any reference to such games is to be taken in a figurative sense.

in practice. Consequently, pursuing the search of game trees only from certain positions called non-quietest was recognized very early [27] as an effective scheme. Section 7 presents an analysis of quietest, and demonstrates that our model is flexible enough to allow an investigation of the theoretical reasons for the advantage of quietest-based search. Section 7 contains a theoretical confirmation of the usefulness of quietest and establishes an appropriate criterion for truncating the search.

Section 8 presents an alternative to the minimax rule of processing the information obtained from nodes farther down in the game tree. The alternative suggested is to propagate winning estimates in the same way winning probabilities would. This results in a product propagation rule. A simple minded analysis of that rule is given in that section.

## 1. Introduction

The games we consider here are two-player, perfect information games in which the players take alternate turns. The most general representation of such games is a directed, possibly infinite graph whose vertices represent positions and whose edges represent legal moves from one position to another. A vertex from which there is no edge, hence no legal move, represents a terminal position. A label is assigned to each terminal position; the outcome of the game depends on what label is associated to the final position.

The two standard ways of labeling terminal positions are traditionally [10,5] called the "normal play rule" and the "misère play rule". The normal play rule states that all terminal nodes are labeled L (loss), indicating that whoever has to play from a terminal position is the loser. The misère play rule states the opposite, that all terminal nodes are labeled W (win); whoever is unable to play when called upon to do so is the winner. Some authors also allow finite games to be drawn by labeling some terminal positions T (tie). For the sake of clarity, the arguments developed in this paper restrict T to games that cannot be won by either player in a finite number of moves; a terminal position is either a loss or a win. Figure 1.1.a gives examples of such games.

Figure 1.1.a : Some games

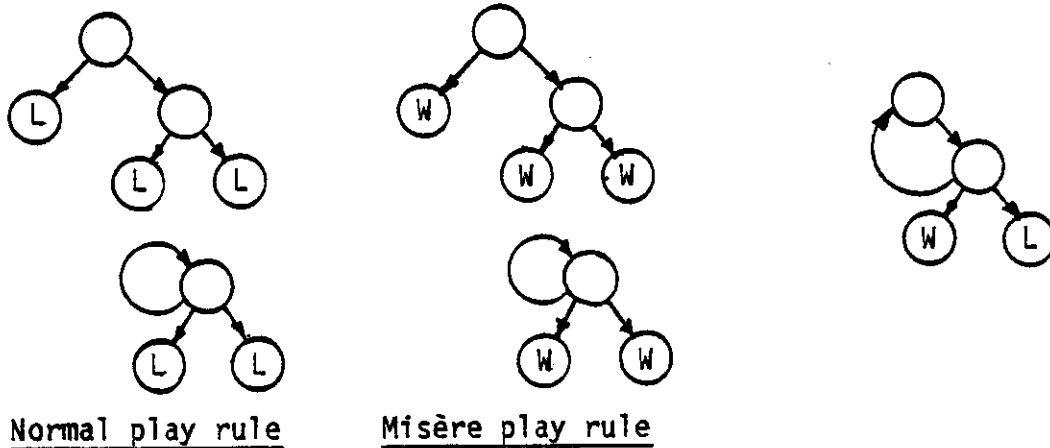


Figure 1.1.b : Extended labeling of some games .

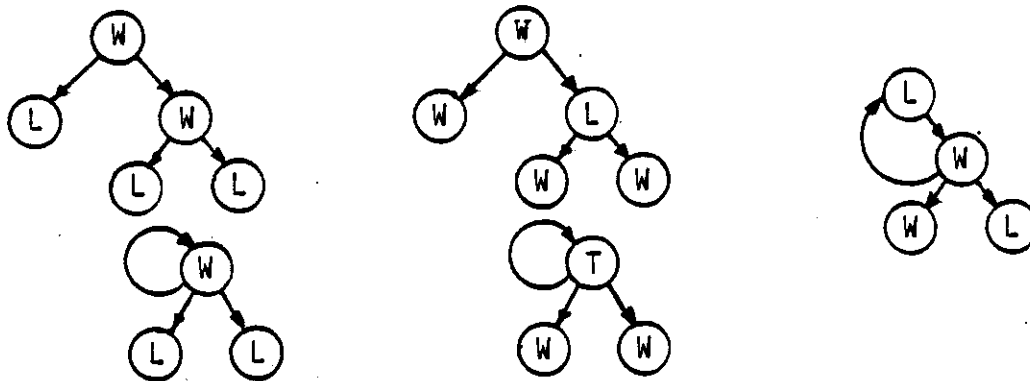


Figure 1.1.b shows how to extend the labeling of terminal nodes to internal nodes. Since perfect players always leave their opponent with the worst possible position, a next player win (label W) is a node with at least one child labeled L, a next player loss (label L) is a node for which all children are labeled W. If the iterative application of those two labeling rules eventually leaves some nodes unlabeled, those are ties (label T). One should be aware, however, that there is more to game playing than this labeling procedure: it is not sufficient when called upon to play from a W node to move into an L node to insure a win in a finite number of moves. This is illustrated by the last example

given in Figure 1.1 where the winner, the player who made the second move, can actually postpone his victory forever while always leaving his opponent in a losing (L) position. Consequently, it is convenient to view a perfect winner as winning in the least possible number of moves against a perfect loser who attempts to postpone his loss as long as possible.

A convenient way to deal with games is to turn them into game trees by unfolding them, duplicating any node with more than one predecessor. This results in an infinite tree whenever the original game contained any cycles or was infinite itself.

The purpose of this paper is to make some observations on the statistical behavior of an interesting ensemble of such game trees. The spectrum of features our model exhibits is rich enough to encompass some characteristics that we feel are essential to the analysis of heuristic game-playing strategies, yet are generally overlooked. Our model is opposed to the now classical analysis (see [10], pp 139-140 or [28]) that assumes equiprobability among games of given depth while making the elegant but not so innocent assumption that only structurally distinct nodes need to be counted as distinct. This simple assumption turns out to be a very difficult one to check against the 'real world'. A game like chess may or may not have many structurally identical positions that look totally different on the board. If the statistical features that one can derive [28] from such simplified models is ever to be useful in practice, then a totally different approach is required. This paper presents such an approach.

We define an ensemble of possible games which may contain all conceivable game trees, and assign reasonable probabilities to sets of those games. Our statistical assumptions have been kept as simple as possible to serve the purpose of obtaining an insight into the essential concepts of Game Solving. Possible extensions of our basic model are discussed in section 5. The results presented here are to be taken in the sense of giving a general feeling of the issues involved in a statistical approach to the notoriously difficult problem of automated game-playing. We suspect that some important aspects of the statistics of games that people actually play are unlikely to show up in such a simplistic frame; however, we do believe that all the features that do show up and are discussed at length here should not be disregarded.

The motivation for a statistical model of games is clear. The combinatorial complexity of a game tree effectively hides away its useful information and makes the outcome of a game highly unpredictable. Furthermore, even though the outcome eventually depends on minute details in the game tree, few players would fail to recognize that like situations are likely to yield like outcomes. This alone would make a statistical approach legitimate. Moreover, since our primary goal is to capture the essence of some of the general features of games through a probabilistic representation, the study of any particular game, no matter how 'typical' we think that particular game is, would be insufficient.

## 2. Impartial Recursive Random Games

In our basic model, we consider that every nonterminal position allows  $n$  legal moves with probability  $f_n$ , regardless of how this position was reached or of whose turn it is to play. Consequently, we call such games impartial. A position is terminal with probability  $f_0$ . We will always assume that  $f_0$  is not equal to zero. Such positions are labeled independently  $W$ , with probability  $p$  and  $L$ , with probability  $q = 1-p$ .

Symbolically, the above definition is best expressed by the following recursive definition of what we call the family  $\mathcal{G}$  of impartial recursive random games.

### Definition 2.1

$$\mathcal{G} = f_0(p\langle W \rangle + q\langle L \rangle) + f_1 \underset{\mathcal{G}}{\mathcal{G}} + f_2 \underset{\mathcal{G}}{\overset{0}{\swarrow \searrow}} + f_3 \underset{\mathcal{G}}{\overset{0}{\swarrow \searrow}} + \dots$$

This formula states that a member of  $\mathcal{G}$  is either a terminal node, a game with one initial option (probability  $f_1$ ), or a game with two initial options (probability  $f_2$ ), etc.

### Definition 2.2

The generating function  $F(z) = \sum_{n=0}^{\infty} f_n z^n$  is called the tree structure of  $\mathcal{G}$ . The pair  $(F, p)$  is referred to as the statistics of  $\mathcal{G}$ .

Definition 2.1 introduces a notation we will use extensively. Script letters (such as  $\mathcal{G}$ ) denote a set of game trees, while a symbol like  $\underset{\mathcal{G}}{\overset{0}{\swarrow \searrow}}$  denotes all games with two options, each a member of  $\mathcal{G}$ .  $\langle W \rangle$  and  $\langle L \rangle$  denote the singletons corresponding to labeled terminal posi-



tions. A set of games is always to be used with its probability as a 'coefficient'. Coefficient 1 is implicit for  $G$  itself. Hence, all such formulas have a numeric counterpart easily obtained by removing all symbols denoting games. The numeric counterpart of Definition 2.1 is:

$$1 = f_0(p+q) + f_1 + f_2 + f_3 + \dots$$

that is:

$$1 = F(1)$$

This result states that the right-hand side of the formula in Definition 2.1 enumerates all possible games. In more complex cases, however, such a numerical result will convey nontrivial information. Actually, the symbolic version of the formula is merely a condensed explanation for the numerical counterpart. The formula states some equality between sets, while the numerical equation asserts the equality of their probabilities.

Figure 2.1.a : Probability of some small games

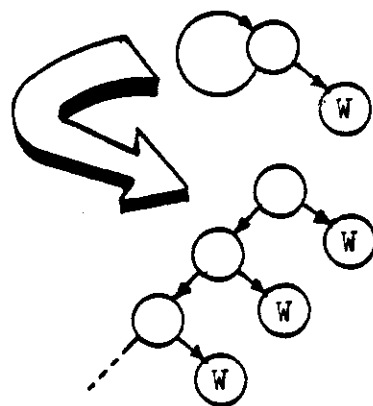
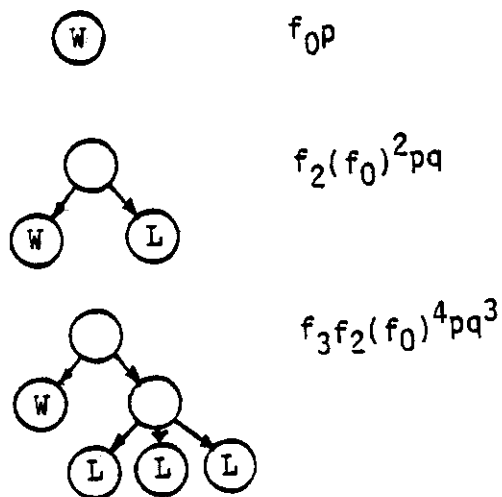


Figure 2.1.b

Figure 2.1.a gives the probability of some small games according to our basic model. Figure 2.1.b gives an example of a game that yields an infinite tree when unfolded and, therefore, almost never (probability 0) occurs in our model. However, even though only finite games have a nonzero probability, sets of infinite games may still have nonzero probability.

Let us consider the set  $\mathcal{F}$  of finite games ( $\mathcal{F} \subseteq \mathcal{G}$ ) and compute its probability  $p_{\mathcal{F}}$ .  $\mathcal{F}$  and  $p_{\mathcal{F}}$  satisfy the following equation, which states that a finite tree is either a leaf or an internal node with only finite offsprings :

$$p_{\mathcal{F}} = f_0(p\langle W \rangle + q\langle L \rangle) + f_1(p_{\mathcal{F}}) \begin{array}{c} \emptyset \\ | \\ \mathcal{F} \end{array} + f_2(p_{\mathcal{F}})^2 \begin{array}{c} \emptyset \backslash \\ | \quad | \\ \mathcal{F} \quad \mathcal{F} \end{array} + f_3(p_{\mathcal{F}})^3 \begin{array}{c} \emptyset \backslash \\ | \quad | \quad | \\ \mathcal{F} \quad \mathcal{F} \quad \mathcal{F} \end{array} + \dots$$

The numerical counterpart of this formula is the fixed point equation :  $p_F = F(p_F)$ .

This equation is always trivially satisfied for  $p_F=1$  and  $f = G$ . However, this is not always the correct solution to our problem. The above fixed point equation may have more than one solution. We need a more refined analysis that consists in asserting that  $f$  is the union of all trees of finite height. Let us define  $f_n$  as the set of finite games of height at most  $n$ , where the height of a finite tree is defined as the longest path from the root (a leaf has height 0, a tree with only one internal node has height 1). We then have a correct inductive definition of  $f_n$ .

Definition 2.3

$$p_F^0 \quad f_0 = f_0(p\langle W \rangle + q\langle L \rangle)$$

$$p_F^{n+1} f_{n+1} = f_0(p\langle W \rangle + q\langle L \rangle) + f_1(p_F^n) \begin{array}{c} 0 \\ | \\ f_n \end{array} + f_2(p_F^n)^2 \begin{array}{c} 0 \\ / \quad \backslash \\ f_n \quad f_n \end{array} \\ + f_3(p_F^n)^3 \begin{array}{c} 0 \\ / \quad | \quad \backslash \\ f_n \quad f_n \quad f_n \end{array} + \dots$$

The numerical counterpart of Definition 2.3 permits us to compute the probability  $p_F^n$  of  $f_n$  :

$$\left\{ \begin{array}{l} p_F^0 = f_0 \\ p_F^{n+1} = F(p_F^n) \end{array} \right.$$

The value of the solution  $p_F = \lim_{n \rightarrow \infty} p_F^n$  depends critically on the parameter  $r = \frac{dF}{dz}(1)$ , as shown in Figure 2.2 .

Definition 2.4

The branching factor  $r$  of a family of trees of structure  $F$  is the expected number of offsprings of a typical node, namely the quantity  $r = \frac{dF}{dz}(1)$  .

Theorem 2.1 ( Galton-Watson processes (1874) [1] )

The members of a family of trees of structure  $F$  are finite with probability 1 if and only if the branching factor  $r$  of the family is less than or equal to 1. Otherwise the probability of finiteness is the least fixed point of  $F$ .

It is not at all difficult, although beyond the scope of this simple illustration, to see that, for  $r=1$ , the average height of a tree is infinite; although infinite trees have zero probability.

The above introductory computation was concerned with the underlying tree structure of our basic model, not with its game properties. The win-loss status of the tip nodes was irrelevant. Now, however, we consider our trees as games by dividing  $\mathcal{G}$  into the three basic families : next player wins, next player losses, and ties, respectively denoted by  $W$  ,  $L$  and  $T$  . Our general scheme is similar to the one used for the above illustration, the main difference being that the concept of a

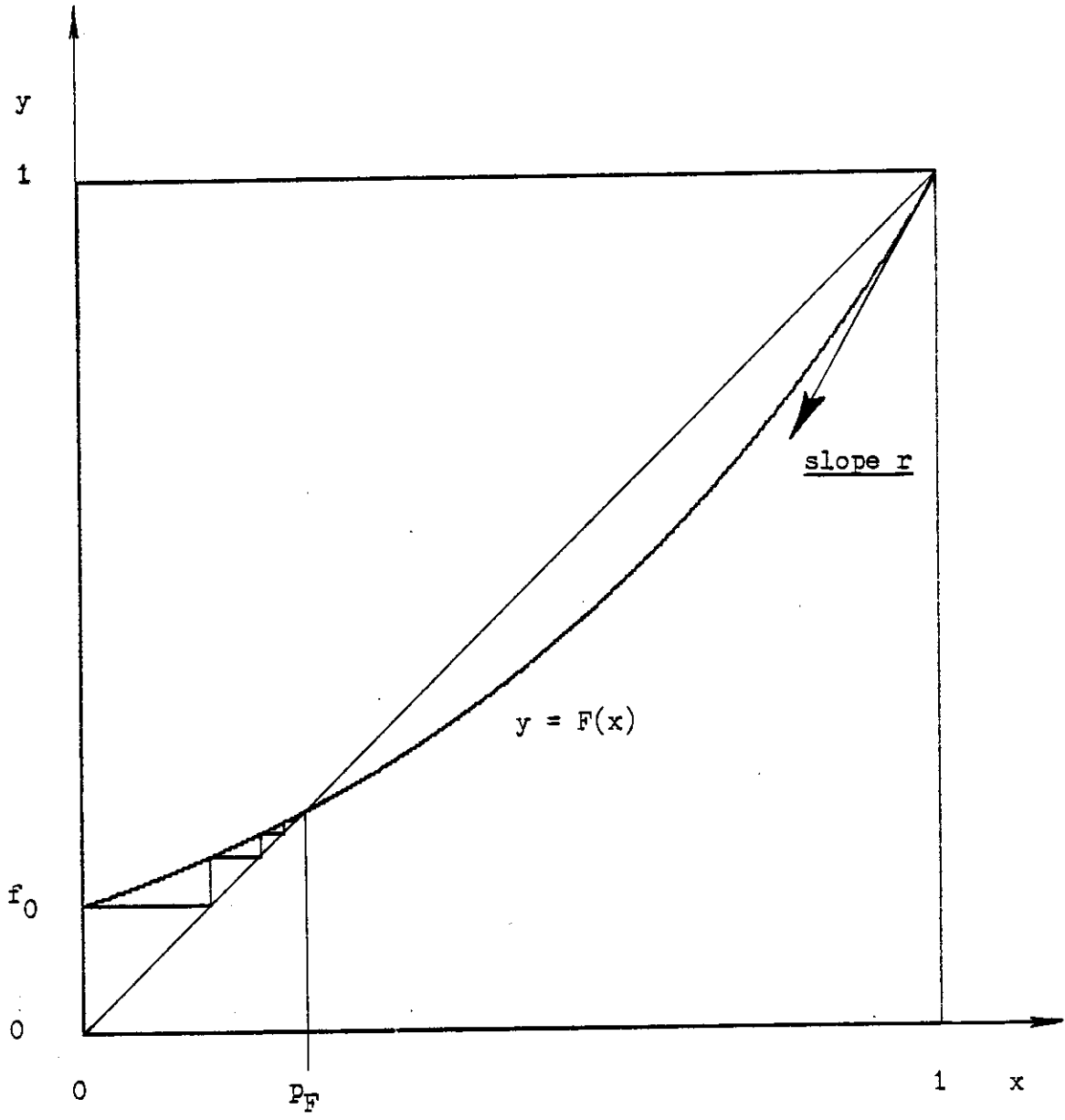


Figure 2.2.a : When  $r > 1$  not all game trees are finite

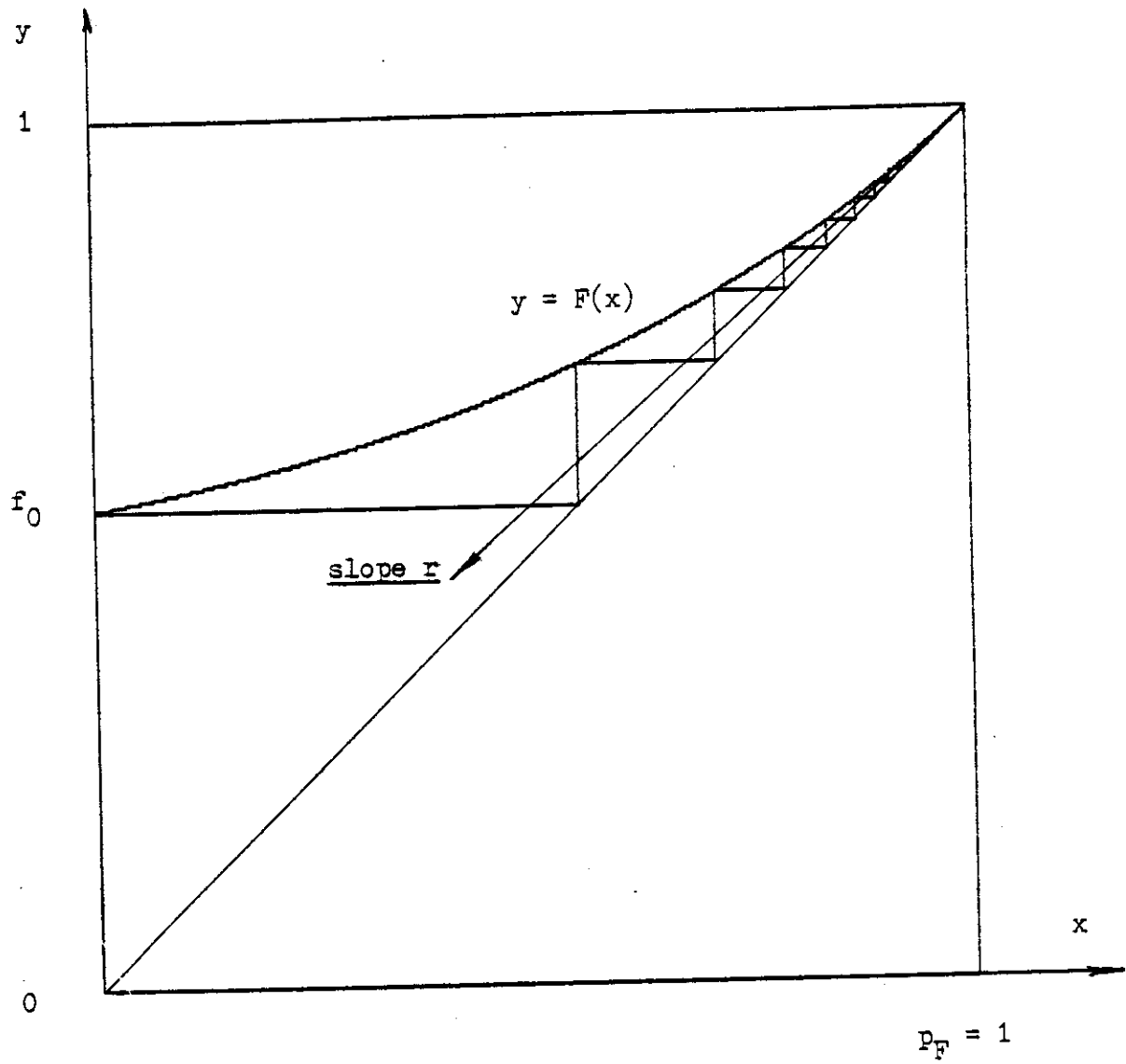


Figure 2.2.b : When  $r \leq 1$  all game trees are finite

tree's height will be replaced by that of a game's length.

Definition 2.5

The length of a game is the number of moves two perfect players would play before reaching a terminal position. Recall that perfect players try to win in the least possible number of moves or to postpone an unavoidable loss as long as possible.

Consider the family  $W_n$  of first player wins of length  $n$  or less and the family  $L_n$  of forced losses of length  $n$  or less. Clearly,  $W$  is the union of all  $W_n$  and  $L$  is the union of all  $L_n$  while  $\tau$  includes everything else, that is, all games of infinite length. Denoting the complement of  $L_n$  by  $L/n$  we obtain :

$$p_L^0 L_0 = f_{0q} \langle L \rangle$$

$$p_W^0 W_0 = f_{0p} \langle W \rangle$$

$$p_L^{n+1} L_{n+1} = f_{0q} \langle L \rangle + f_1(p_W^n) \frac{q}{W_n} + f_2(p_W^n)^2 \frac{q}{W_n W_n} + \dots$$

$$p_W^{n+1} W_{n+1} = f_{0p} \langle W \rangle + f_1(p_L^n) \frac{q}{L_n} + f_2(p_L^n) \frac{q}{L_n G} + p_L^n (1 - p_L^n) \frac{q}{L/n L_n}$$

$$+ f_3(p_L^n / q + p_L^n (1 - p_L^n) / q + p_L^n (1 - p_L^n)^2 / q) + \dots$$

The numerical counterpart of those formulas is

$$\begin{cases} p_L^0 = f_{0q} \\ p_W^0 = f_{0p} \end{cases} \quad \begin{cases} p_L^{n+1} = F(p_W^n) - f_{0p} \\ p_W^{n+1} = f_{0p+1} - F(1 - p_L^n) \end{cases}$$

These relations make it natural to introduce a new function,  $G$ .

Definition 2.6

The game structure  $G$  of an ensemble  $\mathcal{G}$  of statistics  $(F, p)$  is the quantity  $G(z) = f_{0p} + 1 - F(z)$ .

The above relations can be written in a way that deserves to be expressed as a Theorem by using  $G$ .

Theorem 2.2

Both the probability  $u_n = 1 - p_L^n$  that the first player can avoid losing in  $n$  moves or less, and the probability  $v_n = p_W^n$  that he can win in  $n$  moves or less, are given by the following recurrence:

$$\begin{cases} u_0 = G(0) \\ v_0 = G(1) \end{cases} \quad \begin{cases} u_{n+1} = G(v_n) \\ v_{n+1} = G(u_n) \end{cases}$$

Figure 2.3 illustrates the rest of the discussion. One can draw two 'spirals'. The solid line gives  $u_n$  for even values of  $n$  and  $v_n$  for odd values of  $n$ . The dotted line yields  $u_n$  for odd indices and  $v_n$  for even indices. The fixed point of  $G$  is  $t$ ;  $m$  is the slope of  $F$ , the opposite of that of  $G$ , at point  $t$ . Ties almost never occur (see Figure 2.3.a) when both spirals converge toward the fixed point  $t$ . This can only happen when  $t$  is a stable fixed point of  $G$  (i.e.  $m \leq 1$ ). If  $m$  is larger than 1, ties always occur with nonzero probability (see Figure



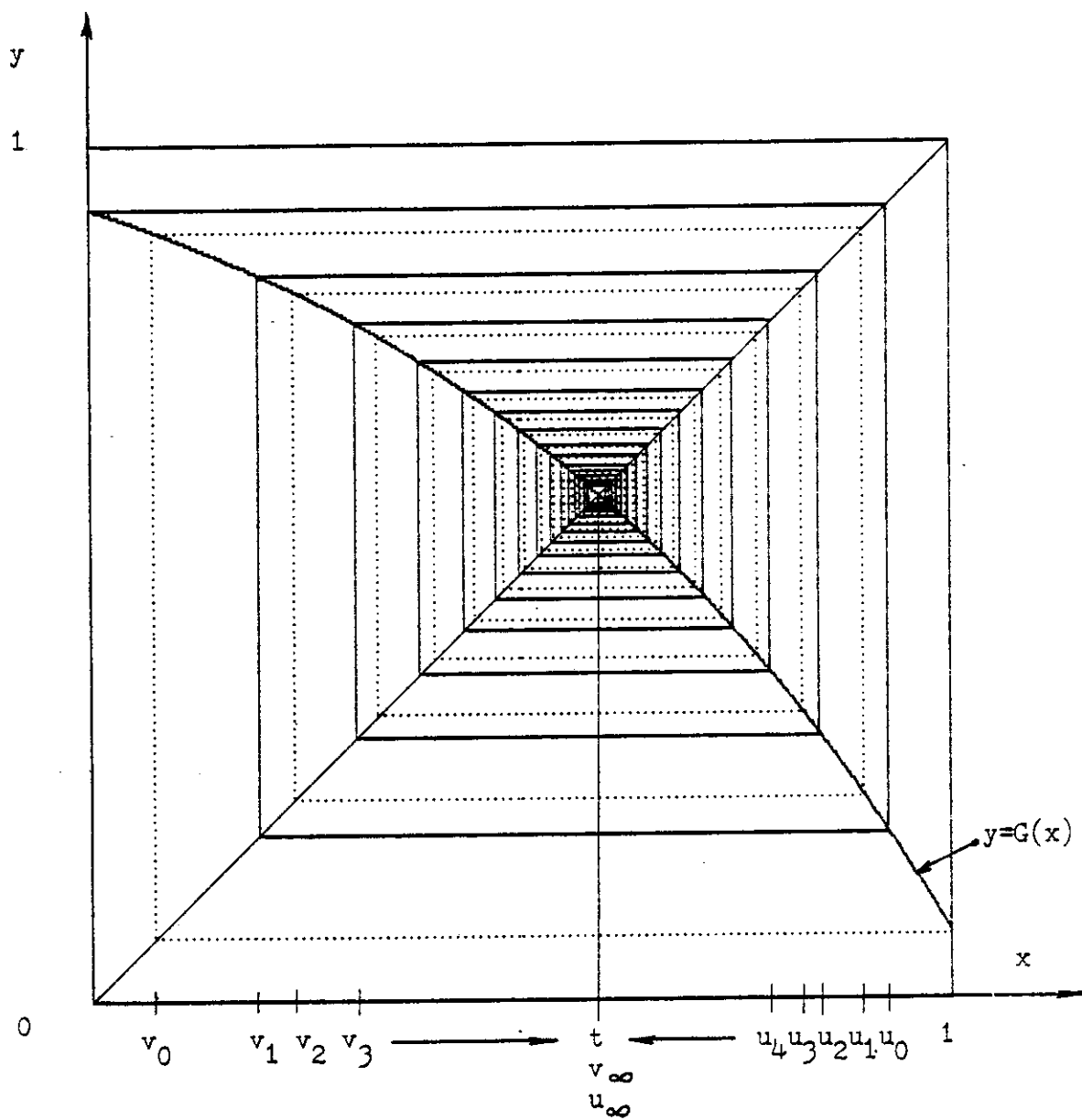


Figure 2.3.a : Ties almost never occur only if  $m \leq 1$

$v_n$  is the probability that the first player can win in  $n$  moves or less  
 $u_n$  is the probability that the first player can avoid losing in  $n$  moves or less

$m$  is the slope of  $-G$  at point  $t$

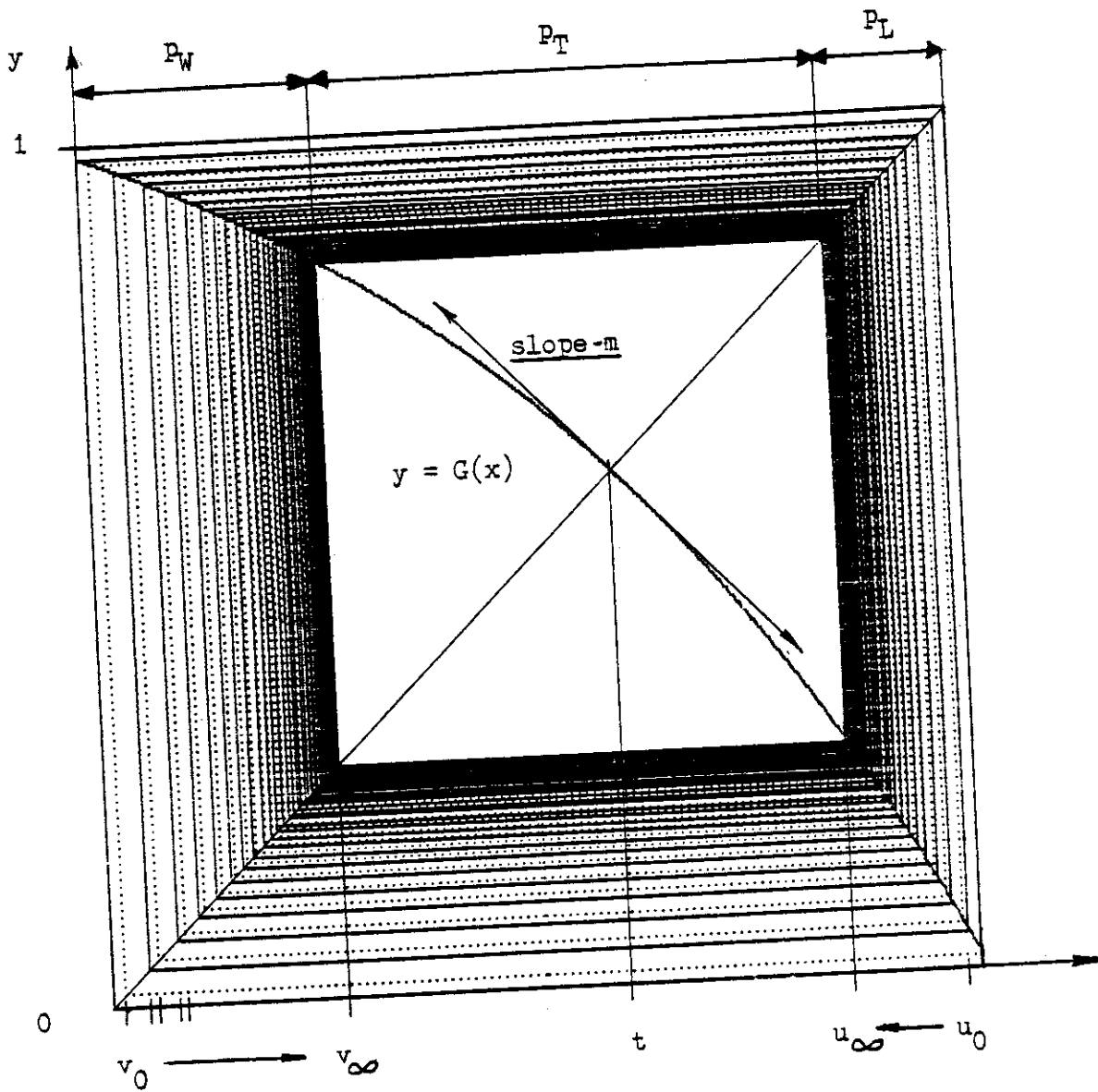


Figure 2.3.b : When  $m > 1$  , ties occur with nonzero probability

t is the fixed point of G  
 m is the slope of -G at point t

2.3.b). If  $m$  is less than or equal to one, the situation illustrated in Figure 2.3.a, where ties almost never occur, is the most common case. One can find functions  $G$  that have stable 'two-cycles' in spite of the fact that  $m < 1$ . [\*] The 'spirals' cannot get past the outermost two-cycle of  $G$  and ties will therefore appear with nonzero probability.

A case of particular interest is the limiting case where ties almost never occur and  $m=1$ , since it almost always (probability 1) yields games which are finite in length when played perfectly, although the expected length of these games is infinite. We call such a statistic  $(F,p)$  a transition statistic. We view a transition statistic as describing the state of affairs commonly called a phase transition in various domains of statistical physics (thermodynamics, ferromagnetism ... ) where a given property changes abruptly with the minute variation of some parameter. As is the case in theoretical physics, discontinuities in our statistical model are brought about due to the fact that infinity is allowed. This leads us to adopt the vocabulary of physicists and to acknowledge the fact that there is more to a phase transition than a sharp transition.

When dealing with a transition statistic, one can expect a game that may well be impossible to play perfectly for players with finite look-ahead capability, yet is almost surely (probability 1) a forced

\* Examples of such 'strange' functions  $G$  are given in Appendix A. A necessary and sufficient condition for the absence of such two-cycles in  $G$  appears in section 5 (Theorem 5.1) in a more general context. Theorem 5.1 implies that ties almost never occur if and only if the two curves of equations  $x=G(y)$  and  $y=G(x)$  have only one point of intersection. This is 'usually' the case when  $m < 1$  so that  $m < 1$  is 'almost' a necessary and sufficient condition.

loss for one of the players. This certainly looks like the type of situation that humans and computers are likely to encounter when playing interesting games whose exact analysis is far beyond their analytic capabilities. Statistics close to the transition point ( $m=1$ ) are therefore more likely to yield a realistic model of interesting games. Games below the transition point ( $m<1$ ) are expected to last very few moves and to lack challenge. Games beyond the transition point ( $m>1$ ) are too challenging even for players with infinite look-ahead, since there is a nonzero probability that such games cannot be settled in a finite number of moves. Listed on Figure 2.4 are some examples of transition statistics with the normal rule of play namely that whoever is unable to move when called upon to do so is the loser. Those could be used to approximate roughly, for analytic purposes, the statistics of some real games.

Figure 2.4 : Examples of transition statistics ( $m=1$ )  
for Impartial Recursive Random Games in Normal Play ( $p=0$ )

Distribution	Generating Function	Branching Factor	Probability of Win
$f_i$	$F = \sum_{n=0}^{\infty} f_n z^n$	$r$	$t$
Poisson	$e e^{z-1}$	$e$	$1 - \frac{1}{e}$
Binomial	$[1 + \frac{1}{d}(\frac{1+d}{d})^{d-1}(z-1)]^d$	$(\frac{1+d}{d})^{d-1}$	$1 - (\frac{d}{1+d})^d$
Truncated Poisson	$1 + \frac{d}{d + \ln(d) - 1} [(\frac{de^d}{e})z - 1 - 1]$	$d$	$1 - \frac{\ln(d)}{d + \ln(d) - 1}$
Degree 0 or $d$	$1 + \frac{1}{d}(1+d)^{1 - \frac{1}{d}}(z^d - 1)$	$(1+d)^{1 - \frac{1}{d}}$	$(1+d)^{-\frac{1}{d}}$

### 3. Matched Statistics and Inert Structures.

In a finite horizon model of games it has been observed [23] that a certain winning probability for the tip nodes (our parameter  $p$ ) implies an identical probability for all internal nodes. A winning probability for the tip nodes satisfying this property will be said to match the structure of the tree. We will now define and characterize matched statistics within our model and explain why matching actually allows us to analyze successfully game trees in which leaves appear with less statistical regularity than we have assumed so far.

First, we find it convenient to separate the statistics  $(F,p)$  of a family of games into its two natural components: that which governs the termination of the game and that which governs the internal structure of a nonterminated game.

#### Definition 3.1

The internal structure of a family of impartial recursive random games is the generating function  $K(z) = \sum_{n=1}^{\infty} k_n z^n$ , where  $k_n = \frac{f_n}{1-f_0}$  is the probability for an internal node to have  $n$  offsprings.

#### Definition 3.2

The external structure of a family of impartial recursive random games is the pair  $(p,b)$  where  $p$  is the winning probability for tip nodes and  $b=1-f_0$  is the probability that a node is terminal.

The quantity  $b$  is introduced as an independent parameter that can be seen as controlling the depth of the tree and that, together with  $K$ , fully describes the statistical structure  $F$  via:  $F(z) = (1-b)+bK(z)$ .

The rest of this section will assume that the internal structure  $K$  is given, and will investigate the variations of  $p_L$ ,  $p_T$  and  $p_W$  for various external structures.

Definition 3.3

The external structure  $(p,b)$  is said to match the internal structure  $K$  if the probability that any given node is a win (resp. a loss) is equal to the probability that an external node is a win (resp. a loss).

In other words, internal and external structures match when  $p_W = p$  and  $p_L = q = 1-p$ .

Definition 3.3 clearly implies that matching may only occur when ties have zero probability. The condition  $p=p_W$  then reduces (see Figure 2.3.a) to  $p = G(p)$ , where  $G(z) = f_0 p + 1 - F(z) = (1-b)p + b(1-K(z))$ , as previously defined. This can be rewritten  $p = 1-K(p)$ , provided  $b \neq 0$ .

Hence, calling  $\tilde{c}_K$  the fixed point of  $1-K$ , we have  $p = \tilde{c}_K$ . Matching occurs under this condition with the added restriction that ties actually have zero probability; this formally implies that  $\frac{dF}{dz}(\tilde{c}_K) \leq 1$  or  $b \leq [\frac{dK}{dz}(\tilde{c}_K)]^{-1}$ . Hence:

Theorem 3.1

An external structure  $(p,b)$  matches an internal structure  $K$  if and only if

- 1/  $p = \tilde{c}_K$  (where  $\tilde{c}_K$  is the fixed point of  $1-K$ )
- and 2/  $b \leq [\frac{dK}{dz}(\tilde{c}_K)]^{-1}$
- and 3/ Ties almost never occur. [\*]

It may appear surprising that the winning probability  $p_W = p = \xi_K$  remains the same over a wide range of  $b$ . This, however, is merely a consequence of the fact that when terminal and internal nodes have the same winning probability, the frequency of terminal nodes is essentially irrelevant. Actually, when the matching condition  $p = \xi_K$  is satisfied, the way terminal nodes appear is completely irrelevant to the computation of the winning probability.  $p_W$  will always be equal to  $\xi_K$  as long as ties are somehow forced to have zero probability.

For example, Pearl [23] uses  $K(z) = z^d$  in his constant length model, where all terminal nodes appear at the same fixed depth, and proves that when tip nodes are independently chosen to be wins with probability  $\xi_K$ , all internal nodes are also wins with probability  $\xi_K$ . In fact our very notation  $\xi_K$  is merely a generalization of a notation  $\xi_d$  first introduced by Baudet [3] in such a model (Baudet also used  $K(z) = z^d$ ).

One major distinction between our model and the model used by Pearl and Baudet is the stability of the matching condition  $p = \xi_K$ . Pearl's model exhibits a high instability since  $p_W$  tends to depart from  $\xi_K$  as the (even) height of the game increases. Quite rapidly [23],  $p_W$  will be extremely close to either 0 or 1 (depending on whether  $p$  is below or above  $\xi_K$ ). By contrast, in our model, (see Figure 3.1) the farther away tip nodes are expected to be the closer  $p_W$  will be to  $\xi_K$ .

\* The second condition is redundant. It is implied by the third. However, condition 2 is interesting as a closed form condition that is usually sufficient to ensure condition 3. The third condition should be kept because  $m < 1$  (section 2) does not always suffice to prevent ties from occurring as shown in Appendix A.



We conjecture that the paradoxical instability found in fixed horizon models is due mostly to a built-in bias toward one player. Those models imply that whoever plays second plays last. What this amounts to is a highly asymmetrical relation between the situations of the two players. This bias does not disappear in large game trees. In fact, Pearl showed that the effect is just opposite: the higher the tree, the greater the bias. These remarks are a warning against the fact that simple models may exhibit features that may or may not have much to do with the original thing. This most probably applies to the model discussed here as well, and explains why we found it so important to match the conclusions derived from the model against one's intuition of what kind of features one would find in a complex game.

The diagrams of Figure 3.1 summarize our discussion so far. Those diagrams are all plotted for a given internal structure (here  $K(z) = z^2$ ) and each diagram corresponds to a given  $p$ . The parameter  $b$  is used as the horizontal coordinate. The diagrams are to be read as follows.

Given a statistic  $(F,p)$  where  $F(z) = (1-b)+bK(z)$ , one considers the vertical slice whose abscissa is  $b$  in the diagram corresponding to  $p$ . Such a slice is of length 1 and the share of the slice that falls into a given region of the diagram (LOSS, WIN or TIE) gives the probability of the game's status (loss, win or tie) for the statistics  $(F,p)$ .

The diagrams are obtained as follows. The equation of the boundary between the WIN and LOSS region as implied by Figure 2.3.a, is:

$$y = G_b(y) \quad (\text{in the binary case : } y = (1-b)p + b(1-y^2))$$

The notation  $G_b$  is equivalent to our former  $G$  but stresses the fact that  $G_b(z) = (1-b)p + b(1-K(z))$  explicitly depends on  $b$ .

The equation of the TIE region as implied by Figure 2.3.b, is: [\*]

$$0 = \frac{y - G_b(G_b(y))}{y - G_b(y)}$$

(in the binary case,  $0 = b^2y^2 - by + 1 - ab$  , where  $a = (1-b)p + b$  )

The transition point at the intersection of the two boundaries is also on the curve of equation (independent of  $p$ ):

$$b = \left[ \frac{dK}{dz}(y) \right]^{-1} \quad (\text{in the binary case : } yb = \frac{1}{2})$$

This curve is the dotted line on Figure 3.1 .

The TIE region is not always as wide as it appears in Figure 3.1 . In fact, it may well be the sole vertical segment at  $b = 1$ . When this happens in the matched case, we call the game inert to emphasize the fact that such statistics need not be 'stabilized' by any provisions for early termination.

---

\* Here we assume that ties do not occur unless the second condition of Theorem 3.1 is violated. This is the most common case.

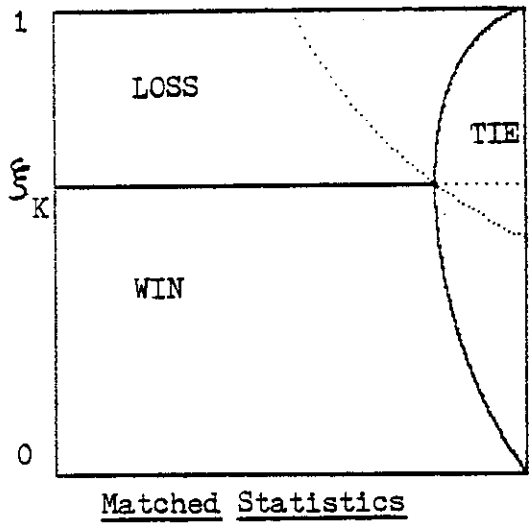
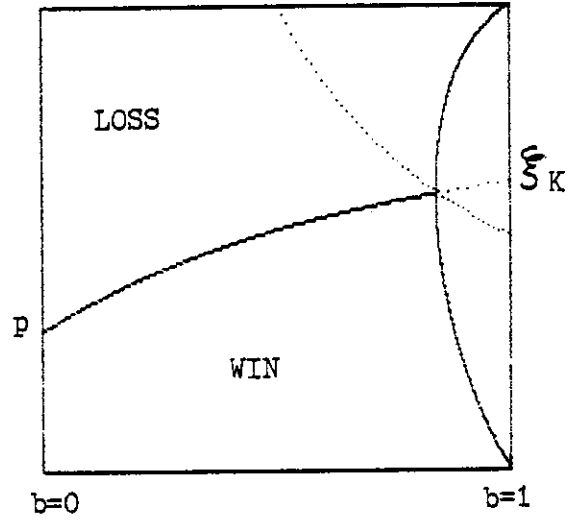
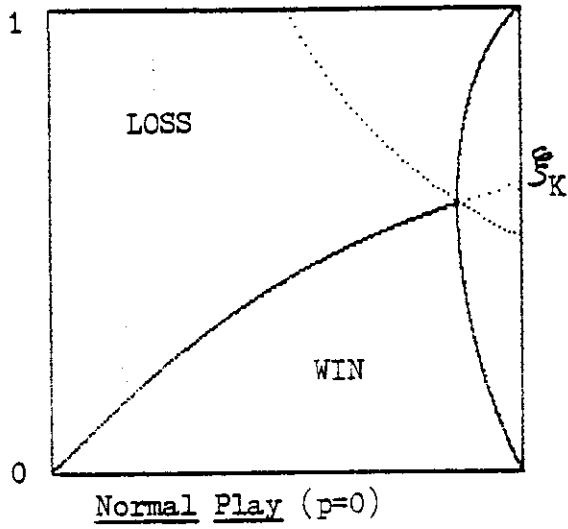
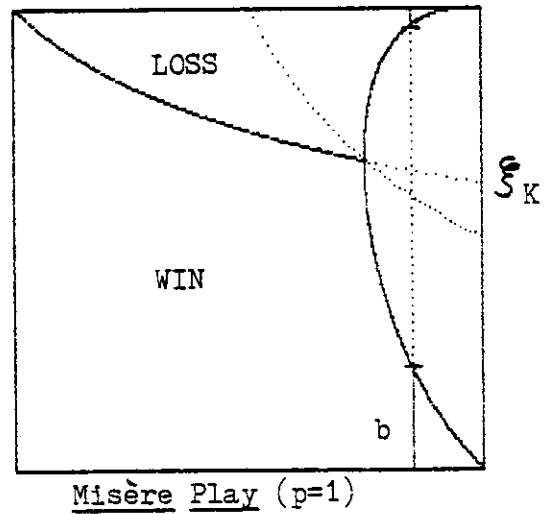
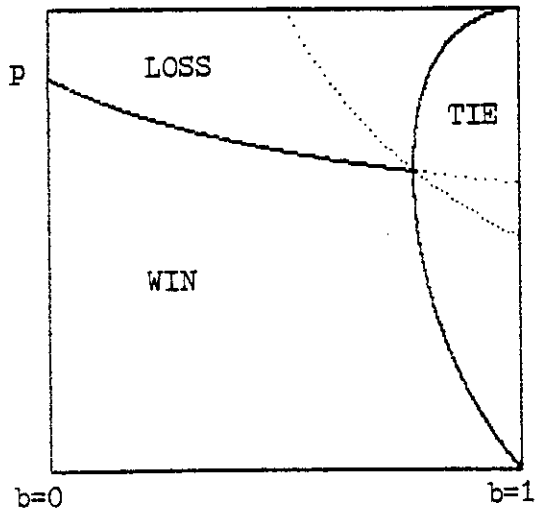
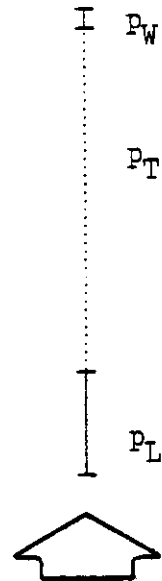


Figure 3.1  
 Probabilities  
 $P_W$ ,  $P_L$  and  $P_T$   
 in the basic model.



### Definition 3.4

An internal structure  $K$  is said to be inert when ties almost never occur in any game structure  $G(z) = (1-b)\hat{\xi}_K + b(1-K(z))$ . [\*]

With an inert internal structure, the probability that a game is a first person win is essentially independent of what happens at the tip nodes (external structure). This situation is certainly paradoxical and it is useful to consider a special case where such a behavior is easily understood without the machinery we have developed so far.

Consider the case of string games where a nonterminal node can only have one offspring. Using our previous notations  $K(z) = z$ ; hence,  $\hat{\xi}_K = \frac{1}{2}$  and  $\frac{dK}{dz}(\hat{\xi}_K) = 1$ . Now, a game of this kind is essentially a string of moves, where no choice is ever left to either player. This string is almost surely finite and is of length  $n$  with probability  $b^n(1-b)$ . Therefore the game will last an even number of moves with probability  $\frac{1}{1+b}$  and an odd number of moves with probability  $\frac{b}{1+b}$ . When  $b$  is close to unity, those two probabilities are close to  $\frac{1}{2}$ . That is to say that both players are equally likely to play last. Therefore advantaging the 'last' player by increasing  $p$  will not seriously bias the game in favor of either the first or the second player. Also, this example shows what  $\hat{\xi}_K = \frac{1}{2}$  really is, namely the winning probability for tip nodes that makes the winning probability for the root node independent of how early the game is expected to terminate.

\* Conditions for inertness are discussed in Appendix B. The definition of inertness makes it explicit only that ties do not occur in the matched case ( $p = \hat{\xi}_K$ ). At this writing, we do not know of any inert internal structure that would allow ties even if the winning probability  $p$  is not matched to it. The condition  $\frac{dK}{dz}(\hat{\xi}_K) \leq 1$  is a 'tight' necessary condition for inertness.

At this point, one probably expects inert games to be essentially similar to string games. This turns out not to be the case, and one can exhibit arbitrarily bushy inert games. More specifically, there are inert structures in which games with less than  $m$  options have probability 0, no matter how large one chooses  $m$  to be, provided games with many more options appear very often (making the game more bushy still ...). For example, a game in which all internal nodes have degree either 2 or 32 is inert, provided that binary nodes represent between 18 and 22.5% of all internal nodes, whereas a game whose nodes are of degree 2 or 50 is inert whenever the proportion of binary nodes is more than 9% but less than 26%. For reasons that relate to the topics we are now going to discuss, inert games still tend to be more common when nodes of degree one are allowed and/or when nodes of relatively large degrees may appear. For instance, a game cannot be inert when the degree of all non-terminal nodes is between 2 and 31.

Let us examine the most complex class of inert internal structures, that consisting of those games that would 'almost' be transition statistics (see section 2) if  $b$  was 'almost' unity. This corresponds to structures  $K$  for which  $\frac{dK}{dz}(\xi_K) = 1$ . It is convenient for the purpose of this study to consider  $t = \xi_K$  as a parameter and  $K$  as a possibly infinite set of positive coefficient  $k_n$  that add up to 1.  $K$  is then merely the unknown in the following two simultaneous parametric equations:

$$\left\{ \begin{array}{l} K(t) = 1-t \\ \frac{dK}{dz}(t) = 1 \end{array} \right.$$

This amounts to the following vectorial equation, adding a coefficient  $[-\ln(t)]$  for the sake of the geometric interpretation:

$$\sum_{n=1}^{\infty} k_n \begin{bmatrix} t^n \\ -nt^n \ln(t) \end{bmatrix} = \begin{bmatrix} 1-t \\ -t \ln(t) \end{bmatrix}$$

This means that point B of coordinates  $\begin{bmatrix} 1-t \\ -t \ln(t) \end{bmatrix}$  is the barycentre of the set of points  $A_n = \begin{bmatrix} t^n \\ -nt^n \ln(t) \end{bmatrix}$  where each point  $A_n$  is given the weight  $k_n$ . The points  $A_n$  are all on the curve of equation  $y = -x \ln(x)$  in the  $(x,y)$  plane, whereas point B is always on the symmetrical curve of equation  $y = -(1-x) \ln(1-x)$ . This situation is summarized in Figure 3.2.a : for a given  $t \leq \frac{1}{2}$ , B will be in the convex hull of the points  $A_n$ , and one can therefore express B as a positive weighted sum of the  $A_n$ . This yields K such that  $t = \xi_K$ . It is possible to do so, as illustrated in Figure 3.2.b, even if the weight of  $A_n$  is zero for n smaller than a given (arbitrarily large) integer m, provided that  $\frac{t}{1-t} > m$ . [\*]

The relative positions of B and the points  $A_n$  are given in Figure 3.2.c for various values of t. The main point here is that inert games, which are in some theoretical sense easy ones, can be arbitrarily complex or bushy. The key to inertness is that the expected situations are varied enough in terms of their number of offsprings. To put it bluntly, a node with many children is likely to be a win whereas a node with few children is likely to be a loss. Moreover, if the statistical structure is inert, the relative frequency of nodes with either 'few' or

\* This means that one has to choose a parameter t (a winning probability) close to unity if m is large.

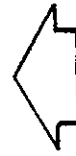
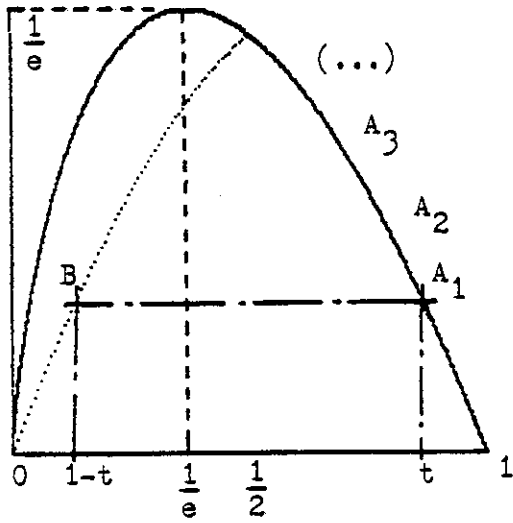


Figure 3.2.a

Since  $t \geq \frac{1}{2}$ , B is in the convex hull of the points  $A_n$ .

Hence, the equation

$$B = \sum k_n A_n$$

has many solutions. Each

corresponds to an internal

structure K s.t.  $\frac{dK}{dz}(\frac{z}{K}) = 1$ .

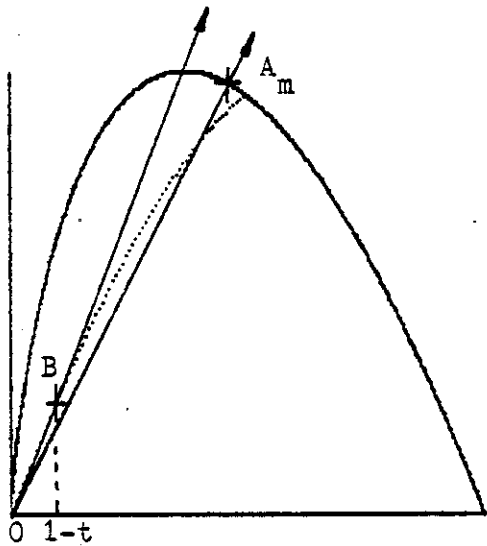


Figure 3.2.b

When  $t > \frac{m}{m+1}$ , B is in the convex hull of the points  $A_n$  for  $n \geq m$ .

$$\text{Slope}(OB) = \frac{-t \ln(t)}{1-t}$$

$$\text{Slope}(OA) = -m \ln(t)$$

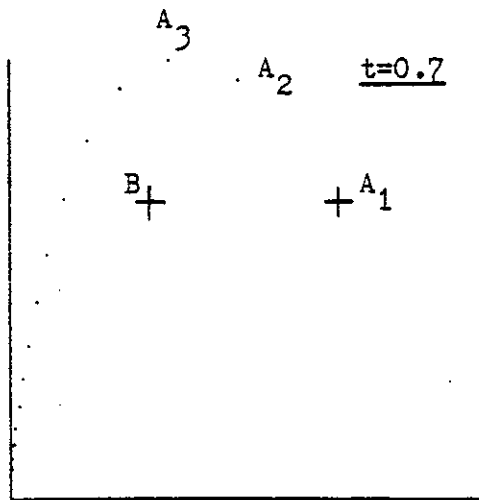
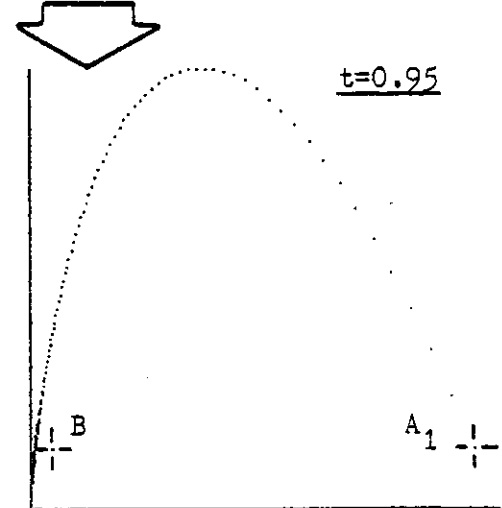


Figure 3.2.c



'many' offsprings actually makes this the dominant effect.

The above discussion was implicitly assuming that the condition  $\frac{dK(\xi_K)}{dz} = 1$  is sufficient to insure inertness. We know (see Appendix A) that this is not quite the case. Actually, some members of the class we just studied are not inert. There is however a very interesting subclass, which is easily proven to contain only inert structures [\*\*]. This class is that of geometric distributions.

The geometric distribution  $K(z) = \frac{z(1-a)}{1-az}$  (where the parameter  $a$  is a nonnegative number strictly less than 1) happens to be a bushy inert statistic in the sense just discussed, i.e.,  $\frac{dK(\xi_K)}{dz} = 1$ , irrespective of the value of  $a$ .

The geometric distribution is quite interesting in both a theoretical sense because it is a bushy inert structure, and a practical sense because most results can be expressed in closed form. [\*] The following is a condensed glossary of the basic features of the geometric distribution.

$$K(z) = \frac{z(1-a)}{1-az} = \frac{1-a}{a} \left[ \frac{1}{1-az} - 1 \right] = \frac{z}{a-(a-1)z}$$

$$k_n = (1-a) a^{n-1}$$

\*\* A direct proof is elementary and is left to the reader.

\* Closed form expressions can even be given for the quantities  $u_n$  and  $v_n$  of section 2.



$$z_K = \frac{1 - \sqrt{1-a}}{a}$$

$$\frac{dK}{dz}(z) = \frac{1-a}{(1-az)^2}$$

$$d = \frac{dK}{dz}(1) = \frac{1}{1-a} \quad (\text{Average internal degree})$$

$$\frac{dK}{dz}(z_K) = 1$$

$$z = 1 - K(1 - K(z)) \quad (\text{There are infinitely many stable two-cycles !})$$

This last property of the geometric distribution is used to build the strange internal structures given in Appendix A. Such strange inert structures yield a shape for the TIE region that departs somewhat from that illustrated in Figure 3.1. In particular, the TIE region may be of a height strictly smaller than unity around  $b=1$  and/or it may include vertical segments of finite length. [\*] More details can be found in Appendix A.

This whole spectrum of possibilities that lies between inert structures, like the geometric distribution, and the typical non inert structures (as illustrated in Figure 3.1) may appear only in some contrived examples for the impartial case we have discussed so far. However, they seem the rule rather than the exception for the partisan games we shall discuss in section 5.

\* When this happens, such vertical segments are to be counted as included in the TIE region. So, for all intents and purposes, we may consider the TIE region as closed and both the WIN and LOSS regions as open (within the unit square minus its rightmost side).

#### 4. Analyzing the complexity of game solving

The first three sections of this paper presented basic ideas and premises. This section will support the claim made in the introduction that many enlightning results are easily obtained using the model presented here. While some of our results were previously obtained in other contexts, giving a valuable hint of their general validity, here we show how easily such results may be obtained. Our model by making basic questions easy to solve, permits more interesting problems to be addressed.

The first problem addressed concerns the time complexity of the basic game solving procedure SOLVE. By solving a game we mean finding whether the first player has a forced WIN, a forced LOSS, or if the game is a TIE. The simple procedure SOLVE described below is intended to compute such a WIN or LOSS status. However, unlike the more general procedure outlined in the introduction, SOLVE cannot terminate when the game is actually a TIE. This is an intrinsic limitation of this model where the possibly finite graph structure (containing cycles) of an infinite tree is totally unknown. There is therefore no way we can assert for sure in a finite number of steps that a game is a TIE. Aside from this limitation, we might actually suspect that SOLVE has a nonzero probability of nontermination well before the transition point (see section 2) where ties occur with nonzero probability. This turns out to be the case and this remark serves as the basis of an estimate of the performance of SOLVE and of other related game solving procedures.

FUNCTION SOLVE (G : Game): Status

IF G is a leaf then RETURN its status

ELSE SOLVE from left to right the options of G until one is found  
which is a LOSS.

IF G has such a LOSS option, RETURN WIN

ELSE RETURN LOSS

ENDFUNC.

The single most important parameter in the analysis of a search procedure like SOLVE is its branching factor namely the expected number of children of a SOLVEd node that would need to be SOLVEd before SOLVE terminates. It is important to note that this number does not assume that SOLVE actually terminates. In fact, the procedure has a nonzero probability of not terminating whenever its branching factor is greater than 1, as implied by Theorem 2.1. The branching factor, by our standards, expresses how many children would need to be expanded on the average. It is understood that all nodes that need to be expanded according to a procedure like SOLVE will all be expanded if and only if the procedure actually terminates.

It turns out that this branching factor can be easily computed directly from the parameters of the basic model.

Let us use the following notations :

$$u = 1-p_L$$

$r$  = The branching factor, the average number of options that need to be examined in order to SOLVE the parent node.

$s_n$  = The expected number of children of a WIN node with  $n$  offsprings that need to be SOLVED in order to SOLVE the parent node. A node is a WIN node with  $n$  offsprings with probability  $bk_n(1-u^n)$ .

Note that the counterpart of  $s_n$  for nodes that are not WINS is always  $n$ , since all options of such a node have to be considered before we can conclude that we do have a no-win situation. The branching factor  $r$  is the weighted average of the expected number of options examined for external nodes where zero options are examined, wins with  $n$  options where  $s_n$  options are expanded, or non-wins with  $n$  options where all  $n$  options are expanded. This translates into:

$$r = b \sum_{n=1}^{\infty} k_n [(1-u^n)s_n + nu^n]$$

A node with  $n$  options is a win with probability  $(1-u^n)$ .

There is a probability of  $u^{k-1}(1-u)$  that a node with  $n$  children will have a leftmost LOSS option as its  $k^{\text{th}}$  offspring. SOLVE will expand  $k$  children of such a node. This yields the following expression for  $s_n$ .

$$(1-u^n)s_n = (1-u) + 2u(1-u) + \dots + nu^{n-1}(1-u)$$

$$= \frac{1-u^{n+1}}{1-u} - (n+1)u^n$$

Hence, we conclude immediately that:

$$r = b \sum_{n=1}^{\infty} k_n \left[ \frac{1-u^{n+1}}{1-u} - u^n \right]$$

$$= b \left[ \frac{1-uK(u)}{1-u} - K(u) \right] = \frac{b(1-K(u))}{1-u}$$

So far, we have not even assumed that we were below the transition point where TIEs have nonzero probability. Making this assumption implies that  $u=1-p_L=p_W=t$ , where  $t$  is the fixed point of  $G_b$  (as discussed at the end of section 2) and, therefore, that  $u = (1-b)p + b(1-K(u))$ , which allows a simplification of the above result.

Theorem 4.1

The branching factor of SOLVE is  $r_{\text{SOLVE}} = \frac{b(1-K(u))}{1-u}$ , where  $u=1-p_L$  is the probability of not having a forced LOSS when playing first. When ties almost never occur,  $u=t=1-p_L=p_W$  and  $r_{\text{SOLVE}} = \frac{t-(1-b)p}{1-t}$ .

As claimed in the beginning of this section, this result is closely related to those obtained for different game models. For example, when the probability of early termination is low (i.e. when  $b$  is close to 1), the above expression clearly becomes  $\frac{t}{1-t}$ . [\*] This expression can be found in [23] (where  $t$  is called  $p^*$ ) as the branching factor of SOLVE.

\* This is also true, in all cases, for normal play ( $p=0$ ), but we consider this as a mere mathematical coincidence. What is really of importance here is the close relation of the result given here and the one quoted. An occasional exact match is not significant given the fact that the models compared are otherwise quite different.

When SOLVE terminates, the model can actually be used to estimate the average number of nodes it expands. The analysis is only slightly more complex than the above but does not seem to bring any new insight and is not given here.

It is of crucial importance, however, to know when SOLVE does terminate. This cruciality may not be apparent if we only concern ourselves with the exact solution to games as we have so far. In section 7, however, we show that the parameter  $b$  is something the programmer has some control over when approximative solutions only are sought. Ultimately, a perfect decision could be expected when the survival rate  $\bar{b}$  used in the decision procedure is the same as the natural survival rate  $(b)$  of the game, as this means that no artificial cutoffs are necessary - a rare situation but a practical assumption for endgame positions. The largest value of  $b$  that almost always ensures the termination of SOLVE (or other game solving procedures) will be used as an indicator of its performance without any further justifications. This is justified at length in section 7.

SOLVE will almost surely (probability 1) terminate if and only if  $r_{\text{SOLVE}}$  is less than 1. Using the expression of  $r_{\text{SOLVE}}$  given in Theorem 4.1, the terminating or 'finiteness' condition is therefore that  $t < \frac{1+(1-b)p}{2}$ . Since  $t$  is the fixed point of the decreasing function  $G_b$  (see section 2), the finiteness condition can be rewritten as  $G_b(\frac{1+(1-b)p}{2}) < \frac{1+(1-b)p}{2}$  or, more explicitly:

$$\frac{1-(1-b)p}{2b} \geq 1 - K(\frac{1+(1-b)p}{2})$$

If we assume the normal play rule, this reduces to:  $b \leq \frac{1}{2(1-K(\frac{1}{2}))}$ .

This last relation can be used to determine the largest  $b$  acceptable to SOLVE. In Figure 4.1.a this maximal value of  $b$  has been plotted as a function of  $d$  when  $K(z)=z^d$ ; note that  $d$  should be an integer. An internal structure  $z^d$  does not allow inertness in games. Therefore, the value of  $b$  at which TIEs first appear, the transition point, has been plotted on the same graph for reference purposes: it represents the best performance any game solving procedure could achieve. The equation of this curve,  $b = \frac{1}{d}(1+d)^{1-\frac{1}{d}}$ , is obtained from Figure 2.4.

For an inert structure, like the geometric distribution  $K(z) = \frac{z}{d-(d-1)z}$ , the conclusions are quite similar, the only difference being that there is no upper bound as to what a game solving procedure can achieve. Figure 4.1.b relates to the geometric distribution. Note that the variable  $d$  is the expected number of options of a nonterminal position in both cases; hence the curves of Figure 4.1.a and Figure 4.1.b are directly comparable.

The graphs show that the maximal value of  $b$  quickly converges toward the limit  $\frac{1}{2}$ . This turns out to be general, whether or not the normal play rule is used, and does not depend on the particular form of  $K$ . As long as we make the average degree of an internal node increase, that maximal value of  $b$  will converge toward a limit  $b_{\text{SOLVE}}(p) = \frac{1-p}{2-p}$ . The value of this limit is obtained by noticing that the quantity  $x = \frac{1+(1-b)p}{2}$  is bounded from above by the ratio  $\frac{1+p}{2}$  and cannot approach 1 (except when  $p=1$ , as discussed below). Therefore, the quantity  $K(x)$

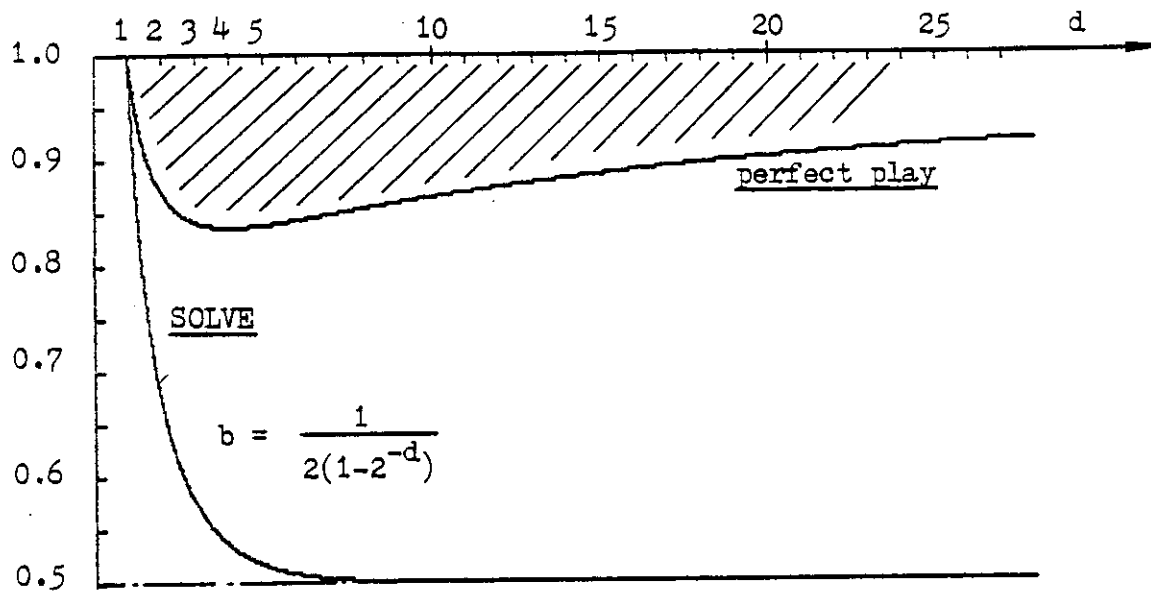


Figure 4.1.a : Performance of SOLVE ( Uniform degree  $d$  )

$$K(z) = z^d, \text{ normal play } (p=0)$$

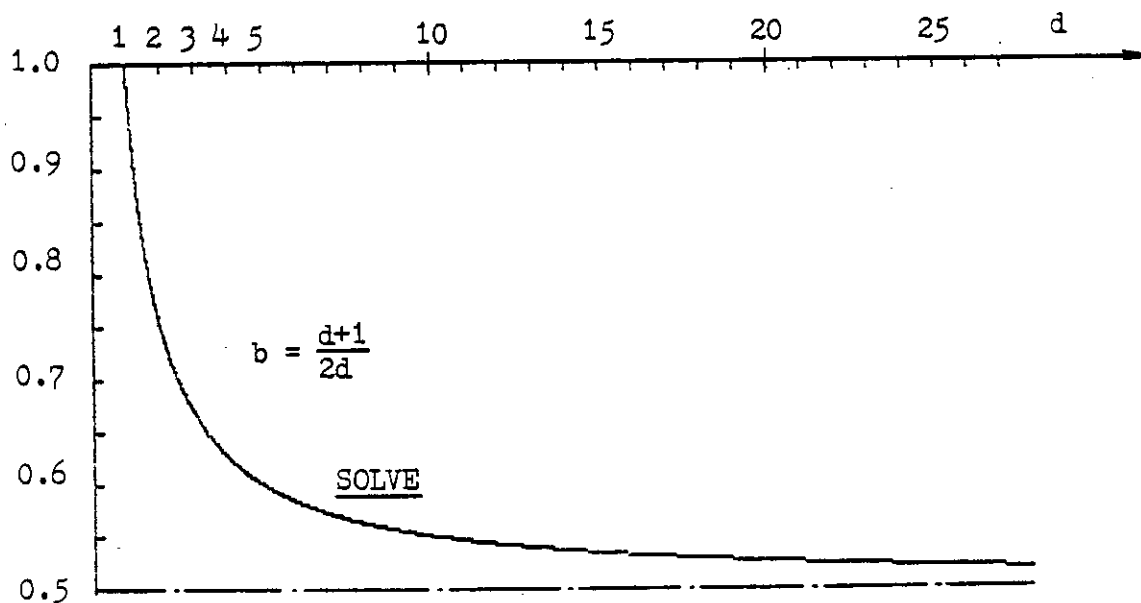


Figure 4.1.b : Performance of SOLVE ( Average degree  $d$ ,  
geometric distribution )

$$K(z) = \frac{z}{d-(d-1)z}, \text{ normal play } (p=0)$$



converges toward zero faster than  $K(\frac{1+p}{2})$  when  $K$  becomes bushy, when the average degree of internal nodes becomes large.

Theorem 4.2

For bushy internal structures  $K$ , the parameter  $b_{\text{SOLVE}}(p)$  below which SOLVE almost surely terminates is given by the expression:

$$b_{\text{SOLVE}}(p) = \frac{1-p}{2-p}$$

When  $p=1$ , the previous argument does not hold but the situation is even easier to analyze directly since the finiteness condition then reduces to  $b(\frac{1}{2} - K(1 - \frac{b}{2})) \leq 0$  and cannot be satisfied for a bushy enough  $K$  unless  $b$  is zero. Therefore the general relation we just gave also holds for  $p=1$  and yields  $b_{\text{SOLVE}}(1) = 0$ .

We may also wish to know whether SOLVE allows some kind of SOLVE-inertness, namely statistics for which  $b$  is almost 1 but for which SOLVE will almost surely terminate. As one suspects, the answer is no, except in the trivial case of string-games. The formal reason is that the finiteness condition then degenerates into  $K(\frac{1}{2}) \geq \frac{1}{2}$ , which is trivially not true of any acceptable function  $K$  except  $K(z)=z$ . A search procedure that handles inert games (in the sense of sections 3 or 5) much better than SOLVE or its variants is discussed in section 8.

The gap that appears in Figure 4.1 between SOLVE and perfect solving can be bridged. To do so, we must provide a way of ensuring a minimal breadth to the search. A foolproof method is to use a modified version of SOLVE that can return the status of any game of length  $M$  or less. We call LSOLVE this depth limited version of SOLVE. A 'Pseudo-

Status' is either WIN,LOSS or UNDEFINED. LSOLVE returns UNDEFINED if and only if the game G is of length greater than M. M is a nonnegative integer.

```
FUNCTION LSOLVE (G : Game, M : Integer): Pseudo-Status
```

```
  IF G is a leaf THEN RETURN its status.
```

```
  IF M is zero THEN RETURN UNDEFINED.
```

```
  ELSE compute from left to right the Pseudo-Status  
    LSOLVE(G',M-1) of each option G' of G until one is  
    found which is a LOSS.
```

```
  IF G has such a LOSS option, RETURN WIN
```

```
  ELSE IF G has an UNDEFINED option, RETURN UNDEFINED
```

```
  ELSE RETURN LOSS
```

```
ENDFUNC.
```

A foolproof game solving procedure is obtained by successively running LSOLVE with a larger and larger M until the result is not UNDEFINED. A similar approach presents certain advantages when applied to game-playing and has been popular in certain commercial chess-playing microprocessor-based machines.

Another alternative that we find interesting is to give up total safety and use LSOLVE as a test within a simple modification of SOLVE that we call  $SOLVE_i$  : SOLVE itself is renamed  $SOLVE_0$  in this context. What  $SOLVE_i$  does is merely a depth-first search similar to that of SOLVE with the additional twist that any node with a game-length of  $i$  or less is always easily solved regardless of the complexity, say, of its left-most option(s).

FUNCTION SOLVE<sub>i</sub> (G : Game): Status

IF LSOLVE(G,i) is not UNDEFINED THEN RETURN this as a  
result. This means that G is a game of length i or  
less.

ELSE SOLVE<sub>i</sub> from left to right the options of G until one is  
found which is a LOSS.

IF G has such a LOSS option, RETURN WIN

ELSE RETURN LOSS

ENDFUNC.

SOLVE<sub>i</sub> is only the simplest of the procedures based on this idea, not the most efficient. In particular, a given node can be visited many times during different auxiliary searches (LSOLVE). It is possible to reduce this waste dramatically through nontrivial bookkeeping if space limitations are not a problem. Such improved versions of SOLVE<sub>i</sub> will, however, terminate if and only if SOLVE<sub>i</sub> does. This makes it interesting to extend to SOLVE<sub>i</sub> the analysis given above for SOLVE<sub>0</sub>. This is summarized below using the same notations. Note that even though SOLVE<sub>i</sub> expands in its main loop less nodes than SOLVE<sub>0</sub>, such expansions cost more in terms of either time (for plain SOLVE<sub>i</sub>) and/or space (for improved versions).

$$r = (u_i - v_i) \sum_{n=1}^{\infty} k_n [(1-u)s_n + nu^n]$$

where  $u_i$  and  $v_i$  are the quantities defined in Theorem 2.2. Besides, we still have:

$$(1-u^n)s_n = (1-u) + 2u(1-u) + \dots + nu^{n-1}(1-u)$$

$$= \frac{1-u^{n+1}}{1-u} - (n+1)u^n$$

$$r = (u_i - v_i) \sum_{n=1}^{\infty} k_n \left[ \frac{1-u^{n+1}}{1-u} - u^n \right]$$

$$= (u_i - v_i) \frac{1-K(u)}{1-u}$$

Theorem 4.3

The branching factor of  $SOLVE_i$  is  $r_{SOLVE_i} = B_i \frac{b(1-K(u))}{1-u}$ . When ties almost never occur, this expression is equal to the quantity  $B_i \frac{t-(1-b)p}{1-t}$ , where  $B_i = \frac{1}{b}(u_i - v_i)$  is a decreasing function of  $i$  whose value is 1 for  $i=0$  and whose limit is  $\frac{PT}{b}$  (zero when ties have probability zero).

$SOLVE_i$  almost always terminates when  $r_{SOLVE_i}$  is less than 1. An argument similar to that used for  $SOLVE_0$  shows that this reduces to the following finiteness condition:

$$\frac{1-(1-b)p}{b(1+B_i)} \geq 1-K\left(\frac{1+B_i(1-b)p}{1+B_i}\right)$$

Using the two inequalities:

$$\frac{1-(1-b)p}{b(1+B_i)} \geq \frac{1}{1+B_i} \quad \text{and} \quad 1-K\left(\frac{1}{1+B_i}\right) \geq 1-K\left(\frac{1+B_i(1-b)p}{1+B_i}\right)$$

we see that the finiteness condition is certainly satisfied when

$\frac{1}{1+B_i} \geq 1-K(\frac{1}{1+B_i})$ . This condition is sufficient and is 'almost' necessary when  $b$  is close to unity, the interesting case. At any rate, this means that SOLVE<sub>*i*</sub> almost always terminates when  $\frac{1}{1+B_i}$  is less than or equal to the fixed point  $\hat{\epsilon}_K$  of  $1-K$ .

Theorem 4.4

SOLVE<sub>*i*</sub> almost always terminates when the fraction  $B_i$  of the non-terminal nodes whose length is greater than  $i$  is less than the quantity  $[1/\hat{\epsilon}_K - 1]$ .

For example, if  $K$  is the geometric distribution as described at the end of section 3, the quantity  $[1/\hat{\epsilon}_K - 1]$  is equal to  $\frac{1}{a+\sqrt{1-a}}$  (i.e.  $\frac{1}{\sqrt{a}}$ ). As a result, the condition given in Theorem 4.4 is never satisfied for SOLVE<sub>0</sub> [\*] but is fairly often satisfied even for lowly SOLVE<sub>1</sub>. Under normal play ( $p=0$ ),  $B_1$  is equal to  $\frac{b(1-a)}{1-ab}$ ; hence the condition of Theorem 4.4 is satisfied as soon as  $b$  is less than  $\frac{1}{a+\sqrt{1-a}}$ . This quantity is close to 1 when  $a$  is close to either its minimum or its maximum 0 and 1 respectively. This means that SOLVE<sub>1</sub> is efficient for either sparse or bushy games. For the intermediate cases, SOLVE<sub>1</sub> does not perform too poorly either since the minimal value of  $\frac{1}{a+\sqrt{1-a}}$  is as high as 4/5. This minimum is reached for  $a=3/4$ , that is  $d=4$ . This means that SOLVE<sub>1</sub> almost always terminate when 20% or more of the nodes are terminal.

\* This is always the case, regardless of the structure  $K$ , as  $B_0$  is always equal to 1 while  $[1/\hat{\epsilon}_K - 1]$  is strictly less than 1 for all structures  $K$ , (with the exception of string-games -  $K(z)=z$ ).

Similarly,  $B_2 = \frac{b^2(1-a)^2}{(1-ab)(1-2ab+ab^2)}$ , and SOLVE<sub>2</sub> almost always terminate when 13.7% of the nodes are terminal (for d=4).

We conclude this section by an investigation of the conditions in which SOLVE<sub>i</sub> terminates when the statistics are matched ( $p = t = \xi_K$ ). What follows is restricted to the geometric distribution ( $K(z) = \frac{z}{d-(d-1)z}$ ).

The matched winning probability is  $\xi_K = \frac{d-\sqrt{d}}{d-1} = 1 - \frac{1}{\sqrt{d}} + O(\frac{1}{d})$ . For this value of p, we have  $t=p$ . The expression of the branching factor of SOLVE<sub>i</sub> given in Theorem 4.3 reduces to  $B_i \frac{bp}{(1-p)} = (u_i - v_i) \sqrt{d}$ . SOLVE<sub>i</sub> almost surely terminates when this quantity is less than or equal to 1. For  $i=0$  this means that b must be less than or equal to  $\frac{1}{\sqrt{d}}$ ; for larger values of i the largest acceptable b can be plotted as a function of d (Figure 4.2).

Surprisingly enough, the largest acceptable value of b converges (slowly) toward a strictly positive limit as d goes to infinity. This limit is a solution of the equation:

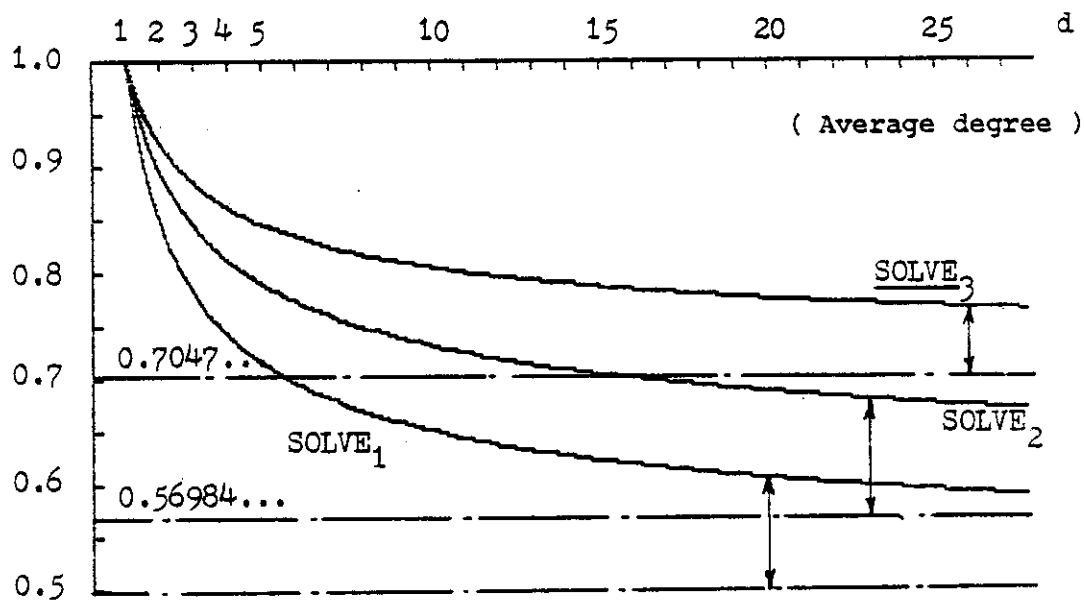
$$b^i = \frac{1-(-b)^{i+1}}{1+b} \cdot \frac{1-(-b)^i}{1+b}$$

This is a consequence of the fact that, for  $i>0$ , we can prove that:

$$u_i = 1 - \frac{x_j}{\sqrt{d}} + O(\frac{1}{d})$$

Figure 4.2

Performance of SOLVE<sub>1</sub> , SOLVE<sub>2</sub> and SOLVE<sub>3</sub>  
(Matched statistics,  $p = \frac{e}{S_K}$ , geometric distribution )



$$v_i = 1 - \frac{x_k}{\sqrt[i]{d}} + O\left(\frac{1}{d}\right)$$

Where  $x_n = \frac{1 - (-b)^{n+1}}{1 - (-b)}$  and  $j = 1 + 2 \lfloor \frac{i}{2} \rfloor$  and  $k = 2 \lceil \frac{i}{2} \rceil$ . Therefore:

$$(u_i - v_i) \sqrt[i]{d} = |x_{i+1} - x_i| + O\left(\frac{1}{\sqrt[i]{d}}\right)$$

This means that  $(u_i - v_i) \sqrt[i]{d}$  is equal to  $1 + O\left(\frac{1}{\sqrt[i]{d}}\right)$  when  $|x_{i+1} - x_i| = 1$ .

This relation can be rewritten as the equation given above. The corresponding values of  $b$  for various values of  $i$  are tabulated below. For example, the table shows that if 19.5% or more of the nodes are terminal, (or 'quiescent', as discussed in section 7), then the procedure SOLVE<sub>6</sub>, which searches 3 full moves ahead before going depth first, will almost surely terminate. In practice, the situation is better than what the table suggests, as the values given are only pessimistic estimates of the largest acceptable  $b$  that are accurate only when the degree  $d$  is unrealistically large (see Figure 4.2).



Figure 4.3

The highest proportion of nonquiescent nodes that make SOLVE<sub>i</sub> terminate for any geometric internal structure.

i=	b= (%)
0	0
1	50
2	56.984
3	70.471
4	73.222
5	79.047
6	80.508
10	87.359
20	93.090
30	95.408

## 5. Partisan games

When the same moves are legal for either player, a game is called impartial. A partisan game, on the contrary, is a game in which the two players may have different options. The rules of such games specify who is allowed to make which move from a given position. A typical partisan game is chess. Each player owns certain pieces and can move only those. His opponent's moves are totally distinct from his own. Partisan games typically allow one to gain a certain advantage in terms of the number of moves one could skip and still be the winner. [\*] This is totally unknown in the world of impartial games where skipping a move always turns a winner into a loser.

One could say that whether a game is partisan or impartial is only a question of how each position is described. Indeed, adding a bit (to indicate whose turn it is to play) to the description of each position of a partisan game would turn it formally into an impartial game. This turns out to be a poor idea. This extra bit of information is artificial and leads to at least two problems. First, several such artificially impartial games are not played simultaneously [10] the way their partisan counterparts are. This is important whenever a game can break down into the disjunctive sum of smaller games. Second, and more relevant to this context, a partisan game may exhibit a statistical asymmetry between the two players that cannot be represented by any impartial model.

---

\* A more precise formulation of this 'advantage' [10,5] leads one to consider numbers of moves that are not integers. This is a natural concept when considering (disjunctive) sums of games.

If anything, partisan games are much more common than impartial ones and this is the primary motivation for this section. Some partisan games may indeed be impartial in a statistical sense but many are not. Usually, partisan games that are statistically impartial in certain even positions, like chess at its beginning, quickly evolve into asymmetrical situations.

Our model is readily modified to account for this kind of basic dissymmetry. We define partisan games as two classes of games, one for each player. The definitions of the two classes are mutually recursive.

Definition 5.1

A partisan recursive random game consists in the pair  $(G, G')$ , where  $G$  and  $G'$  are defined by the following formulas :

$$G = f_0 (p \langle W \rangle + q \langle L \rangle) + f_1 \begin{array}{c} G \\ | \\ G' \end{array} + f_2 \begin{array}{c} /0 \backslash \\ G' \ G' \end{array} + \dots$$

$$G' = f'_0 (p' \langle W \rangle + q' \langle L \rangle) + f'_1 \begin{array}{c} G \\ | \\ G \end{array} + f'_2 \begin{array}{c} /0 \backslash \\ G \ G \end{array} + \dots$$

The player who is called upon to play from a member of  $G$  is referred to as player A. The other player is called A'.

In addition to the above, notations similar to that introduced in the impartial case are used. Those are summarized below so as to avoid

any ambiguity as to which quantity belongs to which player.

Definition 5.2

$$\begin{aligned}
 F(z) &= \sum_{n=0}^{\infty} f_n z^n & F'(z) &= \sum_{n=0}^{\infty} f'_n z^n \\
 &= (1-b) + bK(z) & &= (1-b') + b'K'(z) \\
 G(z) &= fp + 1 - F(z) & G'(z) &= f'p' + 1 - F'(z) \\
 &= (1-b)p + b(1-K(z)) & &= (1-b')p' + b'(1-K'(z)) \\
 r &= \frac{dF}{dz}(1) & r' &= \frac{dF'}{dz}(1)
 \end{aligned}$$

The development is similar to that of the impartial case, with the added twist that most quantities now come in two flavors. The probabilities  $u_n$  [resp.  $v_n$ ] that player A does not have a forced loss [resp. has a forced win] in  $n$  moves or less is now related to the similar quantities for player A', as follows:

$$\begin{aligned}
 \begin{cases} u_0 = G(0) \\ v_0 = G(1) \end{cases} & \quad \begin{cases} u_{n+1} = G(v'_n) \\ v_{n+1} = G(u'_n) \end{cases} \\
 \\ \\
 \begin{cases} u'_0 = G'(0) \\ v'_0 = G'(1) \end{cases} & \quad \begin{cases} u'_{n+1} = G'(v_n) \\ v'_{n+1} = G'(u_n) \end{cases}
 \end{aligned}$$

This is best expressed graphically (see Figure 5.1.a) by introducing the two curves of equations  $x=G(y)$  and  $y=G'(x)$ . Four sets of stairs are drawn using those two curves. [\*] Ties may or may not have zero probability (see Figures 5.1.a and 5.1.b). The graphical interpretation makes it quite clear when they do.

\* The four possible origins of the stairs are the points  $(1, v'_0)$ ,  $(0, u'_0)$ ,  $(v_0, 1)$  and  $(u_0, 0)$ . Only the stairs corresponding to  $(v_0, 1)$  are drawn on Figure 5.1.a and 5.1.b. They yield  $v_n$  for even indices and  $u'_n$  for odd indices.

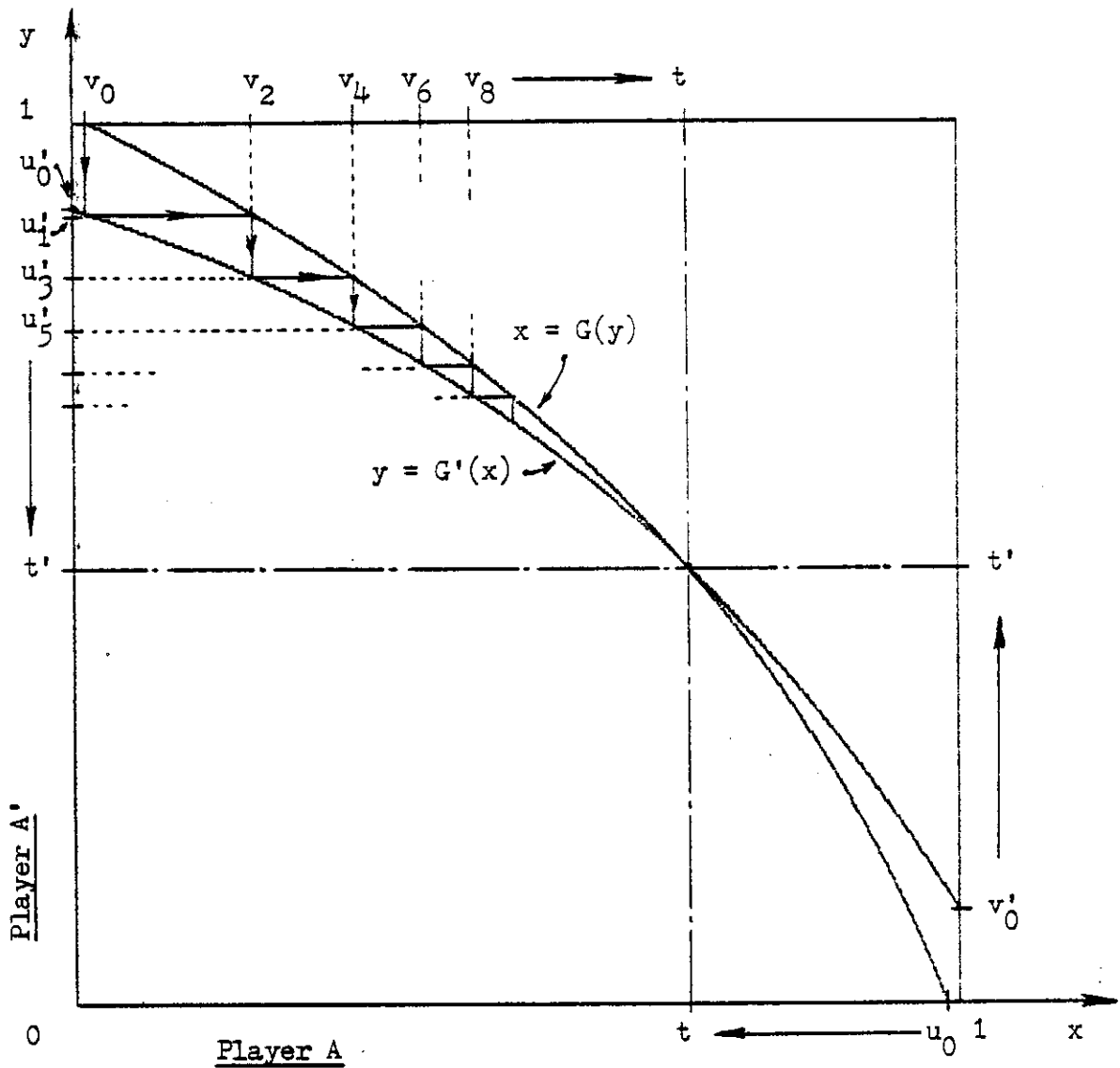


Figure 5.1.a : A partisan game where ties almost never occur.

$v_n$  (resp.  $v'_n$ ) is the probability that player A (resp. player A') can win in  $n$  moves or less when playing first.

$u_n$  (resp.  $u'_n$ ) is the probability that player A (resp. player A') can avoid losing in  $n$  moves or less when playing first.

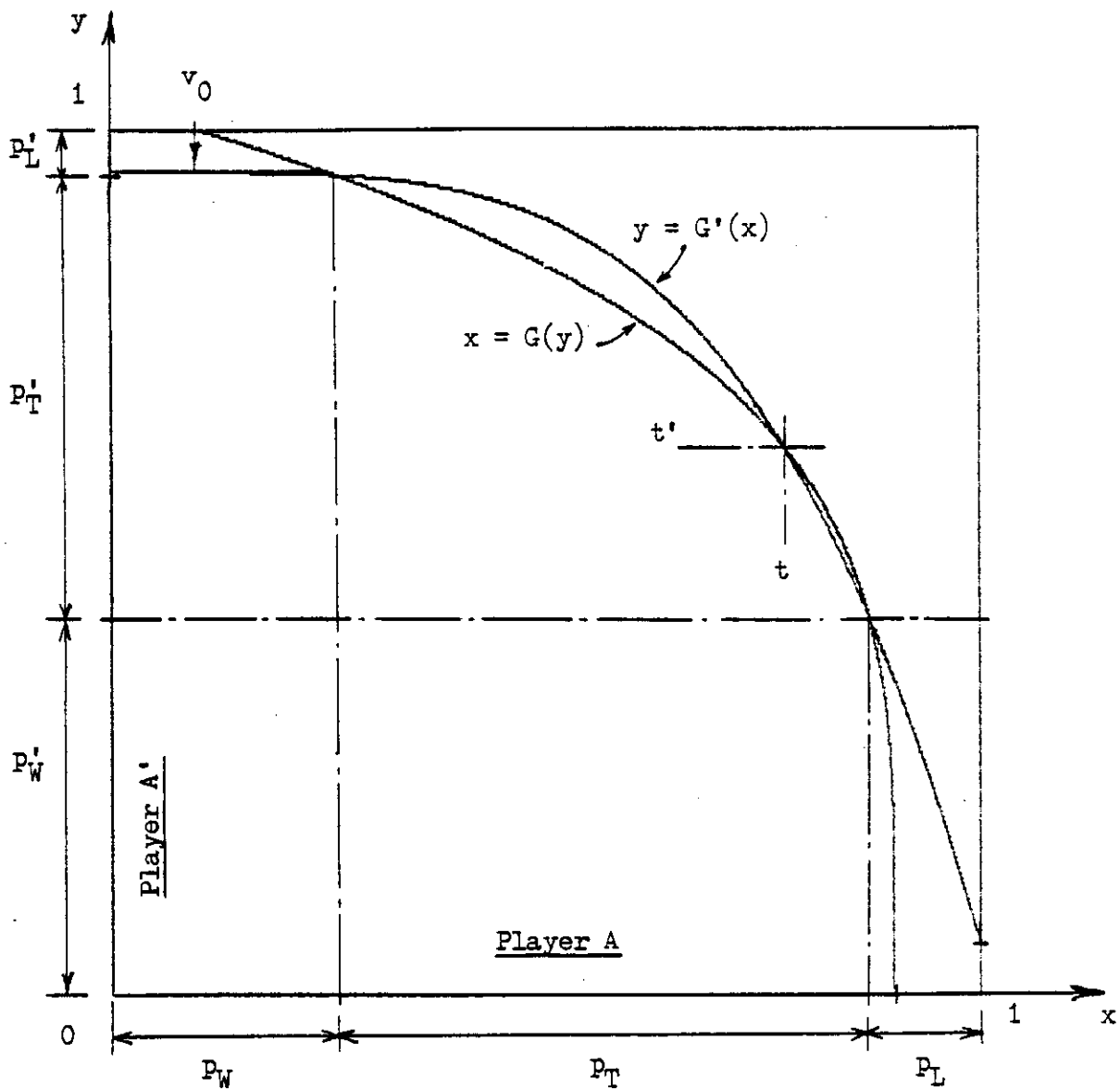


Figure 5.1.b : A partisan game where ties often occur.

$P_L, P_T, P_W$  (resp.  $P_L', P_T', P_W'$ ) are the probabilities that player A (resp. player A') will be given a LOSS, a TIE or a WIN position when playing first.

### Theorem 5.1

Ties almost never occur if and only if the two curves of equations  $x = G(y)$  and  $y = G'(x)$  have only one point in common.

Figure 5.1.c illustrates another point which can occur in the impartial case as well (see Appendix A), but is particularly dramatic in the partisan case. The parameters that produced Figure 5.1.c are only slightly different from those used in Figure 5.1.b. The slight difference, however caused the two extra intersections to disappear. The result is that ties can no longer occur. What were previously ties are now WINS for player A'. However, the wins for A' are clearly separated into two categories. The quick ones (in about 6 moves or less) and the tedious ones (in more than 25 moves). The probability of games that would last between 6 and 25 moves is small. Besides, the probability that A could win a long game is patently ridiculous. In fact, the only games that A is somewhat likely to win are those that last less than 3 or 4 moves.

We can also note, in passing, that a similar graphical interpretation holds true to decide whether or not the underlying tree structures are finite. The tree structure is therefore almost surely finite if and only if the two curves of equations  $x = F(y)$  and  $y = F'(x)$  have only one point in common, namely the point (1,1). Now, we observe that  $[x = F(y)]$  lies below its tangent (of slope  $\frac{1}{F}$ ) at point (1,1) while  $[y = F'(x)]$  lies above its tangent (of slope  $r'$ ) at point (1,1). Therefore, for the tree to be finite it is necessary and sufficient that  $r'$  be less than or

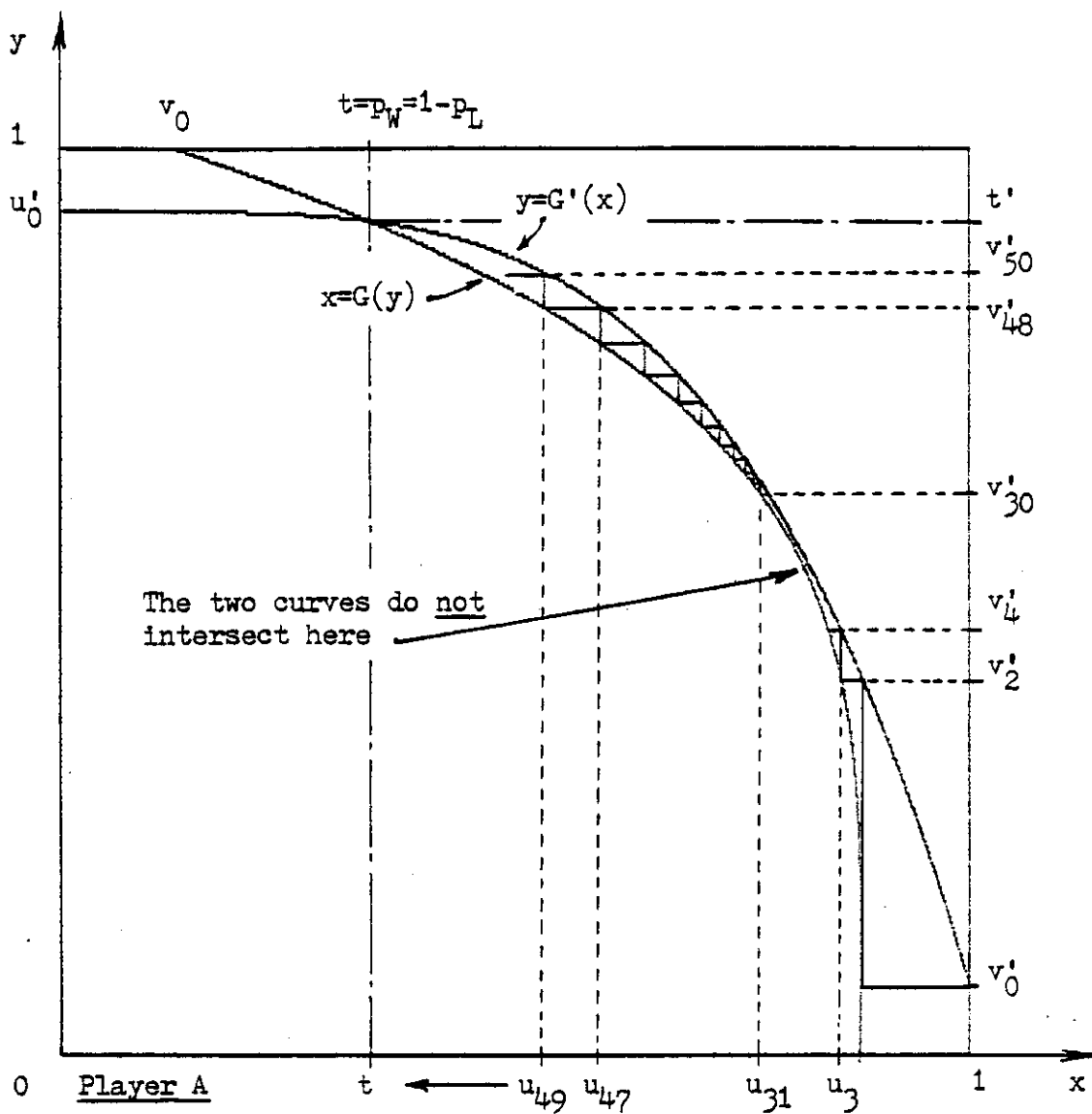


Figure 5.1.c : A minute variation just turned all ties into winning positions (in the long run) for player A'.



equal to  $\frac{1}{r}$ . This can be stated as a partisan counterpart of Theorem 2.1.

Theorem 5.2

A partisan tree structure is almost surely finite if and only if  $r r' \leq 1$ . Where  $r = \frac{dF}{dz}(1)$  and  $r' = \frac{dF'}{dz}(1)$ .

We can further remark that the expected number  $Z_n$  of nodes at depth  $n$  is simply a product of  $n$  terms alternately equal to  $r$  and  $r'$ . The natural definition of the branching factor is as the limit of the quantity  $(Z_n)^{\frac{1}{n}}$ .

Definition 5.3

The branching factor of a partisan tree structure is  $\sqrt[r]{rr'}$ .

It turns out that most of the discussion made in the impartial case is still valid. Among the exceptions, one should mention the stability we observed in section 3 (see Figure 3.1) for impartial games. For partisan games, the quantity  $p_W$  (resp.  $p'_W$ ) is no longer guaranteed to be between  $p$  and  $\xi_K$  (resp. between  $p'$  and  $\xi_{K'}$ ).

Inertness, is defined essentially like it was in the impartial case.

#### Definition 5.4

A partisan internal structure  $(K, K')$  is inert if ties almost never occur in any probability distribution of internal structure  $(K, K')$  and of external structure  $((\xi_K, \xi_{K'}), (b, b'))$  for any quantities  $\xi_K$  and  $\xi_{K'}$  strictly between 0 and 1 satisfying [\*]

$$\begin{cases} \xi_K = 1 - K(\xi_{K'}) \\ \xi_{K'} = 1 - K'(\xi_K) \end{cases}$$

Conditions for inertness are presented in Appendix B in such a partisan context, encompassing the impartial case as well.

The ideas of this section could be extended by defining more than two classes of games at once. Such an extension could provide a better fit for the observed statistical behavior of a given practical game and/or a more accurate heuristical guideline for actual play. The number of types of games one defines need not be finite or even countable. This kind of statistics is sometimes [1] called "general Markov branching processes". However, indulging in such refinements would soon distract us from our primary goal which is to provide a simple model. Furthermore, if any practical use is ever to be made directly from the predictions of the model (see section 7) we may have to estimate the model's parameters with a relatively small sample. A general Markov branching process model simply does not enable us to do this unless the number of types is rather limited. Even then, we need a good way of distinguishing between types a priori. This may not be so difficult to do in practice

\* Note that, unlike what happened in the impartial case, there may be more than one possibility for  $\xi_K$  and  $\xi_{K'}$ . This may even be true if  $K = K'$ , as shown in Appendix A.

as we can rely on common sense. For example, a capture in chess is likely to be interpreted as a type change. Hence, a coarse evaluation function based, say, only on the count of the pieces, could be used to define the a priori type of a node. It could then be practical to have 20 or more types of nodes.

## 6. Error propagation and pathology analysis

This section will address a question whose answer was taken for granted until very recently. It has always been assumed that a deeper search would naturally lead to a better decision. This seems to be the behavior of procedures that deal with real games. However, Nau [20] showed that a deeper search of a game tree could actually lead to a poorer decision. This observation was made using the earlier uniform degree and constant height model. This phenomenon is now known as game pathology. As of this writing, it is still not clear how generally widespread pathology is among real games. Common sense would seem to indicate that it is an artifact of our models for either the structures of games or the heuristics we use on them. A discussion of the probable causes for pathology appears in [24] but the problem is still widely open. Pathology, however, seems restricted to the minimax rule for using one's static estimate of a situation. It does not appear for other rules like the one discussed in section 8.

This section describes a simple-minded investigation of the quality of a decision based on the estimate of positions farther down in the game tree. We give strong evidence of the fact that games with a uniform number of legal options are bound to be pathological. By contrast, we show that games that obey the geometric distribution are not pathological. Finally, we characterize the optimal type of estimates to be used in the kind of search described below inspired from commonly used algorithms. We reach the paradoxical conclusion that it sometimes pays off to make some mistakes on purpose...

We make the following assumptions.

- Estimates are used only for nodes at a given level  $n$ . This is not highly recommended in practice, as suggested in section 7, but is consistent with the conditions in which pathology was first observed.
- Ties almost never occur. This is consistent with Nau's model.
- Terminal positions do not appear at or before depth  $n$ . Again this is consistent with Nau's original assumptions.
- The external and internal statistics are matched. This enables most of our results to be used even though terminal nodes appear according to a rule that may have little to do with our general scheme, as discussed in section 3. This is an easy way to ensure consistency with the previous assumption. Again this does not depart significantly from the assumptions of Nau or Pearl.
- The estimates used are boolean (WIN/LOSS) estimates. [\*]

To summarize, the main difference between our investigation and those appearing in [24,20] is that we allow a position to have a variable number of options.

---

\* This assumption is rarely, if ever, practical. However, it has been shown [23] that when more refined estimates are used and reduces to SOLVE-like procedures. A procedure called SCOUT that does this explicitly (and efficiently) has been introduced in [23]

Given the depth  $n$  introduced in the first condition above, we introduce the four following joint probabilities.

$W_n$  is the probability that a node is a WIN and is estimated as such.

$L_n$  is the probability that a node is a LOSS and is estimated as such.

$\bar{W}_n$  is the probability that a node is a WIN but is estimated as a LOSS.

$\bar{L}_n$  is the probability that a node is a LOSS but is estimated as a WIN.

The quality of our initial, static heuristic is of course given by the value of those four parameters for  $n=0$ . Now, we notice that the estimated status and the true status propagate according to the same rules but independently of each other. That means, for example, that a node is a WIN estimated as a LOSS if and only if at least one of its children is really a LOSS but all of them are estimated as WINS. The probability that this will happen 'at level  $n+1$ ' for a node with  $i$  options is therefore:

$$\bar{L}_n(W_n + \bar{L}_n)^{i-1} + W_n \bar{L}_n(W_n + \bar{L}_n)^{i-2} + \dots + (W_n)^{i-1} \bar{L}_n = (W_n + \bar{L}_n)^i - (W_n)^i$$

Averaging this over all possible number of options, a node having  $i$  options with probability  $k_i$ , yields:

$$\bar{W}_{n+1} = K(W_n + \bar{L}_n) - K(W_n)$$

Similarly,

$$\bar{L}_{n+1} = K(W_n + \bar{W}_n) - K(W_n)$$

We also have the two relations:

$$L_{n+1} = K(W_n)$$

$$W_{n+1} = 1 + K(W_n) - K(W_n + \bar{W}_n) - K(W_n + \bar{L}_n)$$

This last relation was obtained courtesy of the equation  $1 = W_{n+1} + \bar{W}_{n+1} + \bar{L}_{n+1} + L_{n+1}$ . Note that a node could well be correctly estimated as a WIN just because one of its options has been estimated as a LOSS while another of its option really is a LOSS.

For convenience, we will visualize the four quantities just introduced as a 2x2 matrix.

$$M_n = \begin{bmatrix} W_n & \bar{W}_n \\ \bar{L}_n & L_n \end{bmatrix}$$

A diagonal matrix represents a perfect estimation. The nondiagonal terms each represent the probability of making a specific kind of mistake. The four elements add up to 1. Because we assumed matched statistics, the sum of the first row which represents the probability that a node really is a WIN is  $t = \sum_k$ ; and the sum of the second row is  $(1-t)$ . Therefore:

$$K(W_n + \bar{W}_n) = K(t) = 1-t$$

So, the above expression of  $\bar{L}_n$  can be greatly simplified.

$$\text{If } M_n = \begin{bmatrix} x_n & t-x_n \\ y_n-x_n & 1-t+x_n-y_n \end{bmatrix} \quad \text{then } M_{n+1} = \begin{bmatrix} t-K(y_n)+K(x_n) & K(y_n)-K(x_n) \\ 1-t-K(x_n) & K(x_n) \end{bmatrix}$$

In other words  $y_{n+1} = 1-K(y_n)$ . In the uniform case  $K(z) = z^d$  that Nau focused on, we obtain the equation  $y_{n+1} = 1-(y_n)^d$  that Pearl [23] discussed at length. That means that  $y_n$  will eventually oscillate between the two values 0 and 1. Since  $x_n$  must be zero when  $y_n$  is zero, this means that the matrix  $M_n$  will oscillate between the two values:

$$\begin{bmatrix} 0 & t \\ 0 & 1-t \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} t & 0 \\ 1-t & 0 \end{bmatrix}$$

Which of those two values corresponds to even values of  $n$  depends on whether  $y_0$  was initially above or below the fixed point  $t = \frac{t}{K}$ . This type of behavior is common to other type of internal structures as well, namely any typical non-inert structure, that is one that does not have any stable two-cycles besides (0,1). [\*] For such an internal structure, a matrix  $M_0$  that is even slightly nondiagonal will quickly degenerate into the above two-cycle. Even an excellent static evaluation of a node's status will be rapidly spoiled, a phenomenon which Nau [20] calls pathological.

If there are other stable two-cycles however, the situation may look quite different. The only stable two-cycle, for a typical inert structure, reduces to the fixed point  $t$  itself. In general, however, the

\* If one considers non-inert structures like the ones described in Appendix A, the general rule is that  $y_n$  will converge toward  $t$  if and only if  $t$  is a stable fixed point and the initial value  $y_0$  lies within the innermost (unstable) two-cycle of  $1-K$ . If this is not the case,  $y_n$  will eventually be attracted either by the innermost stable two-cycle or by one of the two stable two-cycles that surround  $y_0$ . One of the interesting implications of this is that there are internal structures for which the probability of error converges toward zero if and only if the initial heuristic was above a certain level of quality.



matrix  $M_n$  may now oscillate ultimately between the two values

$$\begin{bmatrix} x & t-x \\ 0 & 1-t \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} t & 0 \\ 1-t-K(x) & K(x) \end{bmatrix}$$

where  $x$  is any stable solution of  $x = 1-K(1-K(x))$ . The extreme example of such a behavior is that of the geometric distribution for which  $x = 1-K(1-K(x))$  for any value of  $x$  ! This very special property of the geometric distribution enables us to compute easily what value of  $x$  is reached in the final two-cycle from the initial matrix  $M_0$ . We can also compute the parity of the final cycle, that is, whether the first of the above matrices corresponds to even or odd values of  $n$ . The argument is based on the fact that, for the geometric distribution,  $y_{n+2} = y_n$ .

If the first of the above matrices corresponds to even values of  $n$ , this means that  $x$ , the sum of its first column, is equal to  $y_0$ , the sum of the first column of  $M_0$ . Since  $t-x$  must be positive, this means that  $y_0$  was originally less than or equal to  $t$ . On the other hand, if even values of  $n$  correspond to the second matrix,  $y_0$  is equal to  $1-K(x)$ . Since  $1-t-K(x)$  must be positive,  $y_0$  was greater than or equal to  $t$ . In either case, the sum of the two nondiagonal terms, which represents the probability of error will be:

$$|y_0 - t| \quad \text{For even (large) values of } n.$$

$$|1-K(y_0) - t| \quad \text{For odd (large) values of } n.$$

This type of behavior is typically nonpathological. If the above quantities are smaller than the initial error term  $t-2x_0+y_0$ , look-ahead

is beneficial. The deeper that we can afford to search, the better. Now comes the real surprise. The above two quantities become equal to zero if and only if  $y_0 = t$ . If this is not the case, even a very deep search will yield a nonzero probability of error. The meaning here is rather surprising. If one wants to make the best use of a deep search, then the accuracy of one's estimates are only of secondary importance. It is more important is to make sure that the two possible kinds of error are equally likely. That is, that there should be as many WIN nodes that we consider to be LOSSES (probability  $t-x_0$ ) as there are LOSS nodes that we estimate as WINS (probability  $y_0-x_0$ ). In other words, one should not use the 'best' estimate available, but should make some mistakes on purpose so as to ensure the above balance!

Most inert structures do not have a dense population of stable two-cycles in the neighborhood of  $x=t$ , as does the geometric distribution. Therefore, the limit of  $y_n$  will usually be  $t$  regardless of the initial value of  $y_0$ . This means that estimates derived from a deep search of an inert structure will tend to be perfect regardless of how poor the initial static evaluation function with which we started. However, the geometric distribution is not that forgiving but appears as one of the less forgiving inert structures. This is one of the reasons that make it especially interesting for the purpose of evaluating search procedures.

At this point, the balance condition  $W_0 = E_0$ , apart from being rather paradoxical may appear somewhat difficult to satisfy in practice. The reason is that, obviously,  $E_0$  cannot be more than the probability of

a next player LOSS. [\*] For reasonably bushy games, most nodes are WINS, that is  $t$  is close to 1, and the condition  $w_0 = \lfloor_0$  therefore implies that the conditional probability that a node is estimated as a LOSS, given it is a WIN, is less than  $\frac{1-t}{t}$  and is therefore pretty close to zero. The next section will show that this problem is taken care of to some extent by some search schemes. However, even though such schemes will eventually correct an initial imbalance, this correction is expensive in term of the extra running time needed to achieve a certain quality of decision. In the remainder of this section we show that it is indeed practical to build up a balanced static evaluation function if the statistical features of the game tree are well known. Our example is, again, that of the geometric distribution.

The only static feature of a node that is naturally accounted for in our model is the number  $n$  of its legal options. Furthermore, the larger the number of legal options, the greater the probability that a node is a WIN. Therefore, the only natural static evaluation functions in our model consist in estimating that a node with  $n$  offsprings is:

A LOSS if  $n$  is less than  $N$

A WIN if  $n$  is more than  $N$

A LOSS with probability  $(1-c)$  and a WIN with probability  $c$  if  $n$  is equal to  $N$ .

Note that if  $n$  is equal to  $N$ , the static evaluation is obtained strictly by flipping a coin. This coin need not be a fair one.

\* Recall the quantities we are dealing with are joint probabilities, not conditional probabilities.

We will now study which values of  $N$  and  $c$  are optimal. Since a position with  $n$  legal options is a LOSS with probability  $t^n$  and a WIN with probability  $1-t^n$ , the optimal purely static evaluation function is clearly to use  $c=0$  and  $N = \left\lfloor \frac{\text{Log}(1/2)}{\text{Log}(t)} \right\rfloor$ , since this is what gives a node its more likely status. Such an approach is only appropriate for a shallow search. For a deep search we just saw that the condition to be respected is that the two probabilities of errors  $\overline{W}_0$  and  $\overline{L}_0$  be nearly equal.

$$\overline{W}_0 = k_N(1-c)(1-t^N) + \sum_{i=1}^{N-1} k_i(1-t^i)$$

$$\overline{L}_0 = k_N c t^N + \sum_{i=N+1}^{\infty} k_i t^i$$

Using the fact that  $K(t)=1-t$ , the condition  $\overline{W}_0 = \overline{L}_0$  reduces to:

$$1-t = c k_N + \sum_{i=1}^N k_i$$

In the geometric case, this means that:

$$t = a^N - c(1-a)a^{N-1} \quad \text{where} \quad t = \frac{1-\sqrt{1-a}}{a}$$

This condition is fairly easy to satisfy. We do not even need to flip a coin ( $c=0$ ) for the values of  $d = \frac{1}{1-a}$  tabulated in Figure 6.1. Values of  $d$  between those tabulated imply some coin flipping to achieve a perfect balance. For comparison the second column of Figure 6.1 shows the optimal value of  $N$  under the maximum likelihood shallow search criterion.

Figure 6.1

Optimal Static Evaluation  
Functions for the Geometric Distribution

Deep Search N=	Shallow Search N=	Average Degree d=
1	1	2.618
2	2	6.595
3	2	12.589
4	3	20.587
5	4	30.586
6	4	42.585
7	5	56.585
8	6	72.584
9	6	90.584
10	7	110.584
20	14	420.583
30	21	930.583

## 7. Quiescence analysis

It is rarely practical to solve a game exactly. Most of the time we will have to perform a truncated search and base our decision on it. This causes two problems. First, one has to decide when to stop the search. Second, one must design a way of evaluating a node statically once it has been decided that the search should be stopped. More emphasis is usually placed on the second aspect, yet the first aspect, growth control, is at least equally important, if not more so.

The static evaluation function is traditionally highly problem-dependent and is often used to embody most, if not all, of whatever problem-specific knowledge is available. Since static evaluation functions are often highly sophisticated, our basic model can only account for the most primitive of them. Coarse evaluation functions based only on the number of offspring have been discussed in the previous section.

The number of offspring of a given node is also useful to help decide about the quiescence of nodes. The term "quiescent" is used in the literature to qualify a node that is not really worth expanding any further. Ideally, static evaluations should only be applied to quiescent nodes. A typical example of a non-quiescent position would be a check in chess.

Even though the actual quiescence of a position could theoretically be unrelated to the number of legal moves from that position, there are at least two reasons why this is not too likely to happen. First, the very purpose of the introduction of quiescent nodes was to control

the growth of the search tree. The important parameter here is the branching degree of each and every position. Second, our intuition of a non-quiet position corresponds to a position where there is an immediate threat. The rules of the game may even dictate that this threat be removed at all cost, as with a check in chess. In fact positions with very few actual options in chess are almost exclusively check positions and are therefore all non-quiet.

In what follows we take it as an axiom that the single most common feature of non-quiet positions is that they tend to have fewer options than the average position. Exactly which positions are quiet is something the programmer has to decide carefully. It turns out, as we will soon see, that the percentage of non-quiet positions among internal nodes is merely a multiplicative factor for the actual parameter  $b$  of the game tree considered. Through his decisions concerning quietness, the programmer does therefore control the survival rate  $b$  seen by his procedure. It is therefore crucial to know the range of  $b$  that ensures termination of a specific procedure. This concern was the primary motivation for the study done at the end of section 4. Since we only use boolean estimates here, the results of section 4 are directly applicable.

It may or may not be the case that the frequency of quiet positions is the only growth control factor. However, non-quiet nodes should always be searched further. Therefore, if the game playing procedure is expected to terminate almost surely in a natural way, the non-quiet nodes should not appear too often. More precisely, if only

non-quiescent nodes are expanded, the procedure should be guaranteed to terminate. An actual game-playing procedure could include some code to ensure termination in any case and force the static evaluation of even non-quiescent nodes beyond a certain depth. However this should only be seen as a catastrophe handler. If it were to be executed on a regular basis, many occurrences of the horizon effect would have to be feared and overall performance would suffer greatly.

Definition 7.1

The minimal search tree of a game is the subtree containing the root whose internal nodes are all non-quiescent and whose leaves are all quiescent.

Quiescence must therefore be defined so that the minimal search tree of the game is either finite or easy to solve. The latter means that the straightforward procedure SOLVE will almost always terminate. Furthermore, since there is a rather high correlation between quiescence and winning status, SOLVE should have an easy task even under the misère play rule (the worst case). [\*] This analysis has already been done in section 4. If we call  $\bar{b}$  and  $\bar{\kappa}$  the survival rate and internal structure of the minimal search tree, we know that the minimal search tree is almost surely SOLVEable even under the misère play rule when:

$$\bar{\kappa}(1 - \frac{\bar{b}}{2}) \geq \frac{1}{2}$$

Since the above relation must be satisfied by any admissible quiescence

\* A nonquiescent node is a position with few options and therefore a probable loss. The minimal search tree of a game tends therefore to obey the misère play rule.



scheme, we will refer to it as the quiescence consistency condition. As just stated, this condition does not assume anything about the particular quiescence scheme used. In general, most of our discussion in this section applies to any scheme. However, we find it beneficial to focus on the scheme we call standard. Even though our discussion is given in quite general terms, this is really the model we have in mind. We will therefore not make any further semantic distinction between 'quiescence' and 'standard quiescence'. Under the standard scheme considered here, a node is quiescent when it has more than  $Q$  offsprings.

Definition 7.2

The quiescence threshold  $Q$  is the largest number of options that a nonquiescent node may have. Internal nodes with  $Q$  options or less are nonquiescent. All other nodes are quiescent.

This implies the two relations:

$$\bar{\sigma} = b \sum_{n=1}^Q k_n$$

$$K_n = \frac{k_n}{\sum_{n=1}^Q k_n}$$

If  $K$  is the geometric distribution,  $k_n = (1-a)a^{n-1}$ , this yields:

$$\bar{\sigma} = b (1-a^Q)$$

$$K(z) = \frac{(1-a)z}{1-az} \cdot \frac{1-(az)^Q}{1-a^Q}$$

Again, we need only consider the worst case where the natural survival rate  $b$  is close to unity. In which case, the quiescence consistency condition reduces to:

$$\frac{1}{2} \geq \kappa \left( \frac{1+a^Q}{2} \right)$$

This relation was used to compute the table of Figure 7.1. There we tabulated the lowest value of  $d$  [\*] that guarantees that the minimal search tree can almost surely be solved when the quiescence threshold is  $Q$ . Notice that non-quiescent nodes should be rare enough if the quiescence consistency condition is to be respected. Augmenting the average degree of the game actually reduces the proportion of non-quiescent nodes. For a given  $Q$ , the bushier the game tree, the skinnier its minimal search tree.

For reference purposes, we included in Table 7.1 the lowest degree  $d$  for which the minimal search tree is almost surely finite. The minimal search tree is almost surely finite when  $\sum_{n=1}^Q nk_n \leq 1$ . In the geometric case, this reduces to:

$$Q a^{Q+1} - (Q+1)a^Q + a \leq 0$$

\* Recall that  $d = 1/(1-a)$  is the average degree of an internal node in the geometric distribution.

Figure 7.1

Acceptable quiescences threshold for the geometric distribution.

For a quiescence threshold $Q =$	The Minimal Search tree is	
	SOLVEable when $d \geq$	Finite when $d \geq$
1	1	1
2	1.565	2
3	3.046	4.303
4	5.185	7.627
5	7.959	11.956
6	11.363	17.287
7	15.396	23.618
8	20.057	30.950
9	25.346	39.283
10	31.263	48.616
11	37.808	58.949
12	44.980	70.282
20	124.949	196.947
30	281.387	445.279
40	500.576	793.612

The quiescence consistency condition turns out to be a crucial condition for the almost certain termination of the two procedures HSOLVE and QSOLVE that we are now going to describe. HSOLVE and QSOLVE are directly inspired from commonly used procedures. For some obscure reason, HSOLVE is much more popular than QSOLVE. We support the opposite

choice and will give both intuitive and quantitative arguments in favor of QSOLVE.

```
FUNCTION HSOLVE (G : Game, M : Integer): Status
  IF G is a leaf THEN RETURN its status
  IF M is zero THEN
    IF G is quiescent, RETURN ESTIMATE(G)
    ELSE RETURN HEXPAND (G,0)
  ELSE RETURN HEXPAND (G,M-1)
ENDFUNC.
```

```
FUNCTION QSOLVE (G : Game, M:Integer): Status
  IF G is a leaf THEN RETURN its status
  IF M is zero THEN
    IF G is quiescent, RETURN ESTIMATE(G)
    ELSE QEXPAND (G,0)
  ELSE IF G is quiescent, RETURN QEXPAND(G,M-1)
    ELSE RETURN QEXPAND(G,M)
ENDFUNC.
```

Besides the function ESTIMATE that returns the estimated status of a quiescent node, HSOLVE and QSOLVE each use their own function yEXPAND. ySOLVE and yEXPAND are mutually recursive.

```

FUNCTION yEXPAND(G : Game, M : Integer): Status
    Compute from left to right the Status ySOLVE(G',M) of each option
    G' of G until one is found which is a LOSS.
    IF G has such a LOSS option, RETURN WIN
    ELSE RETURN LOSS
ENDFUNC.

```

Figure 7.2 illustrate how the search trees of QSOLVE and HSOLVE grow for increasing values of M. For a given M, QSOLVE always expands at least all the nodes HSOLVE does. This means that when typical time constraints are imposed, QSOLVE will have to be called with a lower value of the argument M than the one we could afford with HSOLVE. However, QSOLVE seems a priori a more promising approach. The search seems more balanced and smarter in QSOLVE than it is for HSOLVE. [\*]

The characteristic of QSOLVE is to try out all sequences of moves that include a given number of nontrivial moves. For example, when a given node has only one option, it is foolish to consider that this single option should be given any less attention than what we intended to give to the father node. In general, searching an only child should never be considered as having gone one level deeper in the tree. QSOLVE manages to handle such situations, and many more, by considering as 'trivial' all moves that are made from a quiescent position. By con-

\*—— QSOLVE is a special case of a more general procedure TSOLVE that distinguishes trivial and nontrivial moves: a node reached through a trivial move deserves the same attention as its father. TSOLVE has essentially the same qualities as QSOLVE but it is more flexible and more practical; in chess, captures are trivial moves while positions that allow captures need not be non-quiescent.

Figure 7.2 : Visible Nodes

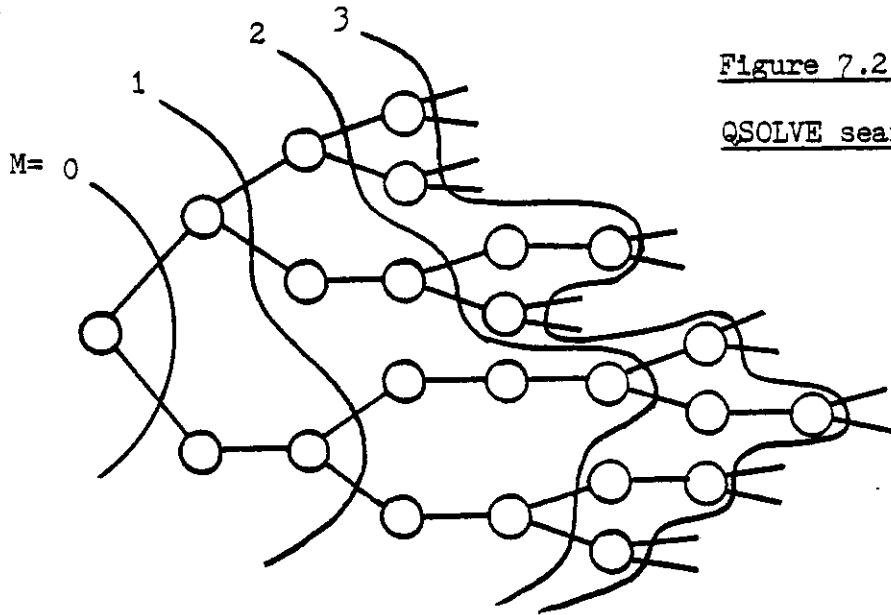


Figure 7.2.a

QSOLVE search (Q=1)

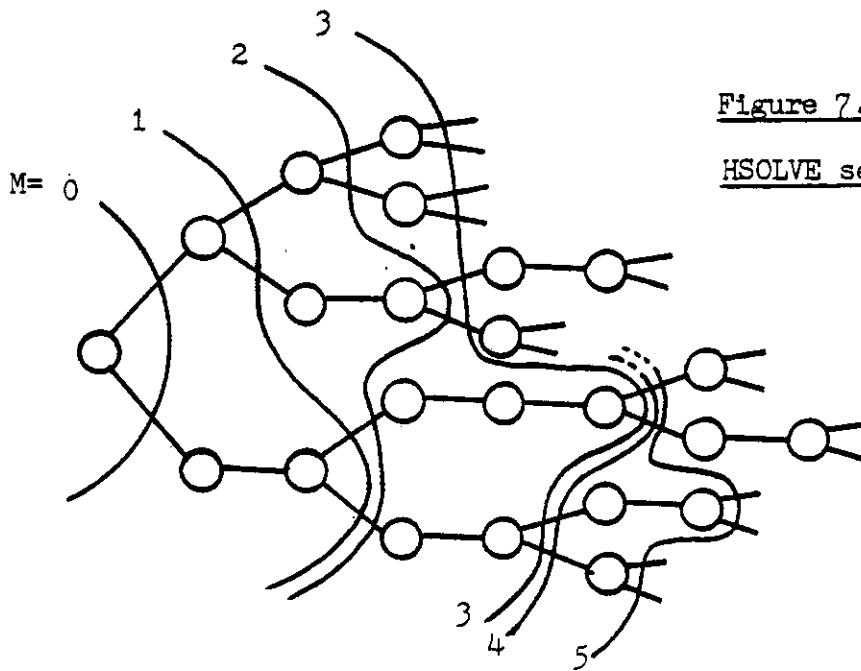


Figure 7.2.b

HSOLVE search (Q=1)

trast, HSOLVE will react correctly only to the non-quiescent nodes that are on its search frontier. The problem is that there might well be a trivial path that leads from the root to a quiescent node on the search frontier of HSOLVE. An intelligent player is likely to zero in on that path even if it is 15 or 20 moves long. By ensuring that any path leading to its search frontier involves a given number of nontrivial decisions, QSOLVE may protect itself efficiently against such a smart opponent.

Another argument against HSOLVE is the following. Consider the last node of a chain of  $k$  consecutive nonquiescent nodes. If such a node is visible [\*] for the first time for a certain value of  $M$ , none of its successors will be visible until  $M$  reaches the value  $M+k$ . Besides,  $M+k$  could well be too large for this ever to take place. Considering that the algorithm detected such a hot sequence relatively early, this is a shameful waste of opportunity. QSOLVE, on the contrary, is paranoid about not wasting such opportunities. Every successor of a node visible at level  $M$  will always be visible at level  $M+1$ .

The nodes that are visible at level 0, for either HSOLVE or QSOLVE since both procedures are identical when  $M=0$ , are precisely the nodes in the minimal search tree. Therefore the quiescence consistency condition introduced earlier is sufficient to ensure the termination of HSOLVE or QSOLVE at level  $M=0$ . The following Theorem therefore shows that it en-

---

\* We say that a node is visible when it would be actually visited if only the 'pure' growth control was in effect, disregarding the game related cutoffs. In other words, the visible nodes are those nodes that would be expanded by  $y$ SOLVE ( $y = Q$  or  $H$ ) if  $y$ EXPAND was systematically computing the status of each and every possible option of  $G$  before returning a result.

sure termination of both procedures at any level M.

Theorem 7.1

HSOLVE (G,M) and QSOLVE (G,M) almost surely terminate if and only if HSOLVE (G,0) [or QSOLVE (G,0) ] does.

The result concerning HSOLVE is actually elementary. The number of nodes at depth M is certainly finite and each one of them that HSOLVE actually reaches will almost surely be solved because HSOLVE(.,0) almost always terminates. Therefore HSOLVE(G,M) will almost surely terminate.

A similar result concerning QSOLVE requires an induction on M. If we know that QSOLVE(.,M-1) almost surely terminates, we know that each leaf of the minimal search tree of G will almost surely be solved if QSOLVE reaches it. This is so because QSOLVE simply requires that all the options of such leaves be solved at level M-1 only. Since the minimal search tree itself is almost surely solved once the status of its leaves is known, QSOLVE will almost always terminate.

The rest of this section will be concerned with the quality of the decisions obtained through either HSOLVE or QSOLVE. This is to be contrasted with the discussion given at the end of the previous section for a straight search. [\*] The notations we will now use are consistent with those of section 6 once it is understood that all positions were considered quiescent there. Furthermore, we now make assumptions similar to those of section 6. In particular, we assume that no real leaf is ever encountered before we truncate our search. This implies that the

---

\* The no-frills search, as discussed in section 6, can be seen as a special case of the type of search given here (a la QSOLVE or HSOLVE) when all nodes are considered quiescent.



quantity  $\bar{\sigma}$  used in the rest of this section corresponds to  $b=1$ ; for example,  $\bar{\sigma} = 1-a^Q$  for the geometric distribution and a standard quiescence scheme with threshold  $Q$ . The subscript  $n$  is the value of the parameter  $M$  of QSOLVE.

$W_n$  is the probability that a quiescent node is a correctly estimated WIN.

$L_n$  is the probability that a quiescent node is a correctly estimated LOSS.

$\bar{W}_n$  is the probability that a quiescent node is a WIN misinterpreted as a LOSS.

$\bar{L}_n$  is the probability that a quiescent node is a LOSS misinterpreted as a WIN.

We will also use the notations  $q_W^n$ ,  $q_L^n$ ,  $q_W^n$  and  $q_L^n$  for the corresponding unconditioned quantities. For example  $q_W^n$  is the probability that a node, quiescent or not, is correctly interpreted as a WIN by QSOLVE when  $M=n$ . Finally,  $\mathbb{H}$  will denote the internal structure of quiescent nodes.  $\mathbb{H}(z)$  is best defined via the relation:

$$K(z) = (1-\bar{\sigma})\mathbb{H}(z) + \bar{\sigma} \mathbb{K}(z)$$

The quantities just introduced satisfy the following relations that are easily obtained through a simple derivation similar to that used in section 6.

$$q_W^n = (1-\delta)\bar{w}_n + \delta [ \kappa(q_W^n + q_L^n) - \kappa(q_W^n) ]$$

$$q_L^n = (1-\delta)\bar{L}_n + \delta [ \kappa(q_W^n + q_L^n) - \kappa(q_W^n) ]$$

$$q_L^n = (1-\delta)L_n + \delta \kappa(q_W^n)$$

$$q_W^n = 1 - q_L^n - q_L^n - q_L^n$$

In addition, the fact that options of quiescent nodes visited at level n+1 by QSOLVE are visited [\*] at level n translates into:

$$\bar{w}_{n+1} = H(q_W^n + q_L^n) - H(q_W^n)$$

$$\bar{L}_{n+1} = H(q_W^n + q_L^n) - H(q_W^n)$$

$$L_{n+1} = H(q_W^n)$$

$$W_{n+1} = 1 - \bar{w}_{n+1} - L_{n+1} - \bar{L}_{n+1}$$

Notice that some other handy relations like  $q_W^n + q_L^n = t$  are still valid here for the unconditioned quantities. The corresponding relation for the quantities relating to quiescent nodes only is slightly different, namely  $w_n + \bar{w}_n = t_0$ , where  $t_0 = 1 - \bar{H}(t)$  is the probability that a quiescent node is a win.

\* By visited here, what we really mean is of course that they would be visited if no 'cutoff' (an elder LOSS sibling) occurs.

Before we proceed in analyzing QSOLVE, we should notice that the fate of HSOLVE is easily settled through the above expressions for  $n=0$ . In particular, we obtain

$$\left( \frac{q_W^0}{W} - \frac{q_L^0}{L} \right) = (1-\beta) (W_0 - L_0) + \beta [K(t - (\frac{q_W^0}{W} - \frac{q_L^0}{L})) - K(t)]$$

Since the sign of the second term is clearly the opposite of that of  $(\frac{q_W^0}{W} - \frac{q_L^0}{L})$ , it follows that

$$\left| \frac{q_W^0}{W} - \frac{q_L^0}{L} \right| \leq (1-\beta) |W_0 - L_0|$$

Now, we know from section 6 that the error term for a deep 'straight' search (of even depth) converges precisely toward  $|W_0 - L_0|$  for the geometric distribution when all nodes are considered quiescent. We conjecture that this quantity is likely to be at least as easy to minimize when only nodes of high degree are considered quiescent. Furthermore, if we consider that:

$$M_0 = \begin{bmatrix} q_W^0 & q_W^0 \\ q_L^0 & q_L^0 \end{bmatrix}$$

then the analysis made in section 6 applies fully to HSOLVE since this is what HSOLVE really perceives as the static estimation matrix for the nodes at depth  $M$ . The above inequality shows that the matrix is even more balanced than was the static evaluation matrix for quiescent nodes. What this means is that, at least for the geometric distribution, HSOLVE is strictly more efficient than a straight (fixed depth) search. This argument, appealing as it is, still relies on the unsupported claim that

it is at least as easy to 'balance' a static evaluation on quiescent nodes as it is in the general case.

The analysis of QSOLVE is somewhat more complex but the conclusions are more definite and more interesting.

Let  $M_n$  be the matrix  $\begin{bmatrix} q_W^n & q_W^n \\ q_L^n & q_L^n \end{bmatrix}$ . The perfect matrix  $\begin{bmatrix} t & 0 \\ 0 & 1-t \end{bmatrix}$  is

a trivial fixed point for  $M_n$  as is seen either through the above relations or through common sense. What matters, however, is the stability of this fixed point. A stable fixed point will be strong evidence that the probability of error in QSOLVE converges toward zero while an unstable fixed point shows that the probability of error can never be zero. [\*] Let:

$$M_n = \begin{bmatrix} t-\alpha_n & \alpha_n \\ \beta_n & 1-t-\beta_n \end{bmatrix}$$

where  $\alpha_n$  and  $\beta_n$  are small quantities that we will treat as infinitesimals. We obtain the following relations by differentiating the above recurrence equations.

$$\alpha_{n+1} = (1-\bar{\delta})\bar{\pi} \beta_n + \bar{\delta} \bar{\kappa} \beta_{n+1}$$

$$\beta_{n+1} = (1-\bar{\delta})\bar{\pi} \alpha_n + \bar{\delta} \bar{\kappa} \alpha_{n+1}$$

where  $\bar{\kappa} = \frac{dK}{dz}(t)$  and  $\bar{\pi} = \frac{dH}{dz}(t)$ . This can be rewritten as

\* Unless the static evaluation was perfect to begin with, a case we systematically disregard here.

$$\begin{bmatrix} \alpha_{n+1} \\ \beta_{n+1} \end{bmatrix} = \frac{(1-\delta)\pi}{1-(\delta\kappa)^2} \begin{bmatrix} \delta\kappa & 1 \\ 1 & \delta\kappa \end{bmatrix} \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix}$$

The solution to this recurrence is

$$\begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix} = \frac{\alpha_0 + \beta_0}{2} \left( \frac{(1-\delta)\pi}{1-\delta\kappa} \right)^n \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \frac{\alpha_0 - \beta_0}{2} \left( \frac{(1-\delta)\pi}{1+\delta\kappa} \right)^n \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The second term is negligible compared to the first. Therefore, the stability condition reduces to:

$$\left| \frac{(1-\delta)\pi}{1-\delta\kappa} \right| < 1$$

Since  $\frac{dK}{dz}(\xi_K) = (1-\delta)\pi + \delta\kappa$ , one can easily see that this condition is satisfied if  $\frac{dK}{dz}(\xi_K) \leq 1$ , as is always the case with inert structures. If the structure is barely inert, like the geometric distribution, the above ratio is equal to one. In such a case, the above first-order analysis is not sufficient. In fact, what happens is that higher order derivatives of  $\pi$  and  $\kappa$  will still make  $\alpha$  and  $\beta$  converge toward zero. [\*] This convergence, will be quite slow in its final stage. However, the difference between  $\alpha$  and  $\beta$  will converge exponentially fast toward zero, as predicted by the above relations. Finally, one should note that even though the rate of convergence of  $\alpha$  and  $\beta$  is eventually quite slow, it is initially rather fast. One reaches error rates as low as 2% or 3% for searches as shallow as  $n=3$  or 4, even with a poor static evaluation function. This, of course, is very important in practice since  $n$  cannot

\* This remark is based on a computer-aided study of the geometric distribution, based on the general equations given in this section.

usually be much greater than 5 or 6.

Theorem 7.2

The probability that QSOLVE will misestimate a node converges toward zero, for the geometric distribution, as the parameter  $M$  increases.

Note that Theorem 7.2 holds true regardless of how poor the initial static evaluation is. The static evaluation could even be worse than random guessing. Theorem 7.2 would still hold true if our static evaluation was systematically considering all LOSSES as WINS and vice-versa!

## 8. Estimating the winning probability after a truncated search

In this section we address the following question. Given a game taken from an ensemble of impartial recursive random games, what is the probability that this game is a next player win? For the sake of simplicity, we will assume throughout this section that we are below transition, no matter how slightly; a game can therefore only be a win (with probability  $t$ ) or a loss (with probability  $(1-t)$ ).

Clearly, we will always be able to answer 0 or 1 to the above question if we search the game exhaustively. On the other hand, if we do not search the game at all, our best answer is that we have a probability  $t$  of being able to win it, assuming, for the moment, that  $t$  and other relevant general statistical parameters are known. However, this estimate will change as more information is obtained, if we perform a limited search from the start position. More precisely, this estimate will merely be a conditional probability: the probability that the root is a WIN, given the structure of the tree searched so far. If we know the parameter  $t$ , this conditional probability can be computed.

Winning probabilities neither propagate according to a minimax rule nor allow the same cutoffs, but follow a product rule. More precisely, if  $t_1, t_2, t_3, \dots, t_n$  are the respective probabilities that the  $n$  children of a given node are wins, then the probability that the parent node is a win will be  $(1-t_1 t_2 t_3 \dots t_n)$ .

We will now focus on the easiest search procedure to analyze, namely a breadth first search where the shallowest node is expanded up

to a fixed depth. We should stress that we do not advocate this method for actual game playing. The following analysis is essentially meant to be compared to what is obtained under similar assumptions in section 6 for the minimax rule.

Let us call  $T_n$  the winning probability of the root node estimated from the search up to depth  $n$ . Since  $T_n$  is an actual conditional probability, it is certainly an unbiased estimator of the uninformed winning probability  $t$  and its expected value is therefore  $t$ . We find it useful, however, to sketch an elementary inductive proof of this because the proof allows us to study the evolution of the bias of  $T_n$  when  $T_0$  is only an estimate of  $t$ . First we notice that: [\*]

$$E(T_{n+1}) = (1-b)p + b \sum_{i=1}^{\infty} k_p (1-E(T_n))^p = G(E(T_n))$$

Therefore, since  $T_0 = t$  and  $G(t) = t$  by definition, we conclude by induction that  $E(T_n) = t$ . So,  $T_n$  is unbiased. The above recurrence also shows that we do not really need to know the precise value of  $t$  and could use a slightly different value for  $T_0$ . The bias fades away as  $n$  gets large and  $E(T_n)$  converges toward  $t$  in a way similar to that illustrated in Figure 2.3.a.

Of course, the mean of  $T_n$  is of only theoretical value to us: what we would like to know is how good is  $T_n$  at distinguishing between likely wins and losses. One easy way to measure such performance is to study the variance of  $T_n$ , namely  $V(T_n) = E(T_n^2) - t^2$ . The higher the variance,

\* Note that a node is recognized as terminal only when it is expanded and children are estimated independently.



the better. The best, maximum variance is  $t(1-t)$  and is obtained for an estimator that returns only 0 or 1, and does so with the corresponding probabilities of losses and wins, so that its mean is still  $t$ . An optimal variance does not necessarily mean our estimator does a perfect discrimination between wins and losses, but it is a strong clue if we know that our estimator was obtained by honest means, by gathering information from the search and not by flipping an irrelevant coin. We call such an estimator statistically perfect. What this means is merely that such an estimator will only give yes/no answers (0 or 1) and do so with the same frequency the perfect estimator would. Even though the variance is not a foolproof indicator in the way the conditional probability is, there are several reasons why the study of its behavior is appealing. Probably the primary reason is that the variance is a good indicator of the clustering of the estimated probability of a win around the values 0 and 1. This indicates the power of the evaluation procedure in distinguishing between wins and losses. Furthermore, we can prove that the variance of  $T_n$  converges toward  $t(1-t)$  even if our estimate  $T_0$  differs from  $t$ . Therefore an experimental estimate of the variance can help directly in refining our initial estimation of  $t$ .

We now compute  $E(T_n^2)$  and prove  $T_n$  becomes statistically perfect for large values of  $n$  and does so rather quickly ...

$$E(T_{n+1}^2) = (1-b)p + b \sum_{i=1}^{\infty} k_p E((1-T_{n,1}T_{n,2}\dots T_{n,i})^2)$$

$$= (1-b)p + b \sum_{i=1}^{\infty} k_p (1-2E(T_n))^p + E(T_n^2)^p$$

$$= 2G(E(T_n)) - G(E(T_n^2))$$

In other words, with  $w_n = E(T_n^2)$ , we have :

$$w_0 = t^2$$

$$w_{n+1} = 2t - G(w_n)$$

Since  $t$  is the fixed point of  $G$ , it does not take long to see that it is also a fixed point of the recurrence equation for  $w_n$ . Furthermore, the shape of  $2t-G(z)$  (all derivatives positive) implies it has two fixed points of which only  $t$  is stable. Our assumption of being below transition precisely implies  $-\frac{dG}{dz}(t) = m < 1$ . Therefore,  $w_n$  is nondecreasing and has a limit of  $t$ . Hence, the limit of the variance of  $T_n$  is the perfect value  $t(1-t)$ . The rate of convergence depends on the parameter  $m$ .

We believe such an analysis of the variance is a good enough measure of the performance of an estimator. However, as we previously observed, it is not entirely foolproof. Here we can almost as easily derive a foolproof indication of good performance, namely the expected value  $E_W(T_n)$  of  $T_n$ , given that the position is a win. The reader may want to use a scheme similar to the above in conjunction with the obvious relation:

$$E(T_n) = tE_W(T_n) + (1-t)E_L(T_n)$$

then obtain the following relation that holds even if  $T_0$  is not equal to  $t$ :

$$tE_W(T_{n+1}) - t = b(K(tE_W(T_n)) - K(E(T_n)))$$

and use it to show that  $E_W(T_n)$  approaches 1 when  $n$  gets large.

It is noteworthy that the provably good performance of the product rule also holds when  $T_0$  is not exactly equal to the usually unknown parameter  $t$ . If this is the case,  $T_n$  is certainly a biased estimator of  $t$ , although this bias tends to fade away quickly as  $n$  increases. Furthermore,  $V(T_n)$  approaches the optimal value  $t(1-t)$  at a rate essentially similar to the one observed in the unbiased case.

What is even more remarkable is the fact that product rule estimates enable us to skillfully play games that would be absolutely intractable to a minimax procedure. Consider, for example, a game whose statistical structure is known to be inert. The above analysis shows that even if  $b$  is very close to unity, the product estimates will converge rather quickly toward the perfect evaluation. Product rule estimates tend to gather information that is present very early in the search tree and to be relatively insensitive to the remoteness of game termination. By contrast, minimax procedures are based essentially on the game logic that is undoubtedly valid for either endgames or brilliant short term tactical gains. However, the validity of the minimax approach when such definite tactical advantages are not accessible is highly questionable and has been seriously challenged recently by Nau

[20] and Pearl [24] . Our opinion is that the product rule is highly valuable as a tie-breaker among those options that have a similar minimax value. We get the best of both worlds if such a philosophy is used. We will still be able to detect checkmates or definite tactical advantages that can be reached in very few moves but will not base our decisions on meaningless minimax values when none is in sight. Present day chess-playing programs are known often to base their play on minute variations of the minimax values and are therefore extremely sensitive to minute changes in the way nodes are evaluated. Such behavior is highly undesirable as it makes use of the static evaluation function for purposes it was not really designed for. This is why we conclude this section with a few practical reflections.

A good game playing program should successively answer the three basic questions :

Is the position quiescent ?

If it is we evaluate it either statically, or by using either the product rule or the minimax rule if it was decided beforehand that some time should be spent examining this position somewhat deeper. In other words, the children of a quiescent node are examined, if at all, less carefully (e.g. with a tighter depth limit) than the parent node.

If it is not, we decide that its children should be examined with essentially the same care as the parent node.

Practically, one could start the search with a certain depth credit and consume one unit when trying out a totally quiescent move, a move that does not seem to make things change a lot, and consume nothing for a move that introduces a drastic change in the position. [\*]

Is there a brilliant short term combination ?

This is taken care of by a minimax approach and is a dominant factor in a decision, should the answer be positive.

Was is the most a priori promising move ?

This is best decided with a limited search using the product rule when no 'minimax-like' advantage is in sight.

---

\* Quiescence scores between 0 and 1 allows unlimited 'fine-tuning'. However, the author participated in the development of an otherwise very simple program that used only 0 and 1 quiescence and could defeat experts (France's 1980 junior champion of International Checkers lost one game at tournament speed even though the program was running on a microprocessor !).

$$\xi_K = \frac{1 - \sqrt{1-a}}{a}$$

$$\frac{dK(z)}{dz} = \frac{1-a}{(1-az)^2}$$

$$d = \frac{dK(1)}{dz} = \frac{1}{1-a} \quad (\text{Average internal degree})$$

$$\frac{dK(\xi_K)}{dz} = 1$$

$$z = 1 - K(1 - K(z)) \quad (\text{There are infinitely many stable two-cycles !})$$

This last property of the geometric distribution is used to build the strange internal structures given in Appendix A. Such strange inert structures yield a shape for the TIE region that departs somewhat from that illustrated in Figure 3.1. In particular, the TIE region may be of a height strictly smaller than unity around  $b=1$  and/or it may include vertical segments of finite length. [\*] More details can be found in Appendix A.

This whole spectrum of possibilities that lies between inert structures, like the geometric distribution, and the typical non inert structures (as illustrated in Figure 3.1) may appear only in some contrived examples for the impartial case we have discussed so far. However, they seem the rule rather than the exception for the partisan games we shall discuss in section 5.

\* When this happens, such vertical segments are to be counted as included in the TIE region. So, for all intents and purposes, we may consider the TIE region as closed and both the WIN and LOSS regions as open (within the unit square minus its rightmost side).

## APPENDIX

### A. Strange non-inert internal structures.

A necessary condition for inertness is that the fixed point of  $K$  be stable. [\*] The following shows that this is not sufficient by giving an explicit counter-example. This internal structure  $K$  can be used to build real game structures  $G$  that do allow ties even though they yield  $m < 1$  (where  $m = -\frac{dG}{dz}(t)$ , as introduced in section 1). This is achieved by taking  $b$  close enough to unity, as will be illustrated numerically.

Our construction is based on the geometric distribution as introduced at the end of section 3.

$$K_d(z) = \frac{z}{d-(d-1)z}$$

The geometric distribution  $K_d$  itself is inert: a game structure  $G$  based on it has no two-cycles whatsoever. Now, consider the internal structure:

$$K(z) = K_d(z) - \alpha z(1-z)(z-z_0) \prod_{i=1}^n (z-z_i)(z'_i-z)$$

---

\* We are considering only impartial games here.

where:

- \*  $\alpha$  is a positive or negative number small enough to ensure that all the coefficients in the expansion of  $K(z)$  are positive.

$$\left( \alpha = \pm \frac{\left(1 - \frac{1}{d}\right)^{2n+3}}{\binom{2n+2}{n+1}} \text{ does the job irrespective of the } z_i \text{ values} \right)$$

- \*  $z_0$  is the fixed point of  $1-K_d$ . That is  $z_0 = \frac{d - \sqrt{d^2 - 4}}{d-1}$ .

- \* The  $n$  numbers  $z_i$  are distinct numbers strictly between 0 and  $z_0$ .

- \*  $z'_i = 1-K_d(z_i)$  (which implies that  $z_i = 1-K_d(z'_i)$  because of the fundamental symmetry of  $1-K_d$  which makes the whole construction work so neatly).

By construction,  $z_0$  is the fixed point of  $1-K$ ; the  $n$  pairs  $(z_i, z'_i)$  are the only two-cycles of  $1-K$  besides  $(0,1)$ . In addition:

If  $\alpha$  is positive, then  $z_0$  is stable and  $1-K$  has  $\left\lfloor \frac{n}{2} \right\rfloor$  stable two-cycles.

If  $\alpha$  is negative, then  $z_0$  is not stable and  $1-K$  has  $\left\lceil \frac{n}{2} \right\rceil$  stable two-cycles.

Now, unlike what happens with the plain geometric distribution, the two-cycles do not disappear for a real game structure  $G=(1-b)p+b(1-K)$ , provided  $b$  is close enough to unity. In fact one extra stable two-cycle  $(0+\epsilon, 1-\epsilon')$  is even created whenever the outermost two-cycle of  $K$  is unstable.



For example, we studied numerically the internal structure:

$$K(z) = K_2(z) - \alpha z(1-z)(z-(2-\sqrt{2}))(z-\frac{1}{2})(\frac{2}{3}-z)$$

where  $\alpha = \pm \frac{1}{45}$ , slightly below the largest acceptable value. The values corresponding to the matched case ( $p = 2-\sqrt{2}$ ) are tabulated in Figure A.1. All entries are percentage probabilities. In addition, the entry  $b=100\%$  is to be interpreted as the limit case when  $b$  is infinitely close to 1.  $b=1$  being, of course, ruled out.

Figure A.1

Two atypical non-inert internal structures

b =	$\alpha = 1/45$ (stable fixed point)		$\alpha = -1/45$ (unstable fixed point)	
	$p_W =$	$1-p_L =$	$p_W =$	$1-p_L =$
0	58.579	58.579	58.579	58.579
99.9701897	58.579	58.579	58.579	58.579
99.9701899	6.114	96.853	58.579	58.579
99.9958	0.650	99.675	58.579	58.579
99.999	0.150	99.925	51.275	65.524
100	0	100	50.000	66.667

Figure A.1 shows the two kinds of strange features internal structures with at least one non trivial two-cycle can exhibit. [\*]

\* Here the only non-trivial two-cycle is (1/2, 2/3).

First, as exemplified by the case  $\alpha = \frac{1}{45}$ , the probabilities  $p_W$ ,  $p_L$  and  $p_T$  may no longer be a continuous function of  $b$ . Ties have zero probability below a certain threshold of approximately  $b = .9997018982$  but, at or above that threshold, ties occur with a probability greater than 90% !

Second, as shown in the case  $\alpha = -\frac{1}{45}$ , the probability of ties may not converge toward 1. Here, the probability  $p_T$  is a continuous function of  $b$  which is equal to zero below the threshold  $b = .9999580055$ . It has an infinite slope at that point and converges toward  $\frac{1}{6}$  as  $b$  approaches unity.

## B. Necessary and sufficient conditions for inertness.

### Theorem B.1

A partisan internal structure  $(K, K')$  is inert if and only if there exists a number  $\xi_K$  in  $]0, 1[$  such that:

$$\left\{ \begin{array}{ll} 1-K(1-K'(x)) \geq x & \text{If } x \text{ is smaller than } \xi_K \\ 1-K(1-K'(x)) \leq x & \text{If } x \text{ is larger than } \xi_K \end{array} \right.$$

This can be stated in a more pictorial way, namely:

### Theorem B.2

An internal structure  $(K, K')$  is inert if and only if:

either the curve  $\Gamma$  of equation  $y=1-K(x)$  and the curve  $\Gamma'$  of equation  $y=1-K'(x)$  coincide,

or the two curves  $\Gamma$  and  $\Gamma'$  have only three points of intersection (namely  $(0,1), (1,0)$  and  $I = (\xi_K, \xi_K)$ ) and  $\Gamma$  is below  $\Gamma'$  if and only if  $x$  is below  $\xi_K$ .

The reason why those conditions are necessary is a consequence from Theorem 5.1. If they were violated, the two curves  $\Gamma$  and  $\Gamma'$  would have to cross each other more than once in the interval  $]0, 1[$ . As a result, the two curves of equation  $y=G(x)$  and  $x=G(y)$  would have more than one point of intersection if the parameter  $b$  is chosen close enough to unity. Ties would therefore occur, proving  $K$  not inert.

Those conditions are also sufficient to ensure the absence of ties in the matched case where  $p = \xi_K$  and  $p' = \xi_{K'}$ , since the curve of equa-

tion  $y = G'(x)$  is then below  $\Gamma$  when  $x \leq \tilde{x}_K$  or, equivalently, when  $y \geq \tilde{y}_{K'}$ , and above it when  $x > \tilde{x}_K$ . The converse holds true for the curve of equation  $x = G(y)$  with respect to  $\Gamma'$ . Therefore, if the relative positions of  $\Gamma$  and  $\Gamma'$  are as indicated in Theorem B.2, the two curves  $[y = G(x)]$  and  $[x = G'(y)]$  can only have one point of intersection (at  $x = \tilde{x}_K$ ). Therefore ties do not occur.

The above necessary and sufficient conditions make a statement that should be satisfied throughout the range of the abscissa  $x$ . In particular, it should hold in the neighborhood of  $x = \tilde{x}_K$ . This is only a necessary condition. But it turns out to be a tight one which is virtually sufficient in practice. Non-inert structures that satisfy it are rather difficult to come up with (see Appendix A). Besides, one should be aware of the great practical similarity there is between ties and games that are expected to last several thousand moves (Appendix A).

### Theorem B.3

An internal structure  $(K, K')$  is inert only if

$$\frac{dK}{dz}(\tilde{y}_{K'}) \frac{dK'}{dz}(\tilde{x}_K) \leq 1$$

### C. Standard pruning transforms N and M.

Some modifications of the rules of a game do not alter the way the game is played. For example, chess is essentially a game whose goal is to capture the opponent's king. [\*] However the traditional rules say that the actual capture should never be carried out. Instead, the game stops whenever the loser cannot avoid the capture of his king in the next move. Moreover, experienced players routinely decide to stop a game one move or more before the checkmate. The game is hardly modified by such conventions.

We call a convention that eliminates some legal moves while not modifying the way the game is played a standard pruning transform. Among the many possible standard pruning transforms we distinguish two that have the property of preserving the general probabilistic scheme described in this paper.

The standard pruning transform N, maps any game into a normal game, while the image of a game through M is a misère game. We will study N first.

The mapping N associates a normal game tree  $N(GT)$  to any given game tree GT with the exception of the game  $\langle W \rangle$  (whose very root is a

---

\* We disregard various special rules, like the 'pat' rule, that allow a game of chess to be drawn in a finite number of moves.

terminal WIN).

Definition C.1

The standard pruning transform  $N$  is recursively defined as follows:

\*  $N(\langle L \rangle) = \langle L \rangle$

\* If  $GT$  is an internal node whose options are all external WINS, then its image  $N(GT)$  is a terminal LOSS node.

\* If the root of  $GT$  is an internal node whose options besides external WINS are  $G_1, G_2, \dots, G_n$ , then the options at the root of  $N(GT)$  are  $N(G_1), N(G_2), \dots, N(G_n)$ .

To put it bluntly,  $N(GT)$  is simply  $GT$  stripped of all its useless WIN tip nodes. Now, it should be clear that, the WIN, LOSS or TIE status of  $N(GT)$  is the same as that of  $GT$ . In fact, there is virtually no difference between  $GT$  and  $N(GT)$  as far as game-playing is concerned. This is also true in a statistical sense and we have to expect virtually identical results whether we use a certain probability distribution  $P$  or its image  $\hat{P}$  defined by

$$\hat{P}(N(S)) = P(S \mid \langle W \rangle \notin S) \quad \text{for any set } S \text{ of game trees}$$

If  $P$  is a partisan probability distribution (see section 5), then so is  $\hat{P}$ . More precisely,  $\hat{P}$  obeys the normal play rule and allows  $n$  options to player A (resp. player A') with probability  $\hat{f}_n$  (resp.  $\hat{f}'_n$ ).

$$\hat{f}'_0 = -\frac{\beta'}{\alpha'} + \frac{1}{\alpha'} \sum_{i=0}^{\infty} f'_i \beta^i$$

$$\hat{f}'_n = \frac{1}{\alpha'} \sum_{i=0}^{\infty} f'_{n+i} \binom{n+i}{i} \alpha^n \beta^i$$

where  $\alpha = 1-\beta = 1-(1-b)p$  and  $\alpha' = 1-\beta' = 1-(1-b')p'$ .

Those expressions express the fact that a node now has  $n$  legal options for player A', whenever it used to have  $n+i$  options  $i$  of which where WINS for player A. Therefore:

$$\begin{aligned} \hat{F}'(z) &= -\frac{\beta'}{\alpha'} + \frac{1}{\alpha'} \sum_{n,i} f'_{n+i} \binom{n+i}{i} \beta^i \alpha^n z^n \\ &= -\frac{\beta'}{\alpha'} + \frac{1}{\alpha'} \sum_{m=0}^{\infty} f'_m \sum_{i=0}^m \binom{m}{i} (\alpha z)^{m-i} \beta^i \\ &= -\frac{\beta'}{\alpha'} + \frac{1}{\alpha'} \sum_{m=0}^{\infty} f'_m (\alpha z + \beta)^m \\ &= \frac{F'(\alpha z + \beta) - \beta'}{\alpha'} \end{aligned}$$

Similarly,

$$\hat{F}(z) = \frac{F(\alpha' z + \beta') - \beta}{\alpha}$$

So, any partisan statistics  $P$  has a normal equivalent  $\hat{P}$ , whose basic features are essentially identical.

Theorem C.1

The normal equivalent of a partisan probability distribution of internal structure  $(K, K')$  and external structure  $((p, p'), (b, b'))$  is the partisan probability distribution of internal structure  $(\hat{K}, \hat{K}')$  and external structure  $((0, 0), (\hat{b}, \hat{b}'))$ , where:

$$\begin{aligned}\hat{K}(z) &= \frac{K(\alpha'z + \beta') - K(\beta')}{1 - K(\beta')} & \hat{K}'(z) &= \frac{K'(\alpha z + \beta) - K'(\beta)}{1 - K'(\beta)} \\ \hat{b} &= b \frac{1 - K(\beta')}{\alpha} & \hat{b}' &= b' \frac{1 - K'(\beta)}{\alpha'} \\ \beta &= 1 - \alpha = (1 - b)p & \beta' &= 1 - \alpha' = (1 - b')p'\end{aligned}$$

Analytically, the relation between a statistic and its normal equivalent essentially reduces to the change of variables  $X = \alpha x + \beta$  and  $Y = \alpha'y + \beta'$ . For example, the relations  $y = \hat{G}'(x)$  and  $x = \hat{G}(y)$  can be rewritten  $Y = G'(X)$  and  $X = G(Y)$ .

The definition and analysis of the mapping M are parallel to that of N.

M associates a misère game  $M(GT)$  to any game  $GT$ , with the exception of the game  $\langle L \rangle$ .



### Definition C.2

The standard pruning transform M is defined as follows.

$$* M(\langle W \rangle) = \langle W \rangle$$

\* If GT is an internal node with at least one option which is a terminal LOSS, then its image M(GT) is a terminal WIN node.

\* If the root of GT is an internal node whose options ( $GT_1, GT_2, \dots, GT_n$ ) are not terminal LOSSes, then the options to the root of its image M(GT) are  $M(GT_1), M(GT_2), \dots, M(GT_n)$ .

To put it bluntly, a WIN in one move in GT becomes a terminal WIN in M(GT). Like N, M transforms a partisan probability distribution P into another partisan probability distribution  $\hat{P}$ .  $\hat{P}$  obeys the misère rule and has a tree structure  $(\hat{F}, \hat{F}')$  defined by

$$\hat{F}(z) = \frac{1}{\gamma} [F(Y'z) - F(Y') + Y]$$

$$\hat{F}'(z) = \frac{1}{\gamma'} [F'(Yz) - F'(Y) + Y']$$

where  $\gamma = 1 - (1-b)(1-p)$  and  $\gamma' = 1 - (1-b')(1-p')$ .

### Theorem C.2

The misère equivalent of a partisan probability distribution of internal structure  $(K, K')$ , and external structure  $((p, p'), (b, b'))$  is the partisan probability distribution of internal structure  $(\hat{K}, \hat{K}')$  and external structure  $((1, 1), (\hat{b}, \hat{b}'))$ , where

$$\begin{aligned}\hat{K}(z) &= \frac{K(Y'z)}{K(Y')} & \hat{K}'(z) &= \frac{K'(Yz)}{K'(Y)} \\ \hat{b} &= b \frac{K(Y')}{Y} & \hat{b}' &= b' \frac{K'(Y)}{Y'} \\ Y &= 1 - (1-b)(1-p) & Y' &= 1 - (1-b')(1-p')\end{aligned}$$

The pruning transforms N and M can be composed to yield other transforms that will not alter the nature of the game either. For example, the transform N o M changes the hypothetical original rules for chess, where the game stops with the capture of a king, into the conventional ones. [\*]

\* The author was taught chess, at the advanced age of seven, by an eight year old. His first game of chess did result in the capture of his own king.

## References

- [1] S. Asmussen and H. Hering, Branching Processes, Birkhauser, Boston (1983).
- [2] Avron Barr and Edward A. Feigenbaum, The Handbook of Artificial Intelligence (Volume 1), HeurisTech Press, Stanford, California (1981).
- [3] G. M. Baudet, "On the branching factor of the Alpha-Beta pruning algorithm," AI Journal Vol. 10, pp.173-199 (1978).
- [4] D.F. Beal, Recent progress in understanding minimax search, Queen Mary College, London EI 4NS, England (1983).
- [5] Elwyn R. Berkelamp, John H. Conway, and Richard K. Guy, Winning Ways, Academic Press, New York (1982).
- [6] Hans J. Berliner, "Experience in evaluation with BKG - A program that plays backgammon," IJCAI 5, pp.428-433 (1977).
- [7] Hans J. Berliner, "Search and knowledge," IJCAI 5, pp.975-979 (1977).
- [8] Hans J. Berliner, "The B\* search algorithm: A best-first proof procedure," CMU-CS-78-112, Department of Computer Science, Carnegie-Mellon University (1978).
- [9] C.L. Bouton, "Nim, a game with a complete mathematical theory," Ann. Math. Princeton (2),3, pp.35-39 (1902).
- [10] John H. Conway, On numbers and games, Academic Press, New York (1976).
- [11] Manfred Eigen and Ruthild Winkler, Laws of The Game, Alfred A. Knopf, New York (1981). Translation of Das Spiel.
- [12] Thomas S. Ferguson, "On sums of games with last player losing," International Journal of Game Theory 3, pp.159-167 (1974).
- [13] Thomas S. Ferguson, "Misère annihilation games," TR No. NSF-34, Statistics Center, MIT, Cambridge, Massachusetts (1981).

- [14] Aviezri S. Fraenkel, "From Nim to Go," Annals of Discrete Mathematics 6, pp.137-156, North-Holland (1980).
- [15] S. H. Fuller, J.G. Gashnig, and J.J. Gillogy, Analysis of the alpha-beta pruning algorithm, Dept. of Computer Science, Carnegie-Mellon University (1973).
- [16] Douglas R. Hofstadter, "Strange attractors [Metamagical Themes]," Scientific American, pp.22-43 (November 1981).
- [17] D.E. Knuth and R.M. Moore, "An analysis of alpha-beta pruning," AI Journal Vol. 6, pp.293-326 (1975).
- [18] Robert M. May, "Simple mathematical models with very complicated dynamics," Nature Vol. 261 (June 10, 1976).
- [19] Dana S. Nau, "The last player theorem," AI Journal Vol. 18(1), pp.53-65 (January 1982).
- [20] Dana S. Nau, An investigation of the causes of pathology in games, CS Dept, University of Maryland (1982).
- [21] Monroe Newborn, Computer Chess, Academic Press, New York (1975). ACM Monograph Series.
- [22] Nils J. Nilsson, Principles of Artificial Intelligence, Tioga, Palo-Alto, California (1980).
- [23] Judea Pearl, "Asymptotic properties of minimax trees and game-searching procedures," AI Journal Vol. 14(2), pp.113-138 (September 1980).
- [24] Judea Pearl, "On the nature of pathology in game searching.," AI Journal Vol. 20(4), pp.427-453 (July 1983).
- [25] Judea Pearl, Heuristics, Addison Wesley, Reading, Massachusetts (1984). (Manuscript).
- [26] I. Roizen and J. Pearl, "A minimax algorithm better than alpha-beta? Yes and no," AI Journal Vol. 21(1-2) (1983).
- [27] C. E. Shannon, "Programming a computer for playing chess," Philosophical Magazine (Series 7) Vol. 41, pp.256-275 (1950).
- [28] David Singmaster, Almost all games are first person games, 1974. Unpublished.
- [29] Michael Tarsi, "Optimal search on some game trees," Journal of the ACM Vol. 30(3), pp.389-396 (July 1983).

- [30] David E. Wilkins, "Using knowledge to control tree searching,"  
AI Journal Vol. 18(1), pp.1-51 (January 1982).

