# Exploring the Average Values of Boolean Functions via Asymptotics and Experimentation

Robin Pemantle[*]
Department of Mathematics
University of Pennsylvania
Philadelphia, PA 19104–6395
pemantle@math.upenn.edu

Mark Daniel Ward
Department of Mathematics
University of Pennsylvania
Philadelphia, PA 19104–6395
ward2@math.upenn.edu

**Abstract**

In recent years, there has been a great interest in studying Boolean functions by studying their analogous Boolean trees (with internal nodes labeled by Boolean gates; leaves viewed as inputs to the Boolean function). Many of these investigations consider Boolean functions of $n$ variables and $m$ leaves. Our study is related but has a quite different flavor.

We investigate the mean output $X_n$ of a Boolean function defined by a complete Boolean tree of depth $n$. Each internal node of such a tree is labeled with a Boolean gate, via $2^n - 1$ IID fair coin flips. The value of the input at each leaf can be simply fixed at 1/2, so the randomness of $X_n$ derives only from the selection of the gates at the internal nodes.

For each $n$, there are $2^{(2^n-1)}$ possible Boolean binary trees to consider, so we cannot expect to obtain a complete description of the probability distribution of $X_n$ for large $n$. Therefore, we perform a twofold investigation of the $X_n$, using both asymptotics and experiments. We prove that, with probability 1, $X_n \to 0$ or $X_n \to 1$. Then we directly compute the asymptotics of the first four moments of $X_n$. Writing $Z_n = X_n(1 - X_n)$, we also prove that $E(Z_n)$ and $E(Z_n^2)$ are both $\Theta(1/n)$. Finally, we utilize `C++` and a significant amount of computation and experimentation to obtain a more descriptive understanding of $X_n$ for small values of $n$ (say, $n \leq 100$).

## 1 Introduction.

We first outline the construction of a Boolean function using a binary tree. We utilize complete binary trees $T_n$ of depth $n$. The tree $T_n$ has $2^n$ leaves and $2^n - 1$ internal nodes. At each of the internal nodes, we place either an AND gate or an OR gate, with probability 1/2 each. Selection of the gates at distinct nodes is independent, so the gates are essentially chosen by IID fair coin flips. In other words, we uniformly select a vector consisting of $2^n - 1$ AND's and OR's, namely $\vec{g}_n \in \{\mathrm{AND}, \mathrm{OR}\}^{2^n-1}$. By labeling the internal nodes of a complete binary tree of depth $n$ with this collection $\vec{g}_n$ of $2^n - 1$ gates, we naturally define a random Boolean function $\phi_n(\vec{g}_n) : \{0,1\}^{2^n} \to \{0,1\}$. The leaves of the tree, say $i_1, i_2, \ldots, i_{2^n}$, are considered as the inputs to the Boolean function. The output at the root of the tree is viewed as the output of the Boolean function. Thus we write

$$\phi_n(\vec{g}_n)(i_1, i_2, \ldots, i_{2^n}) \in \{0,1\} .$$

for each $(2^n - 1)$-tuple $\vec{g}_n$ of gates and each $2^n$-tuple of inputs $i_1, i_2, \ldots, i_{2^n}$.

In this investigation, we are interested in studying the behavior of the random variable $X_n$, which denotes the *mean* output of $\phi_n(\vec{g}_n)$ on $2^n$ Boolean inputs. In other words,

$$X_n := \frac{1}{2^{2^n}} \sum_{i_1, i_2, \ldots, i_{2^n}} \phi_n(\vec{g}_n)(i_1, i_2, \ldots, i_{2^n}) .$$

We observe that $X_n$ is a random variable because the selection of the $2^n - 1$ gates in $\vec{g}_n$ is performed at random. Once the selection of the gates $\vec{g}_n$ is determined, then $X_n$ is completely determined, because $X_n$ is the average of all possible $2^{2^n}$ selections of inputs $i_1, i_2, \ldots, i_{2^n}$ to the Boolean tree described above. So the randomness of $X_n$ does not stem from a random choice of the inputs $i_1, i_2, \ldots, i_{2^n}$ at all; $X_n$'s randomness only depends on the random selection of gates at the internal nodes of the tree. Once the gates at the nodes are chosen, then we average over all possible inputs to the binary tree.

We will see in (2.6) below that $X_n$ converges in distribution to the measure $(1/2)\delta_0 + (1/2)\delta_1$ that gives mass one half each to 0 and 1. Intuitively, there will probably be a sufficient preponderance of AND gates to drive $X_n$ to 0 or a sufficient preponderance of OR gates to drive $X_n$ to 1. This is not too surprising (although it does not occur in the related model of [7], where the random tree of size $n$ has leaves at distance $\Theta(1)$ from the root). A more interesting question is how fast $X_n$ approaches the set $\{0,1\}$, and what this reveals about the structure of the random Boolean function $\phi_n(\vec{g}_n)$.

For each selection $\vec{g}_n$ of gates, we note that $\phi_n(\vec{g}_n)$ is a function with $2^n$ inputs. If the inputs

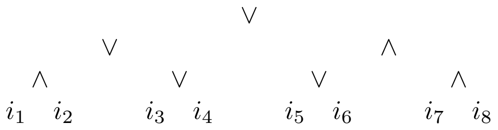
[*] Supported by NSF grant DMS-0401246.

$i_1, \dots, i_{j-1}, i_{j+1}, \dots, i_{2^n}$ are all fixed, then $\phi_n(\vec{g}_n)$ is a *linear* function of $i_j$. Since $i_j \in \{0,1\}$ for each $j$, then we conclude that $X_n$ can be computed easily, once the gates $\vec{g}_n$ are chosen, by simply taking $1/2$ as the value of each input $i_j$ to the Boolean function $\phi_n(\vec{g}_n)$. In other words, for each selection of $\vec{g}_n$, we have

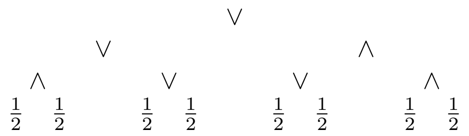
$$X_n = \phi_n(\vec{g}_n)(1/2, 1/2, \dots, 1/2)\,;$$

in this representation, it is perhaps easiest to see that the randomness of $X_n$ is due to the random selection of the gates in the $(2^n - 1)$-tuple $\vec{g}_n$.

An example is useful for clarification. Consider the selection of $\vec{g}_3$ given below in this tree of depth 3:



For complete trees of depth 3, we see that $X_3$ denotes the mean output of a Boolean random function with gates $\vec{g}_3$. If the choice of $\vec{g}_3$ is the one given above, this results in $X_3$ having the value 217/256. To see this, simply evaluate the tree:


∨
∨ ∧
∧ ∨ ∨ ∧
$\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$

Evaluating such a tree with inputs besides the familiar $\{0,1\}$ requires a bit of explanation. The evaluation of expressions such as $i_1 \wedge i_2$ is quite easy. This expression, for instance, evaluates to 1 if both $i_1$ and $i_2$ have the value 1; otherwise, the expression evaluates to 0. Unfortunately, this evaluation is useful only for $i_1, i_2 \in \{0,1\}$. So we instead use the following equivalent interpretation (which is quite standard). We write

$$\begin{aligned} i_1 \wedge i_2 &:= i_1 i_2 \\ i_1 \vee i_2 &:= 1 - (1 - i_1)(1 - i_2)\,. \end{aligned} \tag{1.1}$$

This interpretation has the benefit that $i_1$ and $i_2$ can be any real numbers; in particular, they can each be set to the value $1/2$.

The trees $T_n$ are naturally embedded as increasing rooted subtrees of the infinite binary tree $T$. All the variables $X_n$ may be constructed on a single probability space $(\Omega, \mathcal{F}, P)$ on which are defined variables $\{G(v) : v \in V(T)\}$ taking values AND and OR independently with probability 1/2 each. If $T'$ is any (possibly random) rooted subtree of $T$, let $\vec{g}_{T'}$ be the Boolean function with inputs at the leaves of $T'$ defined by having the gate $G(v)$ at each internal node $v$ of $T'$. Let $X_{T'}$ denote the mean of $T'$ if all input vectors are equally likely. Thus $\phi_n(\vec{g}_n) = \vec{g}_{T_n}$ and $X_{T_n} = X_n$. Let $X_{T'}(v)$ denote the mean value of the Boolean function computed by the subtree of $T'$ rooted at $v$; we may omit $T'$ from the notation when it is understood. Another consequence of linearity is that the values $X_{T'}(v)$ are determined by a recursion. That is, if $v$ is a leaf of $T'$ then $X_{T'}(v) = 1/2$ by definition, while for an internal node $v$ of $T'$ having children $w$ and $w'$,

$$X_{T'}(v) = \Psi(X_{T'}(w), X_{T'}(w'), G(v)) \tag{1.2}$$

where

$$\Psi(x, y, \eta) := \begin{cases} xy & \text{if } \eta = \text{ AND}\,, \\ 1 - (1-x)(1-y) & \text{if } \eta = \text{ OR}\,. \end{cases}$$

Evaluating a binary tree with inputs of 1/2 at each of the leaves yields the value of $X_n$ for each particular selection of gates. Considering all possible selections of gates, however, is computationally infeasible for even small trees of small depth. For only the smallest values of $n$, say $n \le 5$, can we possible hope to compute $X_n$ for all of the possible choices of gates. Therefore, for medium sized values of $n$, say $n \le 20$, we can readily compute the value of $X_n$ for one particular selection of gates for one complete tree of depth $n$, but we cannot hope to compute $X_n$ for every selection of gates. Therefore, we simply compute $X_n$ on a large number of trees, but we cannot perform an exhaustive investigation of all trees and their associated Boolean functions. For large values of $n$, say $n \ge 30$, it becomes computationally intractable to even compute the value of $X_n$ for one particular selection of gates on a complete binary tree of depth $n$. In such cases, we must discriminately choose which gates to evaluate, because we cannot possibly hope to evaluate them all.

In such cases, where we want to approximate the value of $X_n$ on a complete tree of depth $n$, but where we cannot hope to evaluate all gates of the Boolean tree, we consider a growing tree. We begin simply with the root of a Boolean binary tree. At every stage, we select one leaf of the tree and change it into an internal leaf, by giving it two children and a Boolean gate. Which leaves should be transformed into parent nodes first? We utilize the concept of sensitivity of a leaf to select the next leaf to transform. The leaves that are the most sensitive, i.e., that have the largest potential effect on the evaluation of $X_n$, should be first. We now formalize this notion.

Let $T'$ be a finite subtree of $T$ and let $L$ be a leaf of $T'$. Let $\vec{g}_{T',L+}$ be the Boolean function with inputs at all leaves of $T'$ except $L$, computed by evaluating $g_{T'}$ with the input at $L$ set equal to 1. Let $\vec{g}_{T',L-}$ be analogous but with the input at $L$ set equal to 0. Let

$X_{T',L+}$ and $X_{T',L-}$ denote the respective means of the functions $\vec{g}_{T',L+}$ and $\vec{g}_{T',L-}$. Then we formally define the sensitivity of the leaf $L$ in $T'$ as

$$S(L,T') = X_{T',L+} - X_{T',L-}\,.$$

Another description of $S(L,T')$ is quite useful when computing the sensitivities of the leaves.

We label the root node of $T'$ as $v_0$. For a leaf $L$ at depth $k$ in $T'$, we write $v_0, v_1, v_2, \ldots, v_k = L$ to describe the path within the tree, from the root node, to the leaf $L$. For $i \geq 1$, we note that $v_{i-1}$ has two children, namely, $v_i$ and one other child, which we refer to as $w_i$. Thus $v_i$ and $w_i$ are distinct nodes at level $i$ with the same parent; such nodes are frequently referred to as siblings.

Lemma 1.1.

$$\begin{aligned} S(v_k,T') \quad &= \quad \prod_{i=0}^{k-1} \big(\llbracket G(v_i) = \text{AND}\rrbracket X(w_{i+1}) \\ &\qquad + \llbracket G(v_i) = \text{OR}\rrbracket (1 - X(w_{i+1}))\big) \end{aligned} \tag{1.3}$$

*where the Iverson notation $\llbracket A \rrbracket$ is 1 if event A holds and is 0 otherwise.*

*Proof.* Induct on $T'$. When $T'$ is the single vertex $v_0 = v_k = L$ then by definition $g_{T',L+} \equiv 1$ and $g_{T',L-} \equiv 0$, so $S(L,T') = 1$ which is equal to the empty product in (1.3). If not, then assume the lemma is true for the subtree $T'(v_1)$:

$$\begin{aligned} S(L,T'(v_1)) \quad &= \quad \prod_{i=1}^{k-1} \big(\llbracket G(v_i) = \text{AND}\rrbracket X(w_{i+1}) \\ &\qquad + \llbracket G(v_i) = \text{OR}\rrbracket (1 - X(w_{i+1}))\big)\,. \end{aligned}$$

By definition, $S(L,T')$ is the difference of the means of $\vec{g}_{T',L+}$ and $\vec{g}_{T',L-}$. By independence, the mean of $\vec{g}_{T',L+}$ is $\vec{g}_{T'(v_1),L+} X(w_1)$ if $G(v_0) = \text{AND}$, and $1 - (1 - \vec{g}_{T'(v_1),L+})(1 - X(w_1)) = \vec{g}_{T'(v_1),L+}(1 - X(w_1)) + X(w_1)$ if $G(v_0) = \text{OR}$. Subtracting the analogous representation of the mean of $\vec{g}_{T',L-}$ (just replace each $L+$ in the previous sentence with $L-$), it follows that $S(L,T') = S(L,T'(v_1)) \cdot \xi$, where $\xi = X(w_1)$ if $G(v_0) = \text{AND}$ and $1 - X(w_1)$ if $G(v_0) = \text{OR}$, completing the induction.

We developed a `C++` program to investigate the growth of Boolean binary trees, using the sensitivity of the leaves as a guide for which subtrees to explore first. The program is completely adaptive, according to the sensitivities of the leaves. At each stage of the execution of the program, the most sensitive leaf is chosen, using the definition of sensitivity described above. If several leaves have the same sensitivity, the program selects one of the candidate leaves uniformly at random; sometimes the candidate leaves are at different levels, so this is an important subtlety in the implementation of the program. Once a leaf $L$ is selected to be updated, we consider the path $v_0, v_1, \ldots, v_k = L$ from the root of the tree to the leaf. Only the $X$-values $X(v_0), X(v_1), \ldots, X(v_k)$ must be updated; this is extremely efficient in terms of the computation required, because at most $n$ nodes are found on the path from the root to the leaf. The sensitivities of every leaf in the tree must be updated afterwards. We note that if $v'_0, v'_1, \ldots, v'_k = L'$ denotes another leaf, then the sensitivity of $L'$ is exactly

$$\begin{aligned} &\prod_{i=0}^{k-1} \big(\llbracket G(v'_i) = \text{AND}\rrbracket X(w'_{i+1}) \\ &\quad + \llbracket G(v'_i) = \text{OR}\rrbracket (1 - X(w'_{i+1}))\big)\,. \end{aligned} \tag{1.4}$$

A subtlety here is the observation that only $X(w_0), X(w_1), \ldots, X(w_j)$ were changed at this stage in the growth of the tree. For some value of $j$ (which is, with high probability, quite small), we note that $L$ and $L'$ have common ancestors $v_i = v'_i$ for all $i \leq j$, but $v_i \neq v'_i$ for all $i > j$. Thus $w_i = w'_i$ for all $i \leq j$, but $w_i \neq w'_i$ for all $i > j$. Therefore, only the $X$-values $X(w'_1), \ldots, X(w'_j)$ need to be updated in the sensitivity of $L'$, as written in (1.4). In other words, the only $X(w'_i)$-values that need updating are those for which $v_i = v'_i$ is a common ancestor of both $L$ and $L'$.

We wrote several `C++` programs to perform the computations in this project. Some sample output from the programs is presented graphically at the end of this paper.

We have computed millions and millions of values of $X_n$ for various values of $n$. For instance, when $n = 15$, we are able to compute approximately 30 values of $X_n$ per second on a 1.42 GHz Power Macintosh G4 machine. We have built a large database that archives all of the output from these investigations. It has grown so large that it is unwieldy to distribute all of it publicly on the Internet, but we summarize some of the results of our computations at the end of this report.

## 2 Main results

We were inspired to pursue an analysis of $X_n$ because of Gardy and Woods' intriguing study [7], in which various measures on Boolean functions are analyzed. Gardy and Woods consider trees chosen uniformly among all subbinary trees with $n$ leaves; they also place randomly assigned logical gates at the internal nodes. We note that a uniformly chosen tree with $n$ leaves is stringy. The typical random function produced in this way is therefore dominated by the $\Theta(1)$ many inputs at leaves of distance $\Theta(1)$ from the root. Their model is natural for some purposes, but we are interested in considering the model in which, as $n \to \infty$, the distance from the root to the boundary goes to infinity. For this reason, we consider the simplest such model, namely, the complete binary tree. The typical behavior of a random Boolean function produced by a complete binary tree turns out to be interesting but in some ways elusive.

Besides the analysis contained in [7], we note that many attributes of Boolean functions, binary Boolean trees, and tree recurrences have been analyzed recently. For instance, consider [1], [2], [3], [6], [8], [10], [11], [12], and [13]. These papers constitute a starting point from which readers can delve further into the literature.

We recall that $X_n$ is the mean output of a Boolean function defined by a complete Boolean tree of depth $n$. In this report, we prove the following facts about $X_n$.

Theorem 2.1. *The sequence $\{X_n\}$ is a Martingale. With probability 1, the limit $\lim_{n\to X_n}$ exists and is either 0 or 1. The moments of $X_n$ may all be computed recursively. In particular, the first four moments of $X_n$ are*

$$\begin{aligned} E(X_n) &= \frac{1}{2} \\ E(X_n^2) &= \frac{1}{2} - \frac{1}{n} + O\left(\frac{\log n}{n^2}\right) \\ E(X_n^3) &= \frac{1}{2} - \frac{3}{2n} + O\left(\frac{\log n}{n^2}\right) \\ E(X_n^4) &= \frac{1}{2} + \frac{\alpha - 2}{n} + O\left(\frac{\log n}{n^2}\right) \qquad (2.5) \end{aligned}$$

*where $\alpha = \frac{\sqrt{7}-1}{2}$. To understand the rate at which a variable with symmetric distribution on $[0,1]$ converges to $\{0,1\}$, it is natural to analyze the moments of $Z_n := X_n(1 - X_n)$. We have*

$$E(Z_n) = \frac{1}{n + O(\log n)}\,, \qquad (2.6)$$

$$E(Z_n^2) \sim \frac{\alpha}{n}\,. \qquad (2.7)$$

*It follows that, for some $a > 0$, $P(Z_n \geq a) = \Theta(1/n)$.*

Left open is whether the rest of the time $Z_n$ is typically of order $1/n$ or of some smaller order.

Just as the right $1/n$-tail of $Z_n$ is larger than one might initially expect, it is also not hard to show that the left $1/n$-tail of $Z_n$ is quite small.

Proposition 2.1. *There are $c, c' > 0$ such that $P(Z_n < \exp(-cn^2)) > c'/n$.*

We believe in fact that the distribution of $\log Z_n$ is spread over an interval of increasing size as $n \to \infty$. Perhaps, for instance, $\sqrt{\log Z_n}/n$ has a nondegenerate distributional limit.

We point out that there are issues in effective simulation that are bound up with theoretical analyses of the problem. In particular, exact simulation of $Z_n$ (the study of $Z_n$ and $X_n$ is basically interchangeable) requires a time that is exponential in $n$. Thus, for example, we cannot even obtain one sample of $Z_{100}$ (just one sample of $Z_{100}$ requires the generation of $2^{100} - 1$ Boolean gates). We can only hope to obtain exact samples of $Z_n$ for medium-sized $n$. For larger $n$, say $n \geq 30$, our remedy is an extensive experimental analysis of $Z_n$ by repeatedly approximating $Z_n$; we do this by exploring only nodes of the tree that one expects to have high impact on the value of $Z_n$. At each stage in the growth of the tree, there is a well defined most sensitive remaining node (there may be ties); therefore, one may define a greedy search algorithm which always looks next at this node. Revealing the gate will reduce the variance by the most. If one can then compute how close one is to $X_n$ then one will know how far to go in order to simulate a pick from $X_n$ with the desired precision. If, further, one can analyze the growth of the exploration tree, then one will know how long it takes to simulate $X_n$, and this will have implications directly on the distribution of $X_n$. For example, if $X_n$ is typically well approximated by a tree of depth $m < n$, then the distributions of $X_n$ and $X_m$ are close and, if $m = o(n)$, this precludes a limit law with $n$ in the denominator.

Ample data generated by our various **C++** programs for studying the behavior of $X_n$ when $n$ is small (say, $n \leq 100$) can be obtained from the authors. Our files of data are too large to distribute on the internet at present (we have hundreds of megabytes of files, containing millions of samples of various $X_n$).

At the present time, it suffices to present a few graphs of sample data concerning $X_n$ at the end of the paper. In particular, we give plots of values related to possible limit laws for $X_{15}$ and $X_{20}$, using numerical data from millions of samples of $X_{15}$ and $X_{20}$.

## 3 Analysis and Proofs.

We establish the fact that $\{X_n\}$ is a martingale. We also derive the first four moments of $X_n$. Using a similar methodology, one can set up similar recurrences and use analogous arguments to derive any of the moments of $X_n$.

By a *sampling scheme* we mean a rule for producing a sequence of vertices $y_1, y_2, \ldots$ such that for each $k$ the vertices $\{y_1, \ldots, y_k\}$ span a rooted subtree $W_k$ of $T$ and each $y_{k+1}$ is a function of $G(y_1), \ldots, G(y_k)$. Associated with each such rule is the $\sigma$-field $\mathcal{F}^{(k)} := \sigma(G(y_1), \ldots, G(y_k))$ of the first $k$ gates one looks at.

Lemma 3.1. *The sequence $\{X_{W_k}\}$ is a martingale with respect to $\{\mathcal{F}^{(k)}\}$. In particular, it follows from the breadth-first sampling scheme that the sequence $\{X_n\}$ is a martingale with respect to the filtration $\{\mathcal{F}_n\} := \sigma(G(v) : v$ is an internal node of $T_n)$.*

*Proof.* To see that $\{X_{W_k}\}$ is a martingale, observe that, conditional on $\mathcal{F}^{(k)}$, the vertex $y_{k+1}$ is known and its output in the tree $W_{k+1}$ has mean 1/2. By linearity, $E(X_{W_{k+1}}|\mathcal{F}^{(k)})$ is the mean output at the root of $W_k$, with the input at $y_{k+1}$ set to 1/2; however, the mean may be computed by setting all the inputs to 1/2, so this is equal to $X_{W_k}$.

Corollary 3.1. *With probability 1, $\lim_{n\to\infty} X_n$ exists.*

*Proof.* Since $0 < X_n < 1$ for each $n$, we have $E(|X_n|) \leq 1$ for all $n$. By Lemma 3.1, we know that $\{X_n\}$ is a martingale. Thus, the corollary follows immediately by the Martingale Convergence Theorem (see [4], [5]).

We now evaluate moments of $X_n$.

Lemma 3.2.
$$E(X_n) = 1/2\,.$$

*Proof.* Reversing each gate and each input reverses the output but preserves the measure on functions of $\vec{g}_n$. Thus $1 - X_n$ and $X_n$ have the same distribution.

We use the following Lemma to aid in the proof of Theorem 3.2. If we define $Z_n = X_n(1 - X_n)$, then we make the following observations.

Lemma 3.3.

1. *$E(X_n^2)$ increases to the limiting value of 1/2.*
2. *The rate of convergence is given by*
$$E(X_n^2) = \frac{1}{2} - \frac{1}{n} + O\left(\frac{\log n}{n^2}\right)\,.$$
3. *Asymptotics for $Z_n$ are given by*
$$E(Z_n) := E(X_n(1 - X_n)) = \frac{1}{n} + O\left(\frac{\log n}{n^2}\right)\,.$$

*Proof.* That $E(X_n^2)$ increases follows from $\{X_n\}$ being a martingale. To find the limit, we establish a recurrence for $E(X_n^2)$. When computing $X_{n+1}$, we let $X_n' := X_{T_{n+1}}(v_1)$ and $X_n'' := X_{T_{n+1}}(w_1)$ denote the outputs of the Boolean functions for the left and right subtrees of the root node; note that $X_n'$ and $X_n''$ are independent, and each is distributed as $X_n$. Taking expectations in (1.2), we have

$$\begin{aligned} E(X_{n+1}^2) &= \frac{1}{2}E((X_n'X_n'')^2) \\ &\quad + \frac{1}{2}E((1 - (1 - X_n')(1 - X_n''))^2)\,. \end{aligned} \tag{3.8}$$

The variables $X_n'$, $X_n''$, $1 - X_n'$, and $1 - X_n''$ all have the same distribution. Together with independence of $X_n'$ and $X_n''$ this gives

$$E(X_{n+1}^2) = E(X_n^2)^2 + \frac{1}{4}\,. \tag{3.9}$$

Since $E(X_n^2)$ increases and is bounded above by 1, then a limiting value exists; we take a limit on both sides of (3.9) to obtain

$$\lim_{n\to\infty} E(X_{n+1}^2) = (\lim_{n\to\infty} E(X_n^2))^2 + 1/4\,. \tag{3.10}$$

Thus $\lim\limits_{n\to\infty} E(X_n^2) = 1/2$, which completes the proof of the first statement of the Lemma.

Now we observe

$$\begin{aligned} E(Z_n) &= E(X_n(1 - X_n)) \\ &= E(X_n) - E(X_n^2) \\ &= \frac{1}{2} - E(X_n^2)\,. \end{aligned} \tag{3.11}$$

Thus $E(X_n^2) = \frac{1}{2} - E(Z_n)$. From (3.9), it follows immediately that

$$\frac{1}{2} - E(Z_{n+1}) = \left(\frac{1}{2} - E(Z_n)\right)^2 + 1/4\,;$$

after simplifying, we obtain

$$E(Z_{n+1}) = E(Z_n) - E(Z_n)^2\,.$$

For ease of notation, we write $a_n = E(Z_n)$. So we have $a_{n+1} = a_n - a_n^2$. Then we write $b_n = 1/a_n$, and we compute

$$\begin{aligned} b_{n+1} &= \frac{1}{a_n - a_n^2} \\ &= \frac{b_n^2}{b_n - 1} \\ &= b_n + 1 + \frac{1}{b_n - 1}\,. \end{aligned} \tag{3.12}$$

Iterating this yields

$$b_{n+1} = b_1 + n + \sum_{k=1}^{n} \frac{1}{b_k - 1}\,. \tag{3.13}$$

From (3.13), we observe that $b_{n+1} > n$, so $b_k > k-1$ for all $k$. Thus, the summation in (3.13) can be bounded by writing

$$\begin{aligned}\sum_{k=1}^{n} \frac{1}{b_k - 1} &= \frac{1}{b_1 - 1} + \frac{1}{b_2 - 1} + \sum_{k=3}^{n} \frac{1}{b_k - 1} \\ &\leq \frac{1}{\frac{16}{3} - 1} + \frac{1}{\frac{256}{39} - 1} + \sum_{k=3}^{n} \frac{1}{(k-1) - 1} \\ &= O(\log n)\,.\end{aligned}$$

Returning to (3.13), we conclude that

$$b_n = n + O(\log n)\,.$$

Again using (3.13), it follows that

$$\begin{aligned}b_{n+1} &= b_1 + n + \sum_{k=1}^{n} \frac{1}{n + O(\log n)} \\ &= b_n = n + \log n + O(1)\end{aligned}$$

and we conclude that

$$E(Z_n) = a_n = \frac{1}{n} + O\left(\frac{\log n}{n^2}\right)\,.$$

This proves the final sentence of the lemma. All that remains to show is $E(X_n^2) = 1/2 - 1/n + O\left(\frac{\log n}{n^2}\right)$, but this follows immediately from $E(X_n^2) = 1/2 - E(Z_n)$.

We recall from Corollary 3.1 that $\lim_{n\to\infty} X_n$ exists with probability 1. So we define $X := \lim_{n\to\infty} X_n$. Using Lemma 3.3, we have the following result about the limiting behavior of $X_n$.

Corollary 3.2.

$$P(X = 0) = P(X = 1) = \frac{1}{2}\,.$$

*Proof.* By bounded convergence, we have $E(X^2) = \lim_{n\to\infty} E(X_n^2) = 1/2$. Since $X \in [0,1]$ is symmetric around 1/2, this implies $X = 0$ or 1 with probability 1/2 each.

The next two lemmas compute the remaining moments in Theorem 2.1.

Lemma 3.4. *The third moment of $X_n$ is given by*

$$E(X_n^3) = \frac{1}{2} - \frac{3}{2n} + O\left(\frac{\log n}{n^2}\right)\,.$$

*Proof.* As in Lemma 3.3, we establish a recurrence for $E(X_n^3)$. When computing $X_{n+1}$, we again write $X'_n$ and $X''_n$ to denote the output of the Boolean functions for the left and right subtrees of the root node, which are independent. Then we compute

$$\begin{aligned}E(X_{n+1}^3) &= \frac{1}{2}E((X'_n X''_n)^3) \\ &\quad + \frac{1}{2}E((1-(1-X'_n)(1-X''_n))^3)\,.\end{aligned} \tag{3.14}$$

We once again use the fact that $X'_n$, $X''_n$, $1-X'_n$, and $1-X''_n$ share a common distribution. Thus,

$$\begin{aligned}E(X_{n+1}^3) &= \frac{1}{2}E(X_n^3)^2 + \frac{1}{2} - \frac{3}{2}E(X_n)^2 \\ &\quad + \frac{3}{2}E(X_n^2)^2 - \frac{1}{2}E(X_n^3)^2 \\ &= \frac{3}{2}E(X_n^2)^2 + \frac{1}{8}\,.\end{aligned} \tag{3.15}$$

Recall from (3.9) that

$$E(X_{n+1}^2) = E(X_n^2)^2 + \frac{1}{4}\,. \tag{3.16}$$

Plugging this result into (3.15) yields

$$\begin{aligned}E(X_{n+1}^3) &= \frac{3}{2}\left(E(X_{n+1}^2) - \frac{1}{4}\right) + \frac{1}{8} \\ &= \frac{3}{2}E(X_{n+1}^2) - \frac{1}{4}\end{aligned} \tag{3.17}$$

and by Lemma 3.3, we conclude that

$$E(X_n^3) = \frac{1}{8} - \frac{3}{4n} + O\left(\frac{\log n}{n^2}\right)\,. \tag{3.18}$$

This establishes the lemma.

Using the lemmas above, we now establish the following asymptotics for $E(Z_n^2)$.

Lemma 3.5. *The second moment of $Z_n^2$ decays as $E(Z_n^2) \sim \frac{\alpha}{n}$ where $\alpha = \frac{\sqrt{7}-1}{2} \approx .82$.*

*Proof.* As in several of the above lemmas, we observe that

$$\begin{aligned}E(X_{n+1}^4) &= \frac{1}{2}E((X'_n X''_n)^4) \\ &\quad + \frac{1}{2}E((1-(1-X'_n)(1-X''_n))^4)\,.\end{aligned} \tag{3.19}$$

Simplifying via the same method as in the lemmas and using the results established in Lemmas 3.3 and 3.4, it follows that

$$E(X_{n+1}^4) = E(X_n^4)^2 - \frac{3}{2}E(X_n^2)^2 + \frac{3}{2}E(X_n^2) - \frac{1}{8}\,. \tag{3.20}$$

For ease of notation, we define

$$h_n := \frac{1}{2} - E(X_n^4) \tag{3.21}$$

and

$$d_n := \frac{1}{2} - E(X_n^2)\,. \tag{3.22}$$

It follows from (3.20) that

$$h_n = h_{n-1} - h_{n-1}^2 + \frac{3}{2}d_n^2\,. \tag{3.23}$$

The proof of the lemma is finished by the following identity.

Lemma 3.6. *If $d_n \sim 1/n$ and $h_n$ are positive numbers satisfying (3.23) then $h_n \sim \alpha/n$ for $\alpha = (\sqrt{7}-1)/2$.*

*Proof.* Let $u_n = nh_n$. The recursion (3.23) becomes

$$\begin{aligned} u_n &= \frac{n}{n-1}u_{n-1} - \frac{n}{(n-1)^2}u_{n-1}^2 + \frac{3/2+o(1)}{n} \\ &= u_{n-1} + \frac{\left[u_{n-1} - u_{n-1}^2 + \frac{3}{2} + O\left(\frac{u_{n-1}}{n}\right)\right]}{n-1}\,. \end{aligned}$$

One may easily verify that $u_n/n \to 0$, which implies

$$u_n - u_{n-1} = n^{-1}[f(u_{n-1}) + o(1)]\,. \tag{3.24}$$

Checking that $x - x^2 + 3/2$ is positive on $(0, \alpha)$ and negative on $(\alpha, \infty)$, we then see that $u_{n+1} > u_n$ when $u_n \in (0, \alpha - o(1))$ and $u_{n+1} < u_n$ when $u_n \in (\alpha + o(1), \infty)$, so $u_n$ converges. Convergence to something other than $\alpha$ is not possible because in that case eventually $|f(u_n)| > \varepsilon$ and divergence of the harmonic sum in (3.24) would contradict convergence of $u_n$.

We complete the moment computations with:

Corollary 3.3.

$$E(X_n^4) = \frac{1}{2} + \frac{\alpha-2}{n} + O\left(\frac{\log n}{n^2}\right)$$

*where $\alpha = \frac{\sqrt{7}-1}{2}$.*

*Proof.* We note that $E(X_n^4) = E(Z_n^2) - E(X_n^2) + 2E(X_n^3)$, and then the corollary follows immediately from Lemmas 3.3, 3.4, and 3.5.

Finally, to derive the last statement in Theorem 2.1, observe that greatest possible second moment for a random variable in $[0, a]$ with mean $\mu$ is $a\mu$. Thus,

$$\begin{aligned} \frac{\alpha+o(1)}{n} &= E(Z_n^2) \\ &= E(Z_n^2)[\![Z_n \geq a]\!] + E(Z_n[\![Z_n < a]\!])^2 \\ &\leq (\sup_n Z_n^2)P(Z_n \geq a) + aEZ_n[\![Z_n < a]\!] \\ &\leq \frac{1}{16}P(Z_n \geq a) + \frac{a+o(1)}{n}\,, \end{aligned}$$

and hence

$$P(Z_n > a) \geq 16\frac{\alpha - a + o(1)}{n}\,.$$

This finishes the proof of Theorem 2.1 for any value $a < \alpha$.

Now we prove Proposition 2.1, namely, there are $c, c' > 0$ such that $P(Z_n < \exp(-cn^2)) > c'/n$.

*Proof of Proposition 2.1.* Let $T' \subseteq T$ be the subtree whose vertices are the vertices of $T$ that can be reached from the root by a path not containing any OR gate (if there is an OR gate at the root then $T'$ is empty). Denote the size and depth of $T'$ by $|T'|$ and $d(T')$, respectively. The event $A_n$ that $d(T') < n$ and $|T'| > n^2$ is well known to have probability asymptotic to $Cn^{-1}$ for some constant $C > 0$; this follows, for example, from the convergence of $n$ times the law of the path that circumnavigates the tree to Brownian excursion measure [9].

Let $\mathcal{F} = \sigma(T')$ be the information contained in the value of the random tree $T'$. Let $S_n$ be the set of vertices in $T_n$ adjacent to $T'$ but not in $T'$. Since a subtree of $T$ with $k$ vertices has $k+1$ neighbors, we see that on $A_n$, the set $S_n$ satisfies $s := |S_n| > n^2$. Conditional on $\mathcal{F}$, the $s$ subtrees from vertices in $S_n$ are independent and distributed exactly as the gates of $T$ except that the root is always an OR. Consequently, on $A_n$, the output at any vertex of $S_n$ has conditional mean 3/4 given $\mathcal{F}$, since all we know about the gates of this subtree is that there is an OR at the root. Furthermore, $E(X_n \,|\, \mathcal{F}) = (3/4)^s$ on the event $A_n$, since the root outputs a 1 if and only if all vertices in $S_n$ output a 1. The result now follows from $P(A_n) \sim Cn^{-1}$ and $E(X_n \,|\, \mathcal{F}) < (3/4)^{n^2}$ on $A_n$, with any $c < \log(4/3)$ and $c' < C$.

## 4 Further Discussion and Experimental Data

The purpose of the sensitivity-first sampling scheme is to sample approximately from the distribution of $X_n$ by sampling the variable $Y_k := X_{W_k}$ in the sampling scheme $y_1, y_2, \ldots$ which always chooses the leaf of greatest sensitivity. More precisely, when trying to sample from $X_n$, we partition the leaves of $W_k$ into $A_k \cup B_k$ where $A_k$ is the set of leaves of $W_k$ at level $n$ and $B_k$ is the set of leaves of $W_k$ at levels less than $n$; our sampling scheme then designates $y_{k+1}$ to maximize $S(y_{k+1}, W_k)$ over $y_{k+1} \in B_k$. This sampling scheme will halt when $k = 2^n$ and produce $Y_k = X_n$, but our hope is that $Y_k$ is close to $X_n$ for $k$ much less than $n$, for example a polynomial in $\log n$.

We cannot prove this, though we have some shaky evidence. The reason the evidence is shaky is that we have tabulated how great $k$ must be in order to satisfy criteria appearing to give $Y_k$ near to $X_n$ but cannot prove that $Y_k$ is actually close to $X_n$. We conclude the theoretical discussion with some results giving bounds on the distance from $Y_k$ to $X_n$.

We will be applying these bounds knowing $\mathcal{F}^{(k)}$ but not $\mathcal{F}_n$, so we want bounds measurable with respect to $\mathcal{F}^{(k)}$. A crude upper bound is:

Proposition 4.1.

$$E(|Y_k - X_n| \,|\, \mathcal{F}^{(k)}) \le \sum_{y \in B_k} S(y, T_k)\,. \tag{4.25}$$

*Proof.* Let $X_n^*$ be the (random) mean of the random Boolean function obtained from $\vec{g}_n$ by fixing inputs (at random) at vertices in $B_k$. Then $X_n$ is a conditional expectation of $X_n^*$ so, conditional on $\mathcal{F}^{(k)}$, we know that $E(|X_n - Y_k| \,|\, \mathcal{F}^{(k)}) \le E(|X_n^* - Y_k| \,|\, \mathcal{F}^{(k)})$.

For $y \in B_k$, fixing the inputs at leaves of $T_n$ below $y$ as independent fair coin flips produces an output at $y$; for purposes of computing $X_n^*$, we may as well simply fix that output, which will be a 0 half the time and a 1 half the time, independent of nodes not in $T(y)$. Thus another way to compute $X_n^*$ is to fix inputs (independent fair coin flips) at all the leaves of $T_k$ not already at level $n$. Enumerate these as $z_1, \ldots, z_r$, and let $M_j$ denote the mean of the Boolean function obtained from $\vec{g}_{T_k}$ by fixing inputs at $z_1, \ldots, z_k$. The triangle inequality gives

$$E(|X_n^* - Y_k| \,|\, \mathcal{F}^{(k)}) \le \sum_{j=1}^{r} E(|M_j - M_{j-1}| \,|\, \mathcal{F}^{(k)})$$

which is equal to $\sum_{j=1}^{r} (1/2) S(z_j, T_k)$ because the sensitivity at $z_j$ as $z_1, \ldots, z_{j-1}$ is revealed is itself a martingale. This proves (4.25).

Based on this, a reasonable time to halt the algorithm and output $Y_k$ would be when the right-hand side of (4.25) is much smaller than the same sum over leaves of $T_k$ that are at level $n$. Unfortunately, based on our preliminary experiments growing trees with **C++** according to the sensitivities of the leaves, this does not seem to happen until too much of $T_n$ is explored to be efficient. However, the $L^1$ sum in (4.25) is probably an overestimate of how much $Y_k$ will change on the way to evaluating $X_n$. In particular, since $\{Y_k\}$ is a martingale, one might expect that summing in $L^2$ yields sharper estimates.

The incremental variance of the martingale $\{Y_k\}$ is given by the squared sensitivities:

$$E((Y_{k+1} - Y_k)^2 \,|\, \mathcal{F}^{(k)}) = (1/4) S(y_{n+1}, T_k)^2\,.$$

We would like to conclude that the $L^2$ difference between $Y_k$ and $X_n$ is given by the sum of $S(y, T_k)^2$ over leaves $y$ of $T_k$ that are not already at level $n$, but the problem is that, with $z_j, W_j$ as above, it is no longer true that $E(W_{j+1} - W_j)^2 = (1/4) S(z_j, T_k)^2$. This is because the sensitivity at $z_j$ is a submartingale as the gates at $z_1, \ldots, z_{j-1}$ are revealed. We conjecture, however, that

$$E((Y_k - X_n)^2 \,|\, \mathcal{F}^{(k)}) \le C \sum_y S(y, T_k)^2$$

for some constant $C$.

Since we cannot rigorously prove a fast stopping rule that yields reasonable sample values of $X_n$, we have reliable samples of $X_n$ only for $n \le 20$. Various graphs concerning the distribution of $X_{15}$ and $X_{20}$ are given below.

If we write $p = P(X_{15} \le x)$, then the following chart gives the values of $p$ and the analogous $x$ value. The data is based upon four million samples of $X_{15}$.

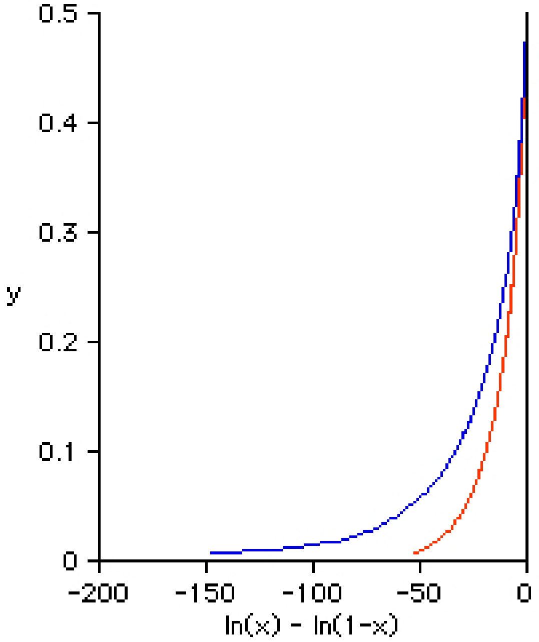
We also give similar data for $X_{20}$, based on 800,000 samples.

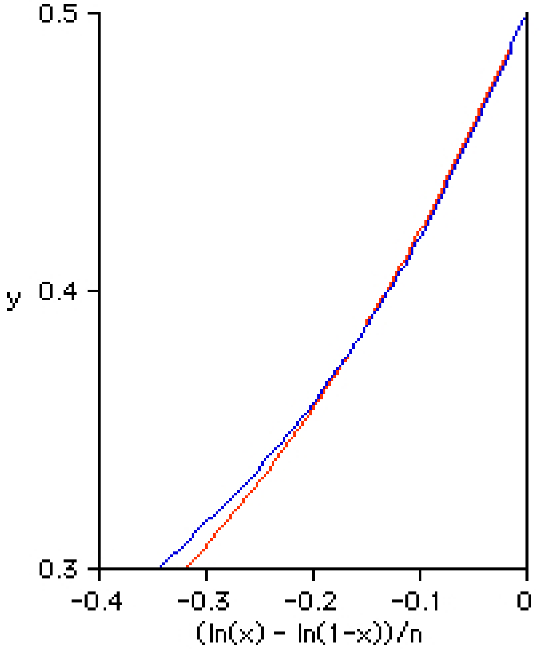
We emphasize that each sample of $X_{15}$ and $X_{20}$ was produced by computing a complete Boolean binary tree of depth 15 and 20, respectively.

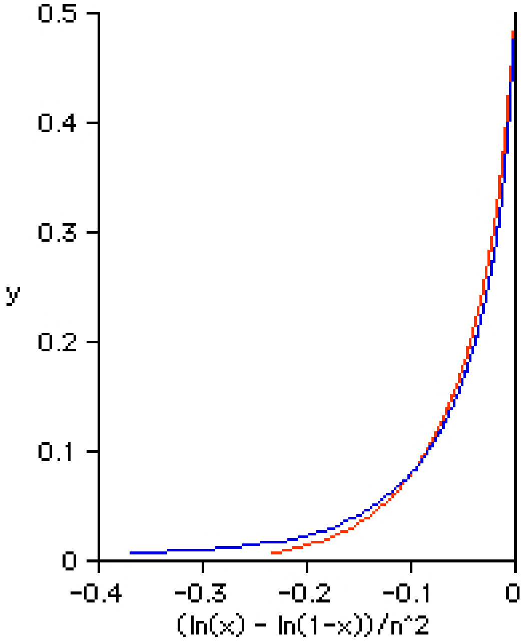
When simulating $X_n$ for larger $n$, such as 100, we used an interactive **C++** program that allows the user to sample values from $X_{100}$, for instance, with interactions concerning when to stop the simulation. The **C++** program is trained to stop the simulation itself if it detects that the sensitivities of the leaf nodes, collectively, are sufficiently small. The stopping condition is easily modified by the user, so we continue to experiment with a variety of stopping conditions.

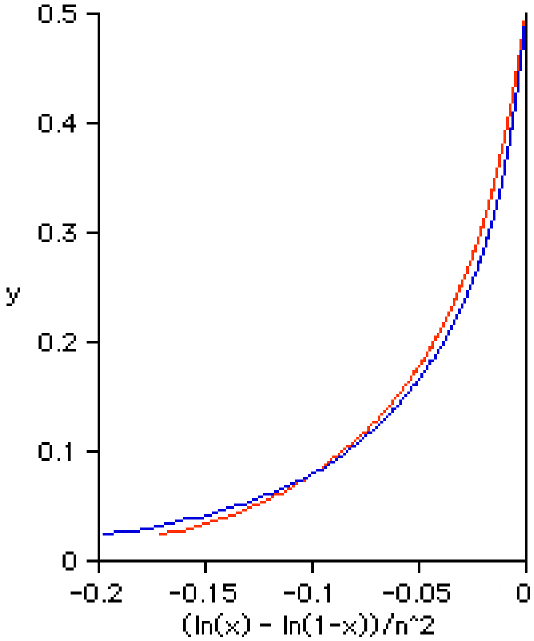
The **C++** program has several other features. For instance, it lets us visualize the data by examining the profile as the tree grows. The evolution of the profile as the most sensitive nodes are selected within the tree is an intriguing phenomenon. Besides further studying the profile, we also plan to continue investigating stopping criteria for the growth of large Boolean binary tree when simulating $X_n$ for large $n$, for example, $n = 100$.

All of the graphs below are for the pairs $(x, y)$ where $y = P(X_{15} \leq x)$ (based on 4,000,000 samples of $X_{15}$) and $y = P(X_{20} \leq x)$ (based on 800,000 samples of $X_{20}$). We rescale the $x$-axis in a variety of ways.









## Acknowledgments

We appreciate the input of Svante Janson, who simultaneously derived several of the observations presented here. We also acknowledge Bob Sedgewick's insightful advice about using randomization in the implementation of data structures (*Finding Paths in Graphs*, A of A 2005).

## References

[1] C. Banderier, M. Bousquet-Mélou, P. Flajolet A. Denise, D. Gardy, and D. Gouyou-Beauchamps. Generating functions for generating trees. *Discrete Mathematics*, 246:29–55, 2002.

[2] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288:21–43, 2002.

[3] B. Chauvin, P. Flajolet, D. Gardy, and B. Gittenberger. And/or tree revisited. *Combinatorics, Probability and Computing*, 13(4–5):475–497, 2004.

[4] R. Durrett. *Probability: Theory and Examples.* Duxbury, Belmont, CA, 3rd edition, 2005.

[5] W. Feller. *An Introduction to Probability Theory and Its Applications.* Wiley, New York, 1968, 1971.

[6] J. Fill, P. Flajolet, and N. Kapur. Singularity analysis, hadamard products, and tree recurrences. *Journal of Computational and Applied Mathematics*, 174:271–313, February 2005.

[7] D. Gardy and A. Woods. And/or tree probabilities of boolean functions. In Conrado Martínez, editor, *2005 International Conference on Analysis of Algorithms*, volume AD of *DMTCS Proceedings*, pages 139–146. Discrete Mathematics and Theoretical Computer Science, 2005.

[8] H. Lefmann and P. Savický. Some typical properties of large and/or Boolean formulas. *Random Structures and Algorithms*, 10:337–351, 1997.

[9] J. Neveu and J. Pitman. The branching processes in a brownian excursion. In *Séminaire de Probabilités, XXIII*, volume 1372 of *Lecture Notes in Mathematics*, pages 248–257. Springer-Verlag, New York, 1989.

[10] P. Savický. Bent functions and random Boolean formulas. *Discrete Mathematics*, 147:211–237, 1995.

[11] P. Savický. Complexity and probability of some Boolean formulas. *Combinatorics, Probability and Computing*, 7(4):451–463, 1998.

[12] P. Savický and A. Woods. The number of Boolean functions computed by formulas of a given size. *Random Structures and Algorithms*, 13(3–4):349–382, 1998.

[13] I. Wegener. *The complexity of Boolean functions.* Teubner, Stuttgart, 1987.