Information and Computation **150**, 187–208 (1999)
Article ID inco.1998.2776, available online at http://www.idealibrary.com on IDEAL®

Malign Distributions for Average

Andreas Jakoby, Rüdiger Reischuk*, and Christian Schindelhauer

Institut für Theoretische Informatik, Medizinische Universität zu Lübeck, Wallstraße 40, 23560 Lübeck, Germany
E-mail: {jakoby,reischuk,schindel}@tcs.mu-luebeck.de

In contrast to machine models like Turing machines or random access machines, circuits are a static computational model. The internal information flow of a computation is fixed in advance, independent of the actual input. Therefore, size and depth are natural and simple measures for circuits and provide a worst-case analysis. We consider a new model in which an internal gate is evaluated as soon as its result has been determined by a partial assignment of its inputs. This way, a dynamic notion of delay is obtained which gives rise to an average case measure for the time complexity of circuits. In a previous paper we have obtained tight upper and lower bounds for the average case complexity of several basic Boolean functions. This paper examines the asymptotic average case complexity for the set of all n-ary Boolean functions. In contrast to worst case analysis a simple counting argument does not work. We prove that with respect to the uniform probability distribution almost all Boolean functions require at least $n - \log n - \log\log n$ expected time. On the other hand, there is a significantly large subset of functions that can be computed with a constant average delay. Finally, for an arbitrary Boolean function we compare its worst case and average case complexity. It is shown that for each function that requires circuit depth d, i.e. of worst-case complexity d, the expected time complexity will be at least $d - \log n - \log d$ with respect to an explicitly defined probability distribution. In addition, a nontrivial upper bound on the complexity of such a distribution will be obtained.

Key Words: average complexity; circuit complexity; Boolean functions; asymptotic complexity; malign distributions; lower bounds.

1. INTRODUCTION

Complexity theory is traditionally based on worst-case measures. Up until recently, the average amount of resources necessary to solve a computational problem has been analyzed only in a few cases, which have mostly considered the expectation

* Corresponding author.

0890-5401/99 $30.00

with respect to a uniform distribution over the input space. Levin has recognized that this simple analysis has serious drawbacks from a complexity theoretic point of view and has made a proposal as to how one can overcome these difficulties [Lev86]. In order to deal with broader classes of distributions a complexity measure for the distributions themselves is needed.

Levin's ideas have further been developed in [Gur91; BCGL92; RS96; Sch96] to define various average-case complexity classes, based on the time complexity of Turing machine computations. This is motivated by the question whether computational problems, difficult in the worst case, might efficiently be solvable, at least on the average (see also the discussion in [WB92]). However, it has been shown that certain $\mathcal{NP}$-complete problems are likely to remain infeasible on the average. As in worst-case analysis this can be done with the help of a reducibility notion between distributional problems and the existence of complete problems in this sense. If the *tiling* problem, for example, could be solved in average polynomial time with respect to the uniform distribution, then every $\mathcal{NP}$-complete problem has polynomial average-case complexity with respect to any distribution that can be computed in polynomial time [Lev86].

For machine models like Turing machines or random access machines—in contrast to static computational models like Boolean circuits—even for fixed problem size, the length of a computation in general depends on the specific input. The data access patterns may vary among different inputs, whereas the information flow in a circuit is totally predetermined—it cannot adjust dynamically. Therefore, the depth of a circuit has served as a worst-case measure for (parallel) time complexity. It is not obvious how to obtain a meaningful notion of time complexity that can be used for an average-case analysis of the circuit model. Such a measure should allow a decrease of the computational resource, at least in certain cases. In particular, due to the trivial logarithmic lower bound on circuit depth that holds for almost all n-argument Boolean functions, is there any way to speed up the computation time below the logarithm? Indeed, in certain favourable cases the result of an output gate may be available much earlier. For example, for an OR-gate this happens as soon as one of its predecessors delivers the value 1. If this predecessor does not lie on a critical path (a path of maximal length between input and output gates) the computational delay is actually smaller than the circuit depth. For a comparison of circuit depth and maximal critical path length see [Kra78].

In [JRS94] we have shown how this timing information can be used to define the notion of delay for gates of a circuit which may be different for each input vector. This way, one obtains a meaningful average case measure of time for the circuit model. The timing information can also be made explicit and then be used in actual circuit designs. Hardware designers have exploited a similar technique when dealing with so-called *self-timed circuits* [DGY89; LBS93]. Thus, good average case upper bounds for the circuit model have important practical implications.

For a number of basic Boolean functions an exponential speedup can be obtained when comparing average delay to circuit depth. A quite important example is the addition of two binary numbers [JRS94]. On the other hand, the parity function requires logarithmic delay even on the average. The addition is basically equivalent to computing all prefixes of a linear formula over a specific semigroup. The average

complexity of the parallel prefix problem for arbitrary semigroups has been investigated in [JRSW94] and in more detail in [Jak98]. By proving matching upper and lower bounds we have shown that the complexity depends only on algebraic properties of the semigroup and that only three different situations are possible. The average delay is either constant, for example in case of the OR-function, or is of order $\log \log n$, for example in case of the addition, or is of order $\log n$ as for the parity function.

The average complexity to some extent depends on the set of distributions that may occur. Restricting an average case analysis only to the uniform distribution is of limited interest. But one has observed that allowing arbitrary distributions the average case complexity equals the worst-case complexity for uniform computational models. Li and Vitanyi have shown that one particular distribution, called *universal* or *Solomonoff–Levin distribution*, has the property that the average complexity of any machine is at most a constant factor smaller than its worst-case complexity, where the constant depends on the particular machine [LV92]; see also [Kob93]. This distribution is closely related to the Kolmogorov complexity of strings [LV93], and thus is not recursive.

Fortunately, one may therefore argue that in real computations such input distributions do not occur. In order to restrict the set of allowable distributions time and space limits have been considered for Turing machines that generate an individual distribution. Levin [Lev86] has proposed the notion *computable*. It requires that the corresponding distribution function can be approximated in polynomial time. A weaker notion based on probabilistic machines is called *sampleable* [BCGL92]. We have defined another natural notion called *rankable* [RS96] and have obtained tight hierarchies for average case complexity classes with respect to time bounds for machines, as well as with respect to the complexity of the distributions.

Milterson has extended the result of [LV92] to subclasses $\mathscr{C}$ of machines with a fixed upper time bound. He calls a distribution *malign* for $\mathscr{C}$ if the average complexity of any machine in $\mathscr{C}$ is at most a constant factor smaller than its worst case complexity. In [Mil91] it has been shown that such a distribution exists. It is computable in exponential time and malign for the class $\mathscr{P}$ (see also Proposition 2 in [BCGL92] for a similar result). For machines with a fixed polynomial time bound malign distributions can be generated with a Σ_2-oracle already in polynomial time [Mil91], and in a slightly different model even a $\mathscr{NP}$-oracle suffices [RS96]. Furthermore, if the probabilities of input strings do not decrease too fast (with respect to their length) to 0 no distribution computable in deterministic polynomial time can be malign for $\mathscr{P}$.

If computability is replaced by the weaker notion of sampleability it has been shown that the class $\mathscr{NP}$ has malign distributions that can be generated, i.e., probabilistically sampled, in polynomial time [BCGL92]. Grape [Gra90] has proved that $\mathscr{P}$ and $\mathscr{NL}$ have malign distributions that are sampleable in logarithmic space.

In this paper we will study the question of *maligness* for the nonuniform circuit model. Recursion theoretic tools like the universal distribution will not be of any help in this case. Instead, completely different and explicit constructions are necessary to obtain hardness results. For the class of all Boolean functions it will be shown

that malign distributions do not exist, i.e., for every distribution there exists a Boolean function for which the average delay is significantly smaller than its worst case delay.

Next, the asymptotic behaviour of the delay measure will be studied. In the worst case the so-called *Shannon effect* holds: almost all n-argument Boolean functions require depth $n - \log\log n$ [Weg87] and this lower bound is optimal since it can be achieved up to a small additive constant [Gas78]. The situation for the average delay turns out to be more complicated. For a large portion of functions we can almost obtain this lower bound even for the uniform distribution. On the other hand, there is a significantly large subset of functions that can be computed with a constant average delay.

Finally, we will show that for any Boolean function f with a given worst case complexity one can explicitly construct a distribution that is *bad* for all circuits realizing f. The average case delay of any such circuit will be smaller by at most an additive term of order $\log n$, compared to the worst-case complexity of f. We will also determine the complexity of such distributions, which will require the most technical effort.

To summarize, for the Turing machine model the trade-off between average time resources and the complexity of distributions has been studied quite extensively. For the circuit model we have determined this trade-off exactly for the whole range of distributions for several specific functions. In particular, the cutpoint where average and worst-case complexity become identical has been located precisely in these cases. This paper solves these questions for the asymptotic setting. Our results imply in particular that even for the nonuniform computational model Boolean circuits for each function one can find a complex distribution that makes the average case complexity almost identical to the worst-case complexity.[1]

2. AN AVERAGE CASE MEASURE FOR CIRCUITS

For the Boolean circuit model the depth is usually taken as a measure for the computational delay. A close relationship between circuit depth and time complexity measures of several parallel machine models has been shown. Note that depth as well as parallel time are so far considered worst-case measures.

DEFINITION 1. Let $\mathbf{B_n^m}$ denote the set of Boolean functions f: $\{0,1\}^n \to \{0,1\}^m$. $\mathbf{B_n} := B_n^1$. $\mathscr{D}_n$ denotes the set of all probability distributions μ on $\{0,1\}^n$. The uniform distribution on $\{0,1\}^n$ that gives equal probability to each of the 2^n possible input vectors is denoted by $\mu_{n,\,\mathrm{uni}}$. Circuits will be defined over the standard basis of AND, OR (fanin 2), and NOT gates. For an internal gate v and an input vector x of a circuit C let $\mathrm{res}_v(x)$ denote the Boolean value computed by v in the computation for x. Let $\mathrm{Cir}(f)$ denote the set of all circuits that compute f and $\mathrm{CirDepth}_n(\mathbf{d})$ all functions in B_n that can be computed by a circuit of depth at most d.

[1] A preliminary version of these results has been presented at *STACS'95, 12th Symposium of Theoretical Aspects of Computer Science, München, 1995* (Springer Lecture Notes in Computer Science, Vol. 900, pp. 628–639).

FACT 1. [Gas78; Weg87]. $B_n = \mathrm{CirDepth}_n(n - \log\log n + O(1))$. *This bound is best possible* (*up to a small additive term*) *for almost all functions in* B_n.

In a circuit C, internal results can be deduced instantenously if, for example, one of the input bits of an OR-gate is already available and has value 1. Then, the output of the OR-gate has to be 1, no matter what Boolean value the other input port will deliver later. For each gate v we will define a function $\mathrm{time}_v\colon \{0,1\}^n \to \mathbb{N}$ to measure the delay of gates. $\mathrm{time}_v(x)$ specifies for each input vector x of C the first time step t when v knows its value $\mathrm{res}_v(x)$ using the information supplied by the partial bit vector of values from its predecessors at step t.

DEFINITION 2. Let C be a circuit and v be a gate of C. For input gates and constant gates v let $\mathrm{time}_v(x)$ equal 0. For an internal nonconstant gate v with k direct predecessors $v_1, \ldots, v_k$ define

$$\mathbf{time}_v(x) := 1 + \min\{t \mid \text{the values } \mathrm{res}_{v_i}(x) \text{ of those } v_i \text{ with } \mathrm{time}_{v_i}(x) \leqslant t \text{ uniquely determine } \mathrm{res}_v(x)\}.$$

For the circuit C itself with output gates $y_1, \ldots, y_m$ we define the global delay function by

$$\mathbf{time}_C(x) := \max_i \mathrm{time}_{y_i}(x).$$

The **worst case time complexity** of a Boolean function f is the smallest maximal delay that can be achieved by circuits computing f

$$\mathbf{time}(f) := \min_{C \in Cir_\mu(f)} \max_x \mathrm{time}_C(x).$$

The class of all functions that can be computed with worst case delay at most t will be denoted by

$$\mathbf{CirTime_n(t)} := \{f \in B_n \mid \mathrm{time}(f) \leqslant t\}.$$

For example, the delay of an OR-gate v with predecessors v_1, v_2 is given by

$$\mathrm{time}_v(x) := 1 + \begin{cases} \max\{\mathrm{time}_{v_1}(x), \mathrm{time}_{v_2}(x)\}, & \text{if } \mathrm{res}_{v_1}(x) = \mathrm{res}_{v_2}(x) = 0, \\ \min\{\mathrm{time}_{v_1}(x) \mid \mathrm{res}(v_i) = 1\}, & \text{else.} \end{cases}$$

Up to this point delay has only been defined implicitly. In [JRS94] we have shown how this information can also be generated, explicitly increasing the circuit size by at most a constant factor.

Let us make a few remarks concerning circuit depth and worst-case time complexity. Obviously, the depth provides an upper bound for the worst cast delay. A circuit of minimal worst-case delay does not necessarily have to be minimal with respect to depth. In Section 6 we will show that a delay-optimal circuit can be

redesigned to achieve also minimal depth. However, such a transformation may result in a blowup of the circuit size.

Given an arbitrary circuit C, in general, it is difficult to determine or approximate its maximal delay. In [JS96] we have shown that this problem is co-$\mathcal{NP}$-hard, whereas the depth can easily be computed in $\mathcal{NC}$. Furthermore, the function time_C is complete for $\mathcal{P}$.

DEFINITION 3. Given a function $t: \{0,1\}^n \to \mathbb{N}$ and a probability distribution $\mu: \{0,1\}^n \to [0;1]$ let $E_\mu(t) := \sum_x t(x)\,\mu(x)$ denote the expectation of t with respect to μ. If D is a set of probability distributions we define

$$\textbf{etime}(f, D) := \max_{\mu \in D} \min_{C \in Cir_\mu(f)} E_\mu(\text{time}_C)$$

as the **expected time complexity** of f with respect to distributions in D. The complexity class of all functions that can be computed within expected time at most t with respect to distributions in D will be denoted by

$$\textbf{ECirTime}_n(t, D) := \{f \in B_n \mid \text{etime}(f, D) \leqslant t\}.$$

We have shown that the expected time complexity can be approximated in polynomial time [JS96].

A simple example of a Boolean function that can be computed significantly faster in the average case, compared to the worst case (for which the trivial logarithmic lower bound for the depth holds) is the n-ary OR-function with the property [JRS94],

$$\text{etime}(\text{OR}_n, \{\mu_{n,\text{un}}\}) = 2 - 2^{-(n-2)}.$$

3. THE COMPLEXITY OF DISTRIBUTIONS

For the average case analysis of circuits a complexity measure for the distributions that generate the random inputs is needed. For uniform computational models we have discussed this issue above. When dealing with circuits we will measure the complexity by the circuit model itself, more precisely by circuit depth. To generate a distribution a circuit gets as input vectors of truly random bits and has to output vectors according to the specific distribution. This may be considered as the circuit analog of *Turing machine sampleable*. In the following we will identify a distribution μ with a random variable X distributed according to μ. Let $X = X_1, ..., X_n$.

DEFINITION 4. Let $C \in \text{Cir}(B_r^n)$ perform a transformation of a random variable Z defined over $\{0,1\}^r$ into a random variable X over $\{0,1\}^n$ as follows. The input vector for C is chosen according to Z. Then X equals the distribution of the output vector generated at the output gates of C. If Z is the uniform distribution over $\{0,1\}^r$ such a circuit will be called a **distribution generating circuit for X**, in short **DG-circuit**.

No interesting results can be obtained if we consider distributions generated by DG-circuits with unbounded fan-out because then all output bits may depend on a single random bit.

As an example, consider a DG-circuit with random inputs $x_0, ..., x_n$ computing $x_0 \wedge x_1, x_0 \wedge x_2, ..., x_0 \wedge x_n$. In this case, it holds $\Pr[X = 0^n] = \frac{1}{2} + 2^{-(n+1)}$ and $\Pr[X = y] = 2^{-(n+1)}$ for $y \neq 0^n$. For $\mathtt{OR}_n$ this distribution implies a lower bound of $\frac{1}{2} \log n$ for the expected delay. However, we have shown that for any probability distribution that is generated by a constant-depth circuit with bounded fan-out the expected delay for $\mathtt{OR}_n$ is constant. So, by "reusing" a random bit an unlimited number of times, within one parallel step one could generate very asymmetric distributions quite easily. Therefore, DG-circuits are required to have a constant fan-out; let us say fan-out 2.

Based on DG-circuits we classify distributions as follows:

DEFINITION 5.

$$\mathscr{D}\ \mathbf{Depth}_n(d) := \{\mu \in \mathscr{D}_n \mid \exists \text{DG-circuit } C \in \mathrm{Cir}(B_n^r)\text{: depth}(C) \leqslant d \wedge C(\mu_{r,\,uni}) = \mu\}.$$

Using this notion, the following classifications for the average complexity of some basic Boolean functions f has been given [JRS94]. Let $\mathtt{AND}_n$ denote the n-ary conjunction, $\mathtt{EQUAL}_n$ the test for equality of two binary strings of length n and $\mathtt{ADDITION}_n$ the addition of 2 binary numbers of length n. $\mathtt{THRESHOLD}_n^m$ denotes the function that is 1 if at least m inputs are 1 and $\mathtt{MAJ}_n := \mathtt{THRESHOLD}_n^{\lceil n/2 \rceil}$:

For $\mathtt{OR}_n$, $\mathtt{AND}_n$, $\mathtt{EQUAL}_n$:

$$\mathrm{etime}(f, \mathscr{D}\mathrm{Depth}(d)) = \Theta(\min(2^d \log n)),$$

for $\mathtt{PARITY}_n$, $\mathtt{MAJ}_n$:

$$\mathrm{etime}(f, \{\mu_{n,\,\mathrm{uni}}\}) = \Theta(\log n),$$

$$\mathrm{etime}\ (\mathtt{ADDITION}_n, \mathscr{D}\mathrm{Depth}(d)) = \Theta(\min(\log\log n + 2^d, \log n)),$$

$$\mathrm{etime}(\mathtt{THRESHOLD}_n^m, \mathscr{D}\mathrm{Depth}(d)) = \Theta(\min(\log m + 2^d, \log n)).$$

4. CIRCUITS DO NOT HAVE MALIGN DISTRIBUTIONS

For any probability distribution there is a nontrivial disjunction depending on only a small subset of variables that yields 1 with high probability. For $x, \alpha \in \{0, 1\}$ let x^α equal x if $\alpha = 1$, and $\neg x$ else. By $\log n$ we denote $\lceil \log_2 n \rceil$, the binary logarithm rounded up.

LEMMA 1. *Let $X = x_1, ..., x_n$ be a random variable on $\{0, 1\}^n$ with an arbitrary distribution μ, and let $\{i_1, i_2, ..., i_l\} \subseteq [1, ..., n]$. Then there exists Boolean constants $\alpha_1, ..., \alpha_l$ such that*

$$\Pr_\mu[\mathtt{OR}(x_{i_1}^{\alpha_1}, ..., x_{i_l}^{\alpha_l}) = 1] \geqslant 1 - 2^{-l}.$$

Proof. Choose $\alpha_1 = 1$ if $\Pr_\mu[x_{i_1} = 1] \geqslant \frac{1}{2}$, and $\alpha_1 = 0$ else. For arbitrary $\alpha_1, ..., \alpha_k$ it holds either $\Pr_\mu[x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_k}^{\alpha_k} = 1] = 1$, or there exists a value α_{k+1} such that

$$\Pr_\mu[x_{k+1}^{\alpha_{k+1}} = 1 \mid x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_k}^{\alpha_k} = 0] \geqslant 1/2.$$

Assume inductively that $\Pr_\mu[x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_k}^{\alpha_k} = 1] \geqslant 1 - 2^{-k}$, then

$$\begin{aligned}
&\Pr_\mu[x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_{k+1}}^{\alpha_{k+1}} = 1] \\
&\quad \geqslant \tfrac{1}{2}(1 - \Pr_\mu[x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_k}^{\alpha_k} = 1]) + \Pr_\mu[x_{i_1}^{\alpha_1} \vee \cdots \vee x_{i_k}^{\alpha_1} = 1] \\
&\quad \geqslant 1 - 2^{-(k+1)}. \quad \blacksquare
\end{aligned}$$

This bound enables us to prove

THEOREM 1. *For every probability distribution μ over $\{0, 1\}^n$ there exists a n-ary Boolean function f with the properties*

$$\operatorname{depth}(f) \geqslant n - \log n - \log\log n, \quad \operatorname{etime}(f, \mu) \leqslant 4.$$

Proof. For given μ, the function f will be defined as follows. Let $l := \log n$ and $\{i_1, ..., i_l\} = \{1, ..., l\}$. With respect to μ choose $\alpha_1, ..., \alpha_l$ according to the lemma above. Using Fact 1 we can find a function $g \in B_{n - \log n}$ such that $g \notin \operatorname{CirDepth}(n - \log n - \log\log n - 1)$. Let

$$f(x_1, ..., x_n) := \mathtt{OR}(x_1^{\alpha_1}, ..., x_l^{\alpha_l}, g(x_{l+1}, ..., x_n)).$$

To prove the lower bound assume $\operatorname{depth}(C) < n - \log n - \log\log n$ and $C \in \operatorname{Cir}(f)$. Replacing the first input bits $(x_1, ..., x_l)$ by the constant vector $(\neg\alpha_1, ..., \neg\alpha_l)$ one obtains a circuit C' of the same depth that computes g—a contradiction.

For the upper bound the OR-subfunction is realized by the average case optimal design presented in [JRS94] with average delay less than 2; i.e., $C_{\mathtt{OR}, n}(x_1, ..., x_n) := \mathtt{OR}(x_n, C_{\mathtt{OR}, n-1}(x_1, ..., x_{n-1}))$. For g we use a depth-optimal circuit of depth at most $n - \log n - \log\log n + O(1)$. Then the overall average delay can be estimated as

$$\begin{aligned}
E_\mu(\operatorname{time}_C) &\leqslant 1 + \Pr_\mu[\mathtt{OR}(x_1^{\alpha_1}, ..., x_l^{\alpha_l}) = 1] \cdot 2 \\
&\quad + \Pr_\mu[\mathtt{OR}(x_1^{\alpha_1}, ..., x_l^{\alpha_l}) = 0] \cdot (\operatorname{depth}(g) + \log n) \\
&\leqslant 1 + 2 + n/n = 4. \quad \blacksquare
\end{aligned}$$

Hence, the class B_n of all n-argument Boolean functions does not have malign distributions.

In the last proof only bits $\alpha_1, ..., \alpha_{\log n}$ depend on the given probability distribution μ. Therefore, even a small set of n functions suffices, such that for any distribution there is a member in this set with a huge difference between average and worst case complexity, i.e., from constant to linear.

5. ASYMPTOTIC BOUNDS WITH RESPECT TO THE UNIFORM DISTRIBUTION

For a circuit C and a natural number t define the set of ***t*-bad inputs** as

$$I[C, t] := \{x \in \{0,1\}^n \mid \text{time}_C(x) > t\}.$$

Chebyshev's inequality implies $|I[C,t]| \leqslant (2^n/(t+1))\, E_{\mu_{n,\,\text{uni}}}(\text{time}_C)$.

THEOREM 2. *Almost all functions $f \in B_n$ have average complexity larger than $n - \log n - \log\log n - 3$ w.r.t. the uniform distribution.*

Proof. Let $d := \lfloor n - \log n - \log\log n - 1 \rfloor$. Assume that given the uniform distribution a function $f \in B_n$ can be computed by an expected d-time bounded circuit C. One can get rid of NOT-gates by supplying also negated variables $\overline{x_j}$ as inputs. Furthermore, without increasing the depth C can be expanded to a circuit C' with fanout 1.

Next, we extend the Boolean domain by a new symbol "?". The OR and AND function are then defined by Table 1.

Let us cut off all gates with distance larger than d from the output gate and replace noninput gates at distance d by "?". This way, we get a binary tree C'' of depth at most d, where internal gates are labeled with AND or OR and input gates with x_i, $\overline{x_i}$ or "?". By adding redundant gates we may assume that C'' is a complete binary tree. There are less than $2^{2^d(1+\log(2n+1))}$ different such C''; thus each such circuit can be encoded by a binary string of length at most $2^d(1+\log(2n+1))$.

For all $x \notin I[C,d]$ it holds that $\text{res}_C(x) = \text{res}_{C'}(x)$. For $x \in I[C,d]$, however, C'' yields the result "?". Thus, the set $I[C,d]$ is uniquely determined by C''.

Now, the function f can be described by C'' and a list of values $f(x)$ for $x \in I[C,d]$. The length of this description is bounded by

$$\begin{aligned}
|C''| + |I[C,d]| &\leqslant 2^d(1+\log(2n+1)) + 2^n \frac{d}{d+1} \\
&\leqslant \frac{2^n}{2n\log n}(3+\log n) + 2^n\left(1 - \frac{1}{d+1}\right) \\
&\leqslant 2^n\left(1 - \frac{1}{n+\log n+\log\log n} + \frac{1}{2n} + \frac{3}{2n\log n}\right) \\
&\leqslant 2^n\left(1 - \frac{1}{\Omega(n)}\right).
\end{aligned}$$

This upper bound contradicts the fact that B_n contains 2^{2^n} elements and, hence, the Kolmogorov complexity of almost all n-ary Boolean functions has to be at least 2^n. ∎

Because of the upper depth bound $n - \log\log n$ this lower bound is best possible up to an additive logarithmic term. On the other hand, it is not difficult to construct a set of $2^{2^{n-\log^n}}$ n-ary Boolean functions with average delay at most 4 with respect

TABLE 1

Three-valued logic for the Boolean functions OR, AND

OR	?	0	1	AND	?	0	1
?	?	?	1	?	?	0	?
0	?	0	1	0	0	0	0
1	1	1	1	1	?	0	1

to the uniform distribution. The portion of such functions that are efficiently computable in the average grows when the delay bound is increased. This result should be compared to the worst case, where any set of $2^{2^{n-\log n}}$ n-ary Boolean functions contains functions that require depth $n-2\log n$.

6. WORST-CASE DELAY VERSUS DEPTH

Consider the circuit C'' constructed in the proof of Theorem 2 again. It is easy to see that $\mathrm{res}_{C''} \equiv \mathrm{res}_C$ if the cut value d equals the worst-case delay $\max_x \mathrm{time}_C(x)$ of C. Therefore, unfolding a circuit as a tree to get a formula with fanout 1 and cutting paths longer than the worst-case delay we get a depth optimal circuit. However, this may result in an exponential increase of the size of the circuit. Using a more clever construction we can show

THEOREM 3. $\mathrm{CirTime}(t) = \mathrm{CirDepth}(t)$. *Furthermore, depth-optimal circuits can be derived from delay-optimal circuits with only a quadratic increase in size.*

Proof. Since the depth of a given circuit is a trivial upper bound for its worst-case delay, it obviously holds that $\mathrm{CirTime}(t) \supseteq \mathrm{CirDepth}(t)$.

To prove the converse inclusion let $f \in \mathrm{CirTime}(t)$ and C be a circuit for f with worst-case delay t. We will approximate the values res_v computed by the internal gates v of C by a sequence $v[0], v[1], ..., v[t]$ of gates. Let v be such a noninput gate v with direct predecessors $u_1, ..., u_k$ computing the Boolean function g_v in C; that is, $\mathrm{res}_v \equiv g_v(\mathrm{res}_{u_1}, ..., \mathrm{res}_{u_k})$. For $\tau = 0$ let $v[\tau]$ be identical to the constant function 0. For $0 < \tau \leqslant t$ connect the copy $v[\tau]$ of v to the copies $u_1[\tau-1], ..., u_k[\tau-1]$ in the same way as v connects to the u_i to get

$$\mathrm{res}_{v[\tau]} \equiv g_v(\mathrm{res}_{u_1[\tau-]}, ..., \mathrm{res}_{u_k[\tau-1]}).$$

Here for an input gate u_i, $u_i[\tau-1]$ simply denotes the gate u_i itself. Note that this new circuit C' has depth t and size equal to $t \cdot \mathrm{size}(C)$, that means at most a quadratic blowup.

We claim that for every input x

$$\mathrm{res}_{v[\tau]}(x) = \mathrm{res}_v(x) \qquad \text{for all} \quad \tau \geqslant \mathrm{time}_v(x).$$

The claim is true for all input gates. If $\tau \geqslant \mathrm{time}_v(x)$ it follows from the definition that some predecessors u_i of v with $\mathrm{time}_{u_i}(x) < \tau$ uniquely determine the result of v.

By induction, we can assume that for these u_i, $\mathrm{res}_{u_i[\tau-1]}(x) = \mathrm{res}_{u_i}(x)$ holds. Hence, no matter what values the other gates $U_i[\tau-1]$ will provide,

$$\begin{aligned}\mathrm{res}_{v[\tau]}(x) &= g_v(\mathrm{res}_{u_i[\tau-1]}(x), ..., \mathrm{res}_{u_k[\tau-1]}(x)) \\ &= g_v(\mathrm{res}_{u_1}(x), ..., \mathrm{res}_{u_k}(x)) = \mathrm{res}_v(x).\end{aligned}$$

For the output gate $\bar{v}$ of C holds $\mathrm{time}_{\bar{v}}(x) \leqslant t$ for all x. Thus choosing $\bar{v}[t]$ as the output gate of C' we get

$$\mathrm{res}_{C'}(x) = \mathrm{res}_{\bar{v}[t]}(x) = \mathrm{res}_{\bar{v}}(x) = \mathrm{res}_C(x) = f(x). \quad \blacksquare$$

Krapchenko has given an example showing that an increase of the circuit size cannot be avoided in general [Kra78] when for a delay-minimal circuit the depth is also minimized.

7. EFFICIENT SELECTION

In this section we will develop some general techniques needed to construct distributions that are hard for the average case. Recall that in order to get distributions of low complexity one has to design DG-circuits of small depth. Because of the fan-out restriction this problem is nontrivial.

Let us denote the ith bit of a binary string a by $a[i]$. The function $\mathrm{bin}(i, n)$ for integers $i \leqslant 2^n - 1$ denotes the binary representation of i of length n. Let $\mathrm{bin}^{-1}(a)$ be the inverse function. The *multiplexer function* will play an important role in our circuit designs.

DEFINITION 6. Let $m = 2^q$ and n an arbitrary natural number. A **(m, n)-multiplexer** takes as input m binary vectors $x_0, ..., x_{m-1}$ each of length n, and a control vector $y \in \{0, 1\}^q$, and outputs $x_{\mathrm{bin}^{-1}(y)}$, i.e., the yth input vector, where y is interpreted as a binary number.

The kth bit $(0 \leqslant k < n)$ of the output vector z can be represented as

$$z[k] := \bigvee_{i=0}^{m-1} \left(x_i[k] \wedge \bigwedge_{j=1}^{q} y_j^{\mathrm{bin}(i,q)[j]} \right)$$

(as defined above y_j^b denotes the bit y_j if $b = 1$, else its negation, and $\mathrm{bin}(i, q)[j]$ the jth bit in the binary representation of i). Computing all output bits in parallel using this formula shows that a (m, n)-multiplexer can be realized by a circuit of depth $\log m + \log(1 + q)$ if there is no fanout restriction. Notice that each control bit y_j is used $m \cdot n$ times. To obtain a circuit of constant fan-out one could duplicate these bits using trees. This increases the depth by $\log m + \log n$.

FACT 2. *A (m, n)-multiplexer can be computed in depth* $2\log m + \log(1 + \log m) + \log n$ *by a circuit with fanout* 2. *It has the additional property that the input bits* $x_i[k]$ *are accessed only by gates of depth at least* $\log m + \log(1 + \log m) + \log n$.

Now, using unary coding for the control input we will construct more efficient multiplexers for the case of bounded fanout.

DEFINITION 7. For given input vectors $x_0, ..., x_{m-1} \in \{0,1\}^n$ and control input $y_0, ..., y_{m-1} \in \{0,1\}^m$ a **unary controlled (m, n)-multiplexers** $\mathtt{UCMX}_{m,n}$ outputs the first input vector x_i for which $y_i = 1$; otherwise the last vector x_{m-1}.

LEMMA 2. *$\mathtt{UCMX}_{m,n}$ can be realized by a circuit of depth* $\log m + \lceil \sqrt{8 \log m} \rceil + \log n + 1$.

Proof. Run n many $\mathtt{UCMX}_{m,1}$ circuits in parallel, one for each output bit. For the initial duplication of the control bits $y_0, ..., y_{m-1}$ we use fanout trees of depth $\log n$. In the following the details of constructing a $\mathtt{UCMX}_{m,1}$ circuit S_m will be described.

First, in one parallel step we compute $x'_i := x_i \wedge y_i$ for all $i < m-1$ and $x'_{m-1} := x_{m-1}$. This way, an unused data bit is set to 0. Observe that $\mathtt{UCMX}_{m,1}(\mathbf{x}, \mathbf{y}) = \mathtt{UCMX}_{m,1}(\mathbf{x}', \mathbf{y}')$. S_m computes the result $r = \mathtt{UCMX}_{m,1}(\mathbf{x}', \mathbf{y})$ as follows (for the case $m = 4$ see Fig. 1):

$$\forall j < m, \qquad s_j := x'_j \wedge \neg \bigvee_{i<j} y_i,$$

$$r := \bigvee_{j<m} s_j.$$

Since each of the control inputs y_i is needed up to m times in a single S_m circuit further duplication is done by fanout trees of depth $\log m$. Thus, the total depth of this construction adds up to $3 \log m + 2$.

Note that the inputs x' are used in depth $2 \log m + 1$ for the first time. Furthermore, unary controlled multiplexers can be combined such that an address is divided into parts and fed into the different subcircuits.

For the following description we emphasize the tree-like construction by using binary strings as indices. Inputs and outputs of the subcircuits will be called r. But the closer they are to the input $\mathbf{x}$ the longer their indices are. Let $r_{\mathrm{bin}(i, \log m)} := x'_i$ and $y_{\mathrm{bin}(i, \log m)} := y_i$ for all $i < m$. The overall result will be called r_λ.

The following gives a complete specification of the whole construction enumerating all internal results that are computed (for a wiring see Fig. 2). It looks like a pipelined version of a bunch of $S_{m'}$-circuits for various values of m'. For $\delta \leqslant \log k$ define

$$\forall p \in \{0,1\}^{\leqslant \log m - 1} \qquad e_p := \bigvee_{q \in \{0,1\}^{\log m - [p]}} y_{pq},$$

$$\forall \ell \in [0; \lceil \log m/\delta \rceil - 1], \forall p \in \{0,1\}^{\delta \cdot \ell}, \forall q \in \{0,1\}^{\delta}$$

$$s_{pq} := r_{pq} \wedge \neg \bigvee_{u \in \{0,1\}^{\delta} \text{ and } u < q} e_{pu},$$

$$\forall \ell \geqslant 0, \qquad \forall p \in \{0,1\}^{\delta \cdot \ell} \qquad r_p := \bigvee_{q \in \{0,1\}^{\delta}} s_{pq}.$$

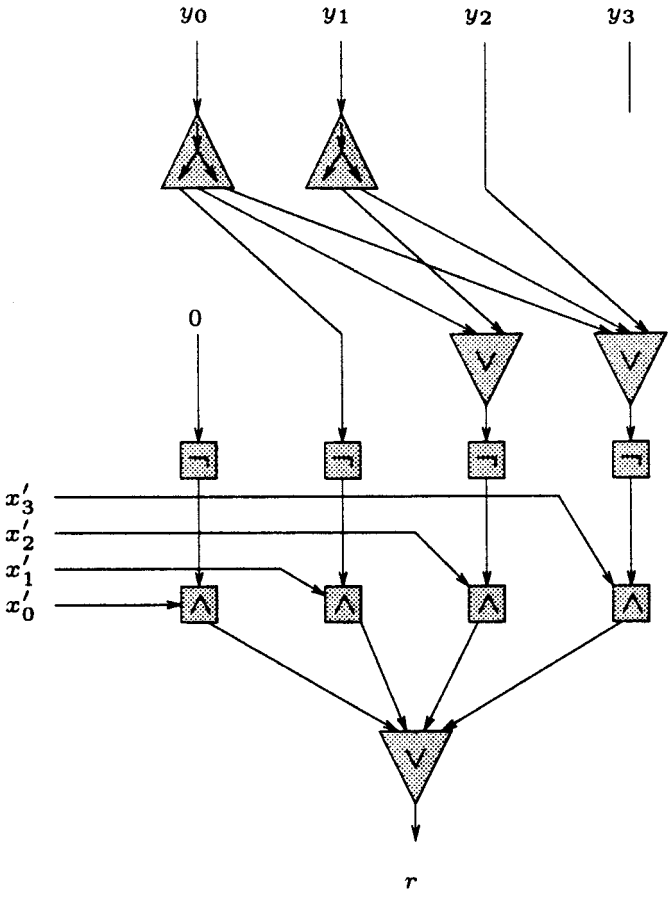

FIG. 1. The $\mathtt{UCMX}_{4,1}$-circuit S_4.

Correctness. For fixed ℓ and $p \in \{0,1\}^{\delta\ell}$ we will prove that

$$r_p = \mathtt{UCMX}_{m,1}(r_{p0^\delta}, ..., r_{p1^\delta}, e_{p0^\delta}, ..., e_{p1^\delta}).$$

The term $\bigvee_{u \in \{0,1\}^\delta \text{ and } u < q} e_{pu}$ indicates whether at least one bit y_{pw} equals 1, where $w \leqslant q$. If this term gets the value 1—which means that r_{pw} is not the first input with control bit $y_{pw} = 1$—then s_{pq} is 0 and r_{pq} does not have any influence in the following computation. Otherwise, s_{pq} equals r_{pq}.

Hence, there is at most one variable s_{pq} for all $q \in \{0,1\}^\delta$ having value 1 and this special case only occurs if $r_{pq} = e_{pq} = 1$, and if e_{pq} is the first control input with value 1. By induction, the value of r_p equals the leftmost input x_{pw} with control bit y_{pw} (resp. equals $x_{p1^{|w|}}$) if there is no such control input.

Efficiency. The computation of e_p for all p can be done by an $\mathtt{OR}$-tree. So value e_p is available in depth $\log m - |p|$. The whole construction consists of levels of thickness at most $\delta + 1$. The level nearest to the inputs $x_0, ..., x_{m-1}$ has thickness

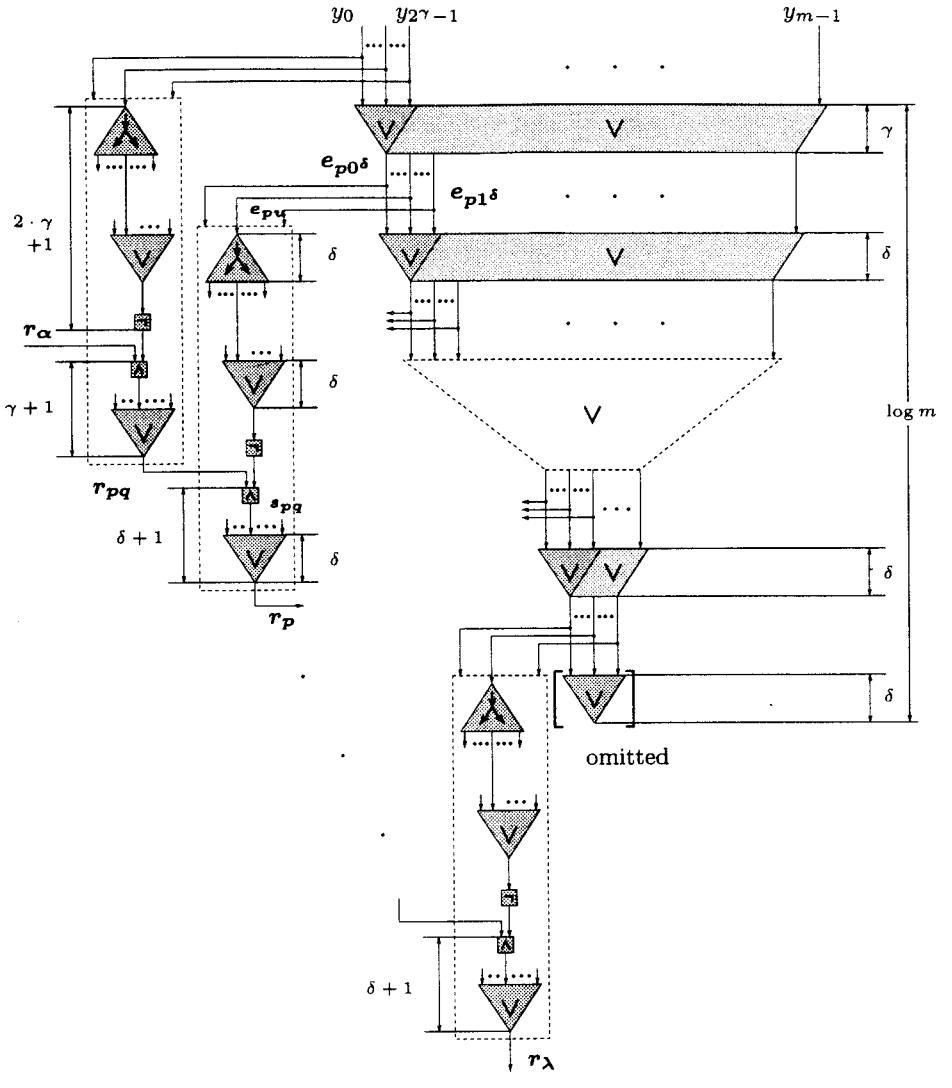

FIG. 2. A circuit for the unary controlled multiplexer.

$\gamma := (\log m - 1) \bmod \delta + 1$. Here e_p for $p \in \{0,1\}^\gamma$ and $r_{\mathrm{bin}(i,\log k)}$ for $i \leqslant k$ are computed in parallel.

For given inputs r_{pq} with $p, q \in \{0,1\}^\delta$ a subcircuit that computes r_p and e_{pu} is embedded into three levels and has depth $3\delta+1$; in the first level every input e_{pu} is broadcasted at most 2^δ times to all subcircuits computing $s_{pq'}$ with $|q'| = |q|$. Therefore, the result of s_{pq} is available in the second level within depth $2\delta+2$ and the result of r_p in the third within depth $3\delta+2$.

Note that inputs r_{pq} and outputs r_p differ by one level only. This holds for the inputs e_{pu} and outputs e_{pq}, too. Therefore, the subcircuit computing r_p can be placed one level below those for r_{pq}.

In the lowest level the output r_λ is computed by a subcircuit using e_u for $u \in \{0,1\}^\delta$. So it is not necessary to compute any e_p with $p < \delta$.

To summarise, there are $\lceil \log m/\delta \rceil + 2$ levels. All levels except the first have thickness $\delta + 1$. The thickness of the uppermost level is bounded by γ. Therefore, the total depth of the $\mathtt{UCMX}_{m,1}$-circuit is given by

$$(\delta+1)\left(\left\lceil \frac{\log m}{\delta} \right\rceil + 1\right) + \gamma \leqslant (\delta+1)\left\lceil \frac{\log m}{\delta} \right\rceil + 2\delta + 1.$$

If we choose $\delta := \lceil \sqrt{\log m/2} \rceil$ the depth of the $\mathtt{UCMX}_{m,n}$-circuit is bounded by $\log m + \lceil \sqrt{8 \log m} \rceil + \log n + 1$. ∎

Next, we consider the problem to construct a random variable that is uniformly distributed over a given set A.

DEFINITION 8. Let m be a power of 2 and n and arbitrary natural number. A **(m, n)-selector** $\mathtt{RS}_{m,n}$ takes as input a sequence of m binary vectors $x_0, ..., x_{m-1} \in \{0,1\}^n$ and outputs a n-bit string z with probability

$$p(z) := |\{j \mid x_j = z\}|/m.$$

LEMMA 3. *A random (m, n)-selector can be realized by a $2 \log m + \log(1 + \log m) + \log n$ depth-bounded DG-circuit using $\log m$ uniformly distributed random input bits. Using m random bits one can achieve the depth-bound $\log m + \log\log m + \log n + 1$.*

Proof. Again, the n output bits $z[k]$ are computed independently in parallel. This requires a duplication of the random bits. The n copies of each random bit are generated at the beginning in depth $\log n$. Thus, we are left with the problem to construct a 1-bit selector $\mathtt{RS}_{m,1}$.

For arbitrary $l \in \mathbb{N}$ define a circuit C_l for $\mathtt{RS}_{l,1}$ using the $(l, 1)$-multiplexer referred to in Fact 2. The control input is provided by $\log l$ random bits $r_1, ..., r_{\log l}$. The total depth of the selector C_l is $2 \log l + \log(1 + \log l)$. For $l = m$ this gives a depth bound of order $(2 + o(1)) \log m$. Below we will show how this can be reduced by a factor 2 by cascading C_l circuits of different sizes in a suitable way for specific values of m.

Remember that the nonrandom input bits in the multiplexer for C_l are used in depth $\log l + \log(1 + \log l)$ for the first time. If $l' \leqslant \sqrt{l}$ the total depth of a circuit $C_{l'}$ is smaller than this number. Thus, instead of feeding C_l directly with inputs $x_i[k]$, one can preselect among different i's by circuits of type $C_{l'}$. In total we add l circuits of type $C_{l'}$ to C_l. This gives a selector of type $\mathtt{RS}_{l \cdot l',1}$ having the same depth as a single C_l. This blowup procedure can be applied to the smaller selectors $C_{l'}$ as well. For an example of a combination of circuits of type C_{2^1}, C_{2^2}, and C_{2^4} see Fig. 3.

In general, for any integer p, combining circuits C_{2^1}, C_{2^2}, C_{2^4}, ..., $C_{2^{2^p}}$ this way, we get a selection circuit D_p for $m_p = \prod_{q=0,...,p} 2^{2^q} = 2^{2^{p+1}-1}$ many inputs. It has the same depth as $C_{2^{2^p}}$, namely $d_p := 2^{p+1} + \log(1 + 2^p)$. Choose p maximal such that $m_p \leqslant m$ and define $m' := m/m^p$; m' is an integer of size less than $2m_p$ since m is a power of 2 and $m_{p+1} = 2m_p^2$.

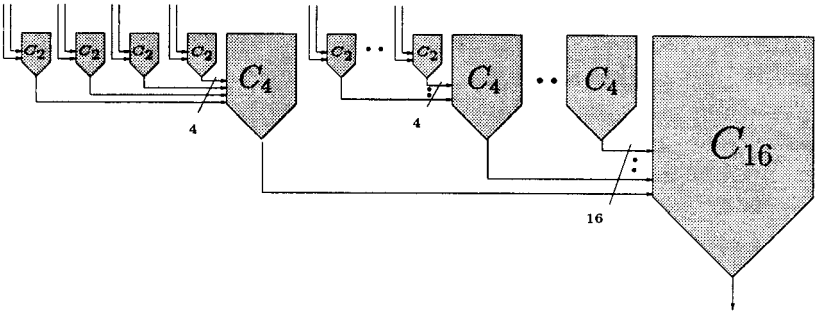

FIG. 3. Combining random selector circuits of type C_2, C_4, C_{16} yields a $\mathrm{RS}_{2\cdot 4\cdot 16,n}$-circuit with the same depth as C_{16}.

The final selector circuit for m inputs is made up of m' circuits D_p plus one circuit $C_{m'}$ that takes the outputs of the D_p's as inputs and selects one of them at random. Note that the delay in the D_p's provides enough time to duplicate the random bits used within $C_{m'}$ and then to compute the $\bigwedge$ over these bits in the formula for $z[k]$. Thus $C_{m'}$ adds a delay of $\log m'$ steps only. Summing up, we achieve a total delay of

$$\begin{aligned}\log n + d_p + \log m' &= 2^{p+1} + \log m' + \log(1+2^p) + \log n \\ &\leqslant \log(2m_p \cdot m') + \log\log m + \log n \\ &\leqslant \log m + \log\log m + \log n + 1. \quad \blacksquare\end{aligned}$$

8. CONSTRUCTING HARD DISTRIBUTIONS

In Section 5 we have shown that the average-case time complexity of most Boolean functions is large—even with respect to the uniform distribution. However, no specific example of such a function has been exhibited yet. Simply from the property that a function has large worst-case delay, let us say larger than $2\log n$, one cannot deduce a lot of information concerning its expected delay.

In the following we will show that for every function f there exists a distribution μ_f that makes the average case complexity of f almost as large as its worst-case complexity. This means, even in this nonuniform model there is no way to exploit information about the likelihood of different input patterns. This contrast to the result of Section 4, where for each distribution a Boolean function has been constructed that has only constant average delay with respect to this distribution. This function depends greatly on the probabilities of individual input vectors.

We will show that there are circuits that can generate such hard distributions. For this purpose, let us introduce a special family of distributions. Each one assigns equal, or almost equal, weights to a particular set of the input domain and zero, or almost zero, weights to the complement. A collection of such distributions, which can be approximated by a circuit of the types presented in the previous section, will make up the final distribution μ_f. Note that, contrary to the linear depth bound for Boolean functions, there is no obvious depth bound for circuits that generate distributions.

In the following construction a lot of effort will be devoted to keep the complexity of the distribution μ_f as small as possible. One can imagine that the complexity of μ_f has to grow with the complexity of f. Let f require depth d. It is not too hard to find a distribution μ_f with complexity $n + O(d)$, i.e., using random selector circuits. We will construct a distribution with an upper bound of the form $\frac{1}{2}(n+d)$. Although the saving of a factor 2 does not seem to be much at first glance, a closer look shows that a simple diagonalization technique like enumerating all 2^n different input patterns cannot be used to obtain such a bound. Instead, a much more involved construction will be necessary.

First, we will investigate distributions that are almost uniform on subsets of $\{0,1\}^n$.

DEFINITION 9. For a nonempty set $A \subseteq \{0,1\}^n$ define the **A-uniform distribution** $\boldsymbol{\mu_A}$ by

$$\mu_A(x) := \begin{cases} |A|^{-1}, & \text{if} \quad x \in A, \\ 0, & \text{else.} \end{cases}$$

If $|A|$ is not a power of 2 this distribution cannot be generated by a DG-circuit since all probabilities of such distributions are (negative) powers of 2. In this case define $\alpha(A) = \lfloor \log_2 |A| \rfloor$. Probability distributions μ are called **nearly A-uniform** if $\mu(x)$ for $x \in A$ is either $2^{-\alpha(A)}$ or $2^{-(\alpha(A)+1)}$, and 0 for $x \notin A$.

It is not obvious how nearly uniform distributions can be generated efficiently.

LEMMA 4. *Any nearly A-uniform distribution of a nonempty set $A \subseteq \{0,1\}^n$ can be generated in depth* $\log |A| + \log\log |A| + \log n + 3$.

Proof. Let $A = \{a_0, ..., a_{|A|-1}\}$ and $m := \log |A|$. We use a $\mathrm{RS}_{m,n}$-circuit and fix the input vectors by $x_i := a_i \bmod |A|$ for all $0 \leqslant i < m$. By Lemma 3 the depth of such a circuit C can be bounded by $\log m + \log\log m + \log n + 1 \leqslant \log |A| + \log\log |A| + \log n + 3$. ∎

If A is relatively large, one could approximate the A-uniform distribution also by generating n-bit strings at random and select one of them that belongs to A. We therefore make the following definition.

DEFINITION 10. The **(A, k)-uniform distribution $\boldsymbol{\mu_{A,k}}$** is given by

$$\mu_{A,k}(x) := \begin{cases} (1-(1-|A|\cdot 2^{-n})^k)\,|A|^{-1}, & \text{if} \quad x \in A, \\ (1-|A|\cdot 2^{-n})^k\,(2^n - |A|)^{-1}, & \text{else.} \end{cases}$$

LEMMA 5. *Let $f \in \mathrm{CirDepth}(d)$ and $A := f^{-1}(1)$. Then the (A,k)-uniform distribution can be generated in depth* $d + \log k + \lceil \sqrt{8 \log k} \rceil + \log n + 1$.

Proof. We use a $\mathrm{UCMX}_{k,n}$-circuit with k random input vectors $x_0, ..., x_{k-1}$ each of length n. First, in parallel all vectors are tested for membership in A using a circuit for f of depth d. The test results provide the control inputs y_i for the multiplexer, which outputs the first vector with a successful test. In case that no test succeeds it outputs the last vector.

The probability that this circuit outputs a string not in A is $(1-|A|/2^n)^k$. All strings in A occur with the same probability. The same holds for the strings in $\bar{A}$. Thus the construction generates a (A, k)-uniform distribution. ▮

These distributions share the following property.

LEMMA 6. *Let $t \in \mathbb{N}$ and μ be a nearly A-uniform distribution or a (A, k)-uniform distribution, where $k \geqslant \ln(2t+1)\, 2^n/|A|$. Further, let h: $\{0,1\} \to \mathbb{N}$ be a function with expectation bounded by t, that is $E_\mu(h) \leqslant t$. Then for the set $A[h, t] := \{x \in A \mid h(y) \leqslant t\}$ it holds that*

$$|A[h, t]| \geqslant \frac{|A|}{2 \cdot (t+1)}.$$

Proof. First consider the case of a nearly A-uniform distribution μ. Let $A_0 := \{x \in A \mid \mu(x) = 2^{-\alpha(A)}\}$, $A_1 := \{x \in A \mid \mu(x) = 2^{-(\alpha(A)+1)}\}$. Then $2 \cdot |A_0| + |A_1| = 2^{\alpha(A)+1}$.

With respect to the bound on $E_\mu(h)$ the set $A[h, t]$ is smallest possible, if for all $x \in A[h, t]$, $h(x) = 0$, and for all $x \in A \backslash A[h, t]$, $h(x) = t+1$.

Since the probability of elements in A_1 is smaller than those in A_0 in order to make the complement of $A[h, t]$ as large as possible such elements should first go to A_1. In other words, A_0 should include as many elements of $A[h, t]$ as possible. It is sufficient to consider the cases $A[h, t] \subseteq A_0$ and $A_0 \subseteq A[h, t]$. In the first case it holds that

$$t \cdot (2 \cdot |A_0| + |A_1|) \geqslant (|A_0| - |A[h, t]|)\, 2(t+1) + |A_1|(t+1).$$

Solving for $|A[h, t]|$ yields $|A[h, t]| \cdot 2(t+1) \geqslant 2|A_0| + |A_1| > |A|$.

In the other case $A_0 \subseteq A[h, t]$: either $|A_0| \geqslant |A|/2(t+1)$ and the claim follows immediately, or $|A_0| < |A|/2(t+1)$. Then

$$t \cdot (2 \cdot |A_0| + |A_1|) \geqslant |A_1 \backslash A[h, t]|\,(t+1) = (|A| - |A[h, t]|)\,(t+1).$$

This implies

$$\begin{aligned}(t+1) \cdot |A[h, t]| &\geqslant (t+1) \cdot |A| - t \cdot (2 \cdot |A_0| + |A_1|) \geqslant |A| - t \cdot |A_0| \\ &\geqslant |A| \cdot (1 - t/(2t+2)) \geqslant |A|/2.\end{aligned}$$

Now let us consider the (A, k)-uniform distribution. Since $k \geqslant \ln(2t+1)\, 2^n/|A|$ it holds for $x \in A$:

$$\mu(x) \geqslant \frac{1 - \left(1 - \dfrac{|A|}{2^n}\right)^k}{|A|} \geqslant \frac{1 - e^{-\ln(2t+1)}}{|A|} = \frac{2 \cdot t}{(2 \cdot t+1)}\, |A|^{-1} =: v.$$

Then, we can conclude

$$t \geqslant \sum_{z \in A \setminus A[h,t]} (t+1)\,\mu(x) \geqslant (|A| - |A[h,t]|)\,(t+1)\,v,$$

$$|A[h,t]| \geqslant |A| - \frac{t}{v \cdot (t+1)} = |A| \left(1 - \frac{t(2t+1)}{(t+1)\,2t}\right) = \frac{|A|}{2(t+1)}. \quad \blacksquare$$

To relate average and worst case delay we first transform average efficient circuits, for which the worst-case delay may be arbitrarily large, into circuits of small depth, which may, however, produce erroneous results. The set of t-bad inputs of a circuit C, $I[C,t]$, will help us to control the quality and quantity of errors.

LEMMA 7. *For $C \in \mathrm{Cir}(f)$ and $t \in \mathbb{N}$ define the functions $f_t^{\langle 1\rangle}$ and $f_t^{\langle 0\rangle}$ by*

$$f_t^{\langle 1\rangle}(x) := f(x) \vee (x \in I[C,t]) \qquad \text{and} \qquad f_t^{\langle 0\rangle}(x) := f(x) \wedge (x \notin I[C,t]).$$

Both functions can be computed in depth t.

Proof. From C construct a circuit C' of depth t as in the proof of Theorem 2, but now setting noninput gates at distance t to the constant 1. As before it holds that $C'(x) = f(x)$ for $x \notin I[C,t]$. Since all internal gates of C' are monotone for $x \in I[C,t]$ this circuit yields the value 1. This is due to the fact that the output gate of C depends on a path of length larger than t and in C' all such paths have been set to 1. Thus C' computes the function $f_t^{\langle 1\rangle}$.

A dual circuit for $f_t^{\langle 0\rangle}$ is obtained similarly, by setting gates at distance t to the constant 0. $\blacksquare$

The size of C' may grow exponentially with t. However, using the construction for Theorem 3 the blowup factor can be kept as small as t.

Let $\chi_{C,t}$ be the characteristic function for the complement of $I[C,t]$.

LEMMA 8. *$\chi_{C,t}$ can be computed in depth $t+3$.*

Proof. For $f_t^{\langle 0\rangle}$ and $f_t^{\langle 1\rangle}$ as defined above, it holds that $\chi_{C,t}(x) \Leftrightarrow (f_t^{\langle 0\rangle}(x) = f_t^{(1)}(x))$. Thus a circuit can be obtained from representation

$$\chi_{C,t}(x) = (\neg f_t^{\langle 1\rangle}(x) \wedge \neg f_t^{\langle 0\rangle}(x)) \vee (f_t^{\langle 0\rangle}(x) \wedge f_t^{\langle 1\rangle}(x)). \quad \blacksquare$$

Combining the results of the last two sections we get the following theorem.

THEOREM 4. *For any $t \in \mathbb{N}$ it holds*:

$$\mathrm{ECirTime}_n\left(t, \mathscr{D}\mathrm{Depth}\left(\frac{t+n}{2} + \sqrt{n-t} + \log n + 3\right)\right)$$

$$\subseteq \mathrm{CirDepth}_n(t + \log n + \log t + 3).$$

Proof. Let $\tau := \frac{1}{2}(t+n) + \sqrt{n-t} + \log(n+t) + 3$ and $f \in \mathrm{ECirTime}_n(t, \mathscr{D}\ \mathrm{Depth}(\tau))$. The strategy to compute f by a circuit of small depth is

1. Select $p \leqslant O(n \cdot t)$ circuits $C_1, ..., C_p$ from $\mathrm{Cir}(f)$ such that the sets $\boldsymbol{X_i} := \overline{I[C_i, t]} = \{x \mid \mathrm{time}_{C_i}(x) \leqslant t\}$ completely cover the input set $\{0,1\}^n$.

2. Using Lemma 7 find circuits $S_1, ..., S_p$ of depth at most t that compute the functions $f_i(x) := f(x) \vee [x \notin X_i]$.

3. $f(x) = \bigwedge_i f_i(x)$.

f is computed by combining the p circuits S_i by a binary tree yielding a circuit of total depth $t + \log p$.

To define X_1 let μ_1 be the uniform distribution over the whole input space $Z_1 := \{0,1\}^n$. Since, by assumption, $f \in \mathrm{ECirTime}_n(t, \mathscr{D}\mathrm{Depth}(\tau))$, there exists a circuit $C_1 \in \mathrm{Cir}(f)$ such that $E_{\mu_1}(\mathrm{time}_{C_1}) \leqslant t$. Define $X_1 := \{x \mid \mathrm{time}_{C_1}(x) \leqslant t\}$ and $Z_2 := Z_1 \setminus X_1$.

To define X_m consider $Z_m := Z_{m-1} \setminus X_{m-1}$. If $\log |Z_m| \leqslant (t+n)/2 + \sqrt{n-t}$ we define μ_m as a nearly Z_m-uniform distribution, otherwise as the $(Z_m, \ln(2t+1)\ 2n/|X|)$-uniform distribution. Let $C_m \in \mathrm{Cir}(f)$ have average complexity at most t with respect to μ_m. So, we obtain X_m as $\overline{I[C_m, t]}$.

Applying Lemma 4 and Lemma 5 distributions μ_m can be generated in depth τ. Using Lemma 6 we get

$$|X_m| \geqslant \frac{|Z_{m-1}|}{2(t+1)}; \qquad \text{thus,} \quad |Z_m| \leqslant |Z_{m-1}| \left(1 - \frac{1}{2(t+1)}\right).$$

This implies $|Z_m| \leqslant 2^n \cdot (1 - (2t+2)^{-1})^m$.

For $m = p := (t+1) \cdot n \cdot \ln 4$ this gives $|Z_p| < 1$. Since for all $m \leqslant p$ by construction $\{0,1\}^n = X_1 \cup \cdots \cup X_m \cup Z_m$ the sets $X_1, ..., X_p$ cover all inputs. Using the circuits $S_1, ..., S_p$ f can thus be computed within depth $t + \log t + \log n + 3$. ∎

COROLLARY 1. *For all Boolean functions $f \in \mathrm{CirDepth}_n(d)$ there exists a probability distribution μ_f computable in depth $\frac{1}{2}(n+d) + \sqrt{n-d} + O(\log n)$ such that with respect to μ_f all circuits for f have an expected delay of at least $d - \log n - \log d - 3$.*

9. CONCLUSION

These results, together with previous upper and lower bounds, provide a detailed understanding of average-case complexity in a nonuniform setting given by the Boolean circuit model. Our current knowledge is visualized in Fig. 4.

For simple probability distributions there are functions like `OR`, `AND`, `THRESHOLD`, and `ADDITION` with substantially smaller average-case complexity. On the other hand, for functions like `PARITY` it has been shown that average-case and worst-case complexity are asymptotically identical for any distribution [JRS94, JRSW94].

We have shown that most functions are hard in the average case, even for the uniform distribution; thus the Shannon effect also occurs in the average case. But the boundary is not as sharp as in the worst case. For every fixed distribution the

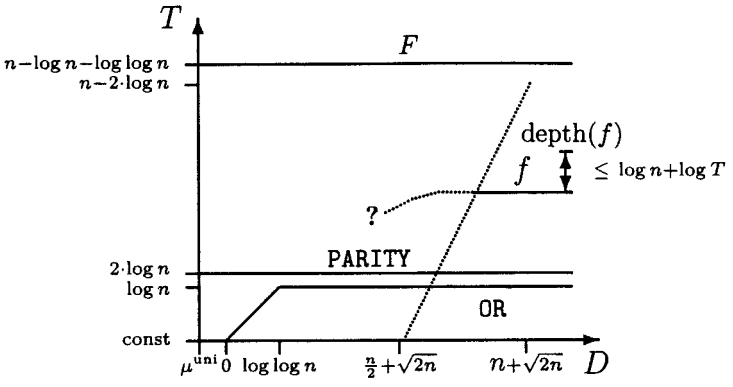

FIG. 4. Lower bounds for the average delay of OR, PARITY and an arbitrary function f with respect to the complexity of distributions as measured by the classes $\mathscr{D}\text{Depth}(D)$. Horizontally we have drawn the depth bound D and vertically the time complexity T. F denotes a functions of maximal asymptotic complexity as referred to in Theorem 1.

number of functions with constant expected delay is quite large; larger than the number of functions with a depth bound $n - 2\log n$.

Finally, we have shown that there is no function for which the circuit depth is substantially worse than the average behaviour for every distribution. But what is the threshold for the complexity of distributions to make the average-case complexity as hard as the worst-case complexity? From [JRS94] and Theorem 4 it follows that it is somewhere between $\log\log n$ and $n + \sqrt{2n}$. Narrowing this gap would yield a better understanding of average complexity for a broader class of functions and distributions.

Received January 15, 1997; final manuscript received October 7, 1998

REFERENCES

[BCGL92] Ben-David, S., Chor, B., Goldreich, O., and Luby, M. (1992), On the theory of average case complexity, *J. Comput. System Sci.* **44**, 193–219.

[DGY89] David, I., Ginosar, R., and Yoelli, M. (1989), An efficient implementation of Boolean functions and finite state machines as self-timed circuits, *ACM Comput. Architecture News*, 91–104.

[Gas78] Gaskov, (1978), The depth of Boolean functions, *Prob. Kybernet.* **34**, 265–268.

[Gra90] Grape, P. (1990), Complete problems with L-sampleable distributions, *in* "Proc. 2nd Scandinavian Workshop on Algorithm Theory," Springer Lec. Notes in Computer Science, pp. 360–367.

[Gur91] Gurevich, Y. (1991), Average case completeness, *J. Computer and System Sciences* **42**, 346–398.

[Jak98] Jakoby, A. (1998), "Die Komplexität von Präfixfunktionen bezüglich ihres mittleren Zeitverhaltens," dissertation, Med. Universität zu Lübeck, Shaker-Verlag, Aachen.

[JRS94] Jakoby, A., Reischuk, R., and Schindelhauer, C. (1994), Circuit complexity: From the worst case to the average case, *in* "Proc. 26th ACM Symposium on Theory of Computation," pp. 58–67.

[JRSW94] Jakoby, A., Reischuk, R., Schindelhauer, C., and Weis, S. (1994), The average case complexity of the parallel prefix problem, *in* "Proc. 21st International Colloquium, Automata, Languages and Programming," Springer Lec. Notes in Computer Science, pp. 593–604.

[JS96] Jakoby, A., and Schindelhauer, C. (1996), On the computational complexity of worst case and expected time in a circuit, *in* "Proc. 13th Annual Symposium on Theoretical Aspects of Computer Science," Springer Lec. Notes in Computer Science, pp. 295–306.

[Kob93] Kobayashi, K. (1993), On malign input distributions for algorithms, *IEICE Trans. Inform. Systems* **76**, 634–640.

[Kra78] Krapchenko, V. (1978), Depth and delay in a network, *Soviet Math. Dokl.* **19**, 1006–1009.

[LBS93] Lam, W., Brayton, R., and Sangiovanni-Vincentelli, A. (1993), Circuit delay models and their exact computation using timed Boolean functions, Proc. AGM/IEEE Design Automation Conference, pp. 128–133.

[Lev86] Levin, L. (1986), Average case complete problems, *SIAM Journal on Computing* **15**, 285–286.

[LV93] Li, M., and Vitanyi, P. (1993), "An Introduction to Kolmogorov Complexity and its Applications," Springer, New York.

[LV92] Li, M., and Vitanyi, P. (1992), Average case complexity under the universal distribution equals worst-case complexity, *Information Processing Letters* **42**, 145–149.

[Mil91] Miltersen, P. (1991), The complexity of malign ensembles, *in* "Proc. 6th Conference of Structure in Complexity Theory," pp. 164–171, see also *SIAM Journal on Computing* **22**, 1993, 147–156.

[RS96] Reischuk, R., and Schindelhauer, C. (1996), An average complexity measure that yields tight hierarchies, *Comput. Complexity* **6**, 133–173.

[Sch96] Schindelhauer, C. (1996), "Average- und Median-Komplexitätsklassen," dissertation, Med. Universität zu Lübeck.

[Weg87] Wegener, I. (1987), "The Complexity of Boolean Functions," Wiley–Teubner, New York.

[WB92] Wang, J., and Belanger, J. (1992), On average $\mathcal{P}$ vs. average $\mathcal{NP}$, *in* "Proc. 7th Conference on Structure in Complexity Theory," pp. 318–326.