Malign Distributions for Average Case Circuit Complexity

Andreas Jakoby, Rüdiger Reischuk, and Christian Schindelhauer

March 23, 1998

Medizinische Universität Lübeck
Institut für Theoretische Informatik, Wallstraße 40, 23560 Lübeck, Germany
email: jakoby / reischuk / schindel @ informatik.mu-luebeck.de

Proposed Running Text: Average Case Circuit Complexity

Abstract

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, the size and the depth are natural and simple measures for circuits and provide a worst case measure. We consider a model where an internal gate can be evaluated when the result is determined only by parts of its input. So we obtain a dynamic definition of delay. In [JRS94] we have defined an average case measure for the time complexity of circuits. Using this notion tight upper and lower bounds could be obtained for the average case complexity of several basic Boolean functions.

Here, we will examine the asymptotic average case complexity of the set of all n-ary Boolean functions. In contrast to worst case analyses a simple counting argument does not work. We prove that almost all Boolean function require at least $n - \log n - \log\log n$ expected time even for the uniform probability distribution. On the other hand, there are significant subsets of functions that can be computed with a constant average delay.

Finally, we compare worst case and average case complexity of Boolean functions. We show that for each function that is not computable by circuits of depth less than d, the expected time complexity will be at least $d - \log n - \log d$ with respect to an explicitely 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, asymptotic complexity of Boolean functions, 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 with respect to a uniform distribution over the input space as the average case measure. Levin 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]. One suggestion is to deal with broader classes of distributions, and it turned out that for this purpose a complexity measure for the distributions themselves was needed.

Levin's ideas have been further 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 hard problems – 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 the 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, even when fixing the problem size, the length of a computation in general depends on the specific input data, whereas the information flow in Boolean circuits is fixed. Thus, the depth of a circuit has typically been used as a worst case measure for the parallel time complexity. It is not obvious how to obtain a meaningful notion of time complexity for circuits that can be used for an average case analysis. 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 speedup 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 length see [Kra78].

In [JRS94] we have shown how this timing information can be used to define the notion of delay for circuit gates which in general is 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 if we compare average delay with circuit depth, most significantly for the addition of two binary numbers [JRS94]. On the other hand the parity function requires logarithmic delay even on the average. It is well known that the addition is basically equivalent to compute all prefixes of a linear formula over a specific semigroup. In [JRSW94] and more general in [Jak98] the average complexity of the parallel prefix problem for arbitrary semigroups has been investigated. By proving matching upper and lower bound it has been 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, as for the `OR`-function, or of order $\log \log n$, as for the addition, or of order $\log n$ as for the parity function.

For the Turing machine as well as for the circuit model the average complexity of computational problems 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 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 there has been time and space limits considered for a Turing machine in order to provide information about the 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 the time bounds of the machines as well as with respect to the complexity of the distributions.

Milterson has extended the result of [LV92] to subclasses $\mathcal{C}$ of machines with a fixed upper time bound. He calls a distribution *malign* for $\mathcal{C}$ if the average complexity of any machine in $\mathcal{C}$ is at most a constant factor smaller than its worst case complexity, and has proved the existence of such a distribution. This distribution is computable in exponential time and malign for the class $\mathcal{P}$ (see also Prop. 2 in [BCGL92] for a similar result). Furthermore, no distribution computable in polynomial time has this property if the probabilities of input strings do not decrease too fast to 0 with respect to their length. When computing a malign distribution for machines with a fixed polynomial time bound the exponential time bound can be improved to a polynomial time bounded generator with a Σ_2-oracle [Mil91], and in a slightly different model even to a generator with an $\mathcal{NP}$-oracle [RS96].

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

We will show that for a nonuniform model malign distributions do not exist for the class of all Boolean functions. A Boolean function exists for any type of distribution where 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 function 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 is 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 are significant subsets of functions that can be computed with a constant average delay.

The main technical contribution of this paper shows that for any Boolean function f with a given worst case complexity one can explicitely construct a distribution that is bad for all circuits realizing f. Their average case delay will be smaller by at most an additive term of order $\log n$ compared to the worst case delay. We will also determine the complexity of such distributions. In [RS96] the tradeoff between average time resources and complexity of distributions has been studied for the Turing machine model. In the circuit model, for several specific functions we have determined this tradeoff exactly for the whole range of distributions [JRS94]. In particular, the cutpoint where average and worst case complexity become identical has been located precisely in these cases. The results here solve the asymptotic question. They imply that allowing complex distributions for each function one can find distributions that make the average case complexity almost identical to the worst case complexity. 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 900, pp. 628-639).

2 An Average Case Measure for Circuits

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

Definition 1 *Let* $\boldsymbol{B_n^m}$ *denote the set of Boolean functions* $f : \{0,1\}^n \to \{0,1\}^m$*.* $\boldsymbol{B_n} := B_n^1$*.* $\boldsymbol{\mathcal{D}_n}$ *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. Let* $\boldsymbol{Cir(f)}$ *denote the set of all circuits that compute* f *and* $\mathbf{CirDepth}_{\boldsymbol{n}}\boldsymbol{(d)}$ *all functions in* B_n *that can be computed by a circuit of depth at most* d*.*

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

In a circuit information can be propagated immediately if, for example, one of the inputs of an `OR`-gate is already available and has the value 1. Then, its output is determined as 1 independent of the value of the other input. More formally, define a function time : $\{0,1\}^n \to \mathbb{N}$ for each gate v of a circuit C. It specifies for each input x the step when v can compute its result $\mathbf{res}_{\boldsymbol{v}}\boldsymbol{(x)}$ using the values of its predecessors.

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

$$\mathbf{time}_{\boldsymbol{v}}(\boldsymbol{x}) \;:=\; 1 + \min\{t \mid \textit{the values } \mathrm{res}_{v_i}(x) \textit{ with } \mathrm{time}_{v_i}(x) \leq t \textit{ uniquely determine } \mathrm{res}_v(x)\} \;.$$

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

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

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

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

Up to this point the notion of time has only been defined implicitly. In [JRS94] we have shown how this information can also be generated explicitly within the same delay increasing the circuit size by at most a constant factor. For a given Boolean function the worst case time complexity and depth complexity coincide. This follows by techniques shown in the proof of Theorem 2. But transforming circuits optimal with respect to the critical path (worst case) into depth optimal circuits leads to an increase of circuit size [Kra78]. On the other hand, we have shown that the computational complexity of determining or approximating the worst case delay of a circuit is co-$\mathcal{NP}$-hard, where its depth can be computed in $\mathcal{NC}$ [JS96]. Furthermore, the computation of time_C is complete for $\mathcal{P}$ and expected time (defined below) can be approximated in polynomial time.

Definition 3 *For a function $t : \{0,1\}^n \to \mathbb{N}$ and a probability distribution $\mu : \{0,1\}^n \to [0;1]$ let* $\mathrm{E}_\mu(\mathrm{t}) := \sum_{\mathrm{x}} \mathrm{t}(\mathrm{x})\, \mu(\mathrm{x})$ *be the expectation of t with respect to μ. If D is a set of probability distributions we define*

$$\mathbf{etime}(\boldsymbol{f}, \boldsymbol{D}) \;:=\; \max_{\mu \in D} \; \min_{C \in Cir_\mu(f)} \; \mathrm{E}_\mu(\mathrm{time}_\mathrm{C})$$

as the optimal **expected circuit delay** *of f with respect to distributions in D.*

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. In [JRS94] we have shown

$$\mathrm{etime}(\mathtt{OR}_n, \{\mu_{n,\mathrm{uni}}\}) \;=\; 2 - 2^{-(n-2)} \;.$$

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

$$\mathbf{ECirTime}_{\boldsymbol{n}}(\boldsymbol{t}, \boldsymbol{D}) \;:=\; \{f \in B_n \mid \mathrm{etime}(f, D) \leq t\} \;.$$

3 The Complexity of Distributions

For the average case analysis of circuits we also have to define a complexity measure for the distributions that generate the random inputs. The complexity will be measured by the circuit model itself: by the depth of a circuit. This circuit has an input vector of truly random bits and generates the specific distribution. This may be considered as the circuit analog of Turing machine sampleable.

Definition 4 *Let $C \in \mathrm{Cir}(B_r^n)$ 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 values obtained at the output gates.*

If Z is the uniform distribution over $\{0,1\}^r$ such a circuit will be called a **distribution generating circuit, DG-circuit***, which generates the distribution of X.*

In the following we will identify a distribution μ with a random variable X distributed according to μ. Let $X = X_1, \ldots, X_n$.

No interesting results can be obtained if we consider distributions generated by DG-circuits with unbounded fan-out, which means that every output may depend on a single random bit.

Consider a DG-circuit with random inputs $x_0, \ldots, x_n$ computing $x_0 \wedge x_1$, $x_0 \wedge x_2$, $\ldots$, $x_0 \wedge x_n$. Now 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 the $\mathtt{OR}_n$, for example, 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 [JRS94]. 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.

With the notion of DG-circuits we classify distributions as follows

Definition 5 $\boldsymbol{\mathcal{D}}\mathbf{Depth}_{\boldsymbol{n}}\boldsymbol{(d)}$ $:=$
$\{\mu \in \mathcal{D}_n \mid \exists$ DG-circuit $C \in \mathrm{Cir}(B_n^r):\ \mathrm{depth}(C) \leq d \ \wedge\ C(\mu_{\mathrm{uni},r}) = \mu\}$.

Using this notion, the following classifications for the average complexity of some basic Boolean functions can be given [JRS94].

$$\begin{aligned}
\mathrm{etime}(f, \mathcal{D}\mathrm{Depth}(d)) &= \Theta(\min(2^d, \log n)) \quad f \in \{\mathtt{OR}_n,\, \mathtt{AND}_n,\, \mathtt{EQUAL}_n\}\ , \\
\mathrm{etime}(f, \{\mu_{n,\mathrm{uni}}\}) &= \Theta(\log n) \qquad\qquad\qquad f \in \{\mathtt{PARITY}_n,\, \mathtt{MAJ}_n\}\ , \\
\mathrm{etime}(\mathtt{ADD}_n, \mathcal{D}\mathrm{Depth}(d)) &= \Theta(\min(\log\log n + 2^d, \log n))\ , \\
\mathrm{etime}(\mathtt{THRESH}_n^m, \mathcal{D}\mathrm{Depth}(d)) &= \Theta(\min(\log a + 2^d, \log n))\ .
\end{aligned}$$

Here $\mathtt{AND}_n$ denotes the n-ary conjunction, $\mathtt{EQUAL}_n$ the test for equality of 2 binary strings of length n and $\mathtt{ADD}_n$ the addition of 2 binary numbers of length n. $\mathtt{THRESH}_n^m \in B_n^m$ is 1 if at least m inputs are 1 and $\mathtt{MAJ}_n := \mathtt{THRESH}_n^{\lceil n/2 \rceil}$.

4 Circuits Do Not Have Malign Distributions

For any probability distribution there is a non-trivial disjunction depending on only a small subset of variables that yields 1 with high probability. Let $x, \alpha \in \{0,1\}$, and x^α equal x if $\alpha = 1$, and $\neg x$ else.

Lemma 1 *Let $X = x_1, \ldots, x_n$ be a random variable on $\{0,1\}^n$ with an arbitrary distribution and let $\{i_1, i_2, \ldots, i_l\} \subseteq [1..n]$. Then there exists Boolean constants $\alpha_1, \ldots, \alpha_l$ such that* $\Pr[\mathtt{OR}(x_1^{\alpha_1}, \ldots x_l^{\alpha_l}) = 1] \ \geq \ 1 - 2^{-l}$.

Proof: by induction over the length of the input vector:
Let $\alpha_1 = 1$ if $\Pr_\mu[x_1 = 1] \geq \frac{1}{2}$, and $\alpha_1 = 0$ else. It is clear that for all $\alpha_1, \ldots, \alpha_k$ there exists a constant $\alpha_{k+1} \in \{0,1\}$ such that $\Pr_\mu[x_{k+1}^{\alpha_{k+1}} = 1 \mid x_1^{\alpha_1}, \ldots, x_k^{\alpha_k} = 0] \ \geq \ 1/2$ or $\Pr_\mu[x_1^{\alpha_1} \vee \ldots \vee x_k^{\alpha_k}] = \ 1$.
Let $\Pr_\mu[x_1^{\alpha_1} \vee \ldots \vee x_k^{\alpha_k}] \geq 1 - 2^{-k}$, then

$$\begin{aligned} \Pr_\mu[x_1^{\alpha_1} \vee \ldots \vee x_{k+1}^{\alpha_l}] &= (1 - \Pr_\mu[x_1^{\alpha_1} \vee \ldots \vee x_k^{\alpha_k}])/2 \ + \ \Pr_\mu[x_1^{\alpha_1} \vee \ldots \vee x_k^{\alpha_k}] \\ &\geq 1 - 2^{-k-1} \ . \end{aligned}$$

∎

Using this property we can prove

Theorem 1 *For every probability distribution μ over $\{0,1\}^n$ there exists an n-ary Boolean function f with*

$$\mathrm{depth}(f) \ \geq \ n - \log n - \log\log n \quad \text{and} \quad \mathrm{etime}(f, \mu) \ \leq \ 4 \ .$$

Proof: For given μ, the function f is defined as follows. Let $l := \log n$. With respect to μ choose $\alpha_1, \ldots, \alpha_l$ according to the lemma above. Choose $g \in B_{n-\log n}$ with $g \notin \mathrm{CirDepth}(n - \log n - \log\log n - 1)$. Due to the Shannon bound such a g exists (Fact 1). Let $f(x_1, \ldots, x_n) \ := \ \mathtt{OR}(x_1^{\alpha_1}, \ldots, x_l^{\alpha_l}, g(x_{l+1}, \ldots, x_n)\ )$.

To prove the lower bound assume $\mathrm{depth}(C) < n - \log n - \log\log n$ and $C \in \mathrm{Cir}(f)$. Replace input vector $(x_1, \ldots, x_l)$ by the constant inputs $(\neg\alpha_1, \ldots, \neg\alpha_l)$, obtaining circuit C'. Obviously, $\mathrm{depth}(C') = \mathrm{depth}(C)$ and $C' \in \mathrm{Cir}(g)$ holds — contradicting the Shannon bound.

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. $C_{\mathtt{OR},n}(x_1, \ldots, x_n) := \mathtt{OR}(x_n, C_{\mathtt{OR},n-1}(x_1, \ldots, x_{n-1}))$. 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

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

∎

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

In the last proof only bits $\alpha_1, \ldots, \alpha_{\log n}$ 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 **t-bad input set** of C by

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

It is easy to see that $|I[C,t]| \ \leq \ \frac{2^n}{t+1} \ \mathrm{E}_{\mu_{\mathrm{n,uni}}}(\mathrm{time_C})$.

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 $d := \lfloor n - \log_2 n - \log_2 \log_2 n - 1 \rfloor$. Assume $f \in B_n$ can be computed by an expected d-time bounded circuit C with respect to the uniform distribution. In order to restrict internal gates to `AND` and `OR` we allow that input gates of C may also be given by the negated variables $\overline{x_i}$. Furthermore, without increasing the depth C can be expanded to a circuit C' with fanout 1.

We extend the Boolean domain by a new symbol "?". The `OR` and `AND` function are now defined by table 1.

`OR`	?	0	1
?	?	?	1
0	?	0	1
1	1	1	1

`AND`	?	0	1
?	?	0	?
0	0	0	0
1	?	0	1

Table 1: Extended versions of the Boolean functions `OR`, `AND`.

Let us cut off all gates with distance larger than d from the output gate and replace non-input 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 $\exp(2^d(1+\log(2n+1)))$ 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 $\mathrm{res}_C(x) = \mathrm{res}_{C'}(x)$. 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

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

Since there are 2^{2^n} Boolean n-ary functions 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 to the uniform distribution. The portion of such in the average efficiently computable functions grows if we increase the delay bound. Compare this result 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 Efficient Selection

Efficient DG-circuits are necessary in order to construct bad distributions of low complexity. Because of the fan-out restriction this problem is not that easy. In the following, we will develop general techniques for this task.

Let us denote the i-th bit of a binary string a by $a[i]$. The function $\mathrm{bin(i,n)}$ for integer $i \leq 2^n - 1$ denotes the binary representation of i of length n, and $\mathrm{bin}^{-1}\mathrm{(a)}$ its reverse.

For the following constructions the multiplexer function plays an important role.

Definition 6 *For given input vectors $x_0, \ldots, x_{m-1}$ each of length n and control vector $y \in \{0,1\}^{\log m}$ a* **(m,n)-multiplexer (MX)** *outputs $x_{\mathrm{bin}^{-1}(\mathrm{y})}$, i.e. the y-th input, where y is interpreted as a binary number.*

With unbounded fanout the depth of a (m,n)-MX-circuit can be bounded by $\log m + \log(\log m + 1) + 1$, since for the output vector z holds

$$z[k] \quad := \quad \bigvee_{i=0}^{m-1} \left(x_i[k] \wedge \bigwedge_{j=1}^{\log m} y_j^{\mathrm{bin(i,\log m)[j]}} \right) \quad , \qquad \text{for all } k \leq n .$$

Every control bit y_j is used $m \cdot n$ times.

To obtain a circuit with constant fan-out we can 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 $2\log m + \log(\log m + 1) + \log n + 1$ by a circuit with fanout 2.*

We can construct more efficient multiplexers by using unary coding for the control input.

Definition 7 *For given input vectors $x_1, \ldots, x_m \in \{0,1\}^n$ and control inputs $y_1 \ldots y_m \in \{0,1\}$ 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.*

Lemma 2 *A unary controlled (m,n)-multiplexer can be computed by a circuit of depth $\log m + \lceil \sqrt{8 \log m} \rceil + \log n + 1$.*

Proof: The $\mathtt{UCMX}_{m,n}$ circuit to be constructed consists of n parallel $\mathtt{UCMX}_{m,1}$ circuits. For the initial distribution of the control bits $y_0, \ldots, y_{m-1}$ we use fanout trees of depth $\log n$. What remains to be described is the construction of such a $\mathtt{UCMX}_{m,1}$.

First, in one parallel step we compute $x'_i := x_i \wedge y_i$ for all $i < m-1$ and $x'_{m-1} := x_{m-1}$. So an unused data bit is set to 0. Note that $\mathtt{UCMX}_{m,1}(\vec{x}, \vec{y}) = \mathtt{UCMX}_{m,1}(\vec{x'}, \vec{y})$.

The simple construction of a $\mathtt{UCMX}$-circuit S_m yields the result $r = \mathtt{UCMX}_{m,1}(\vec{x'}, \vec{y})$ as follows (see figure 1):

$$\begin{aligned} \forall j < m \qquad s_j &:= x'_j \wedge \neg \bigvee_{i<j} y_i \;, \\ r &:= \bigvee_{j<m} s_j \;. \end{aligned}$$

Since the control inputs y_i are needed up to m times each, we have to use fanout trees of depth $\log m$. So the whole depth of the construction is $3 \log m + 2$.

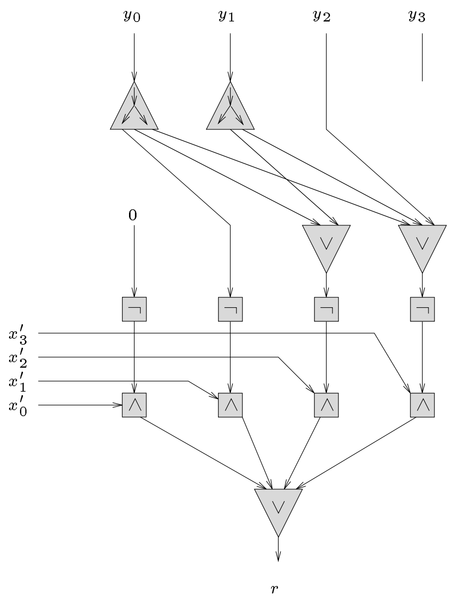

Figure 1: The $\mathtt{UCMX}_{4,1}$-circuit S_4.

Note that the inputs x' are used in depth $2 \log m + 1$ for the first time. Further, unary controlled multiplexer can be combined such that the 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 $\vec{x}$ 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 is called r_λ.

The following definition gives a brief and complete description of the whole construction (see fig. 2), which is somehow a pipelined version of a lot of S_δ-circuits.

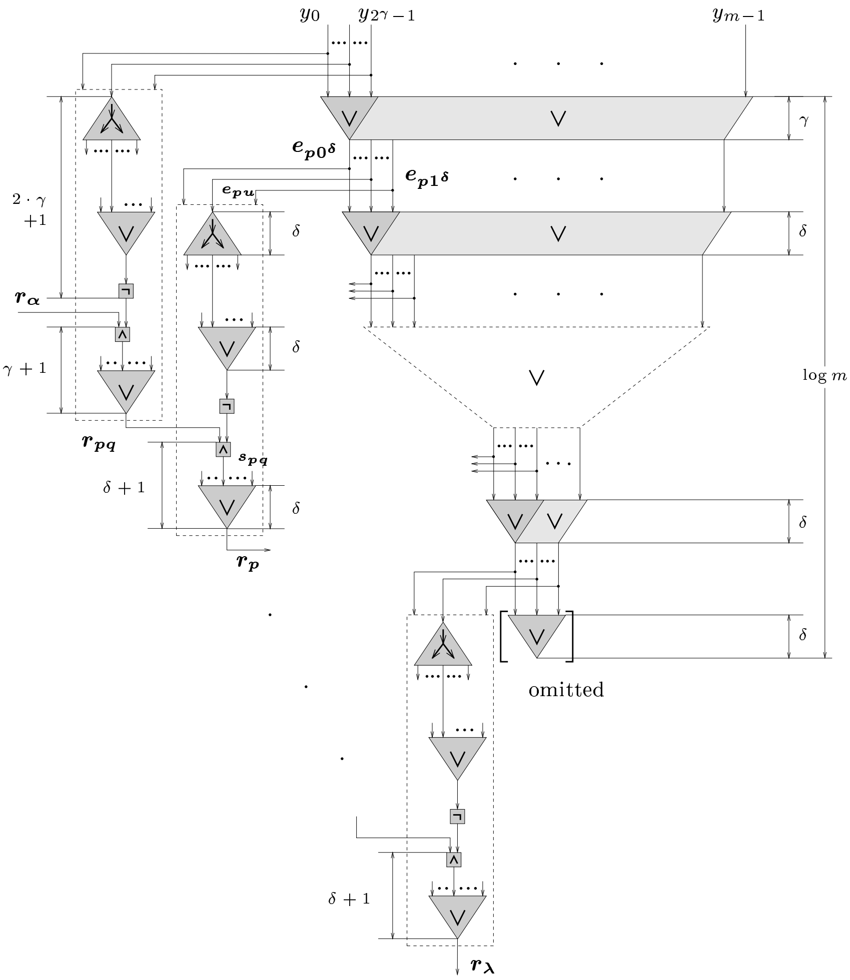

Figure 2: A circuit for the unary controlled multiplexer.

For $\delta \leq \log k$ define

$$\forall p \in \{0,1\}^{\leq \log m - 1} \quad e_p \quad := \quad \bigvee_{q \in \{0,1\}^{\log m - |p|}} y_{pq} \ ,$$

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

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

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

Correctness: Fix ℓ and $p \in \{0,1\}^{\delta\ell}$. We will prove that

$$r_p = \mathtt{UCMX}_{m,1}(r_{p0^\delta}, \ldots, r_{p1^\delta}, e_{p_{0^\delta}}, \ldots, e_{p_{1^\delta}}) \ .$$

The term $\bigvee_{u\in\{0,1\}^\delta \text{ and } u<q} e_{pu}$ indicates whether at least one bit $y_{pw} = 1$, where $w \le q$. If the term is 1, this 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 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} and $x_{p1^{|w|}}$ if there is no control input.

Efficiency: The computation of e_p for all p can be done by an 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 $x_0 \ldots x_{m-1}$ has thickness $\gamma \ := \ (\log m - 1) \bmod \delta + 1$. Here e_p for $p \in \{0,1\}^\gamma$ and $r_{\mathrm{bin(i,\log k)}}$ for $i \le k$ 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 3 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 only one level. This also holds for the inputs e_{pu} and outputs e_{pq}. Therefore, the subcircuit computing r_p can be placed one level below those of 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

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

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

The next fundamental problem is how to construct a random variable uniformly distributed over a given set A by a DG-circuit.

Definition 8 *Let m be a power of 2. For given input vectors $x_0, \ldots, x_{m-1} \in \{0,1\}^n$ and uniformly distributed random input bits a* **random (m,n)-selector** ($\mathtt{RS}_{m,n}$) *outputs a with probability $|\{j \mid x_j = a\}|/m$.*

Lemma 3 *A random (m,n)-selector can be implemented by a $\log m + \log(\log m + 1) + \log n + 3$ depth bounded DG-circuit.*

Proof: Define a $\mathtt{RS}_{l,n}$ circuit C_l using the multiplexer circuit described in Fact 2. The control input is given by random bits $r_1, \ldots, r_{\log l}$. Because of the fanout restriction this input is delivered in depth $\log l + \log n$ to the subcircuits $S_{i,l}$ computing $\bigwedge_{j=1}^{\log l} r_j^{\mathrm{bin(i,\log l)}}$ for all i. Finally, we compute the disjunction of all l partial results $S_{i,l} \wedge x_i[k]$. The total depth of the random selector C_l is $2\log l + \log\log l + \log n + 2$.

Taking a closer look one notices that the inputs x_i are used for the first time in depth $\log l + \log\log l + \log n + 1$. Since the depth of $C_{l'}$ for $l' \leq \sqrt{l}$ is larger than this number, we can combine l circuits $C_{l'}$ with a circuit C_l. This construction generates a distribution equivalent to that of a $\mathtt{RS}_{l \cdot l',n}$ circuit using the same depth as a C_l circuit. Of course, the same works for all $C_{l'}$-circuits.

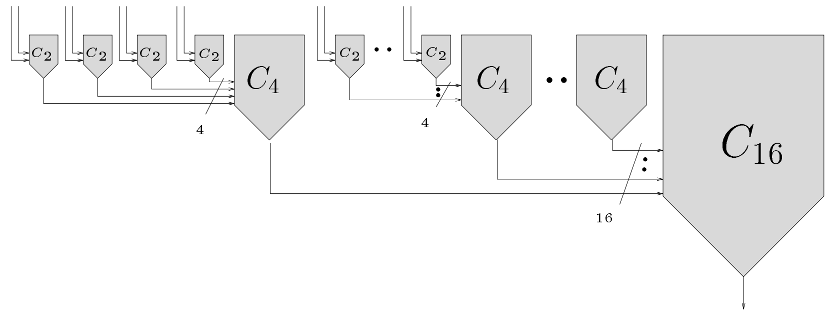

Figure 3: Combining random selector circuits of type C_2, C_4, C_{16} yields a $\mathtt{RS}_{2\cdot 4\cdot 16,n}$-circuit with the same depth as C_{16}.

Recursively, we define the final circuit by joining circuits C_{2^1}, C_{2^2}, $C_{2^4}, \ldots, C_{2^{2^p}}$ to a $\mathtt{RS}_{\alpha,n}$ circuit E (see figure 3), and add circuits of this type to a $\mathtt{RS}_{\beta,n}$-circuit C_β, for $p := \log\log(2m) - 2$, $\alpha := \exp(1 + 2 + 4 + \ldots + 2^p)$ and $\beta := \left\lceil \frac{2m}{2^{2^{p+1}}} \right\rceil$. By construction all elements of A can be selected, since $\alpha \cdot \beta \geq m$. The depth of E is $2 \cdot 2^p + p + 2 + \log n$. Within this depth all random bits of C_β can be distributed, since $2^{2^p} \geq \sqrt{\beta}$. After selecting the inputs by some E-circuits additional depth $\log 2m - 2^{p+1} + 1$ is needed by circuit C_β. Therefore, the total depth is given by $\log m + \log(\log m + 1) + \log n + 3$. ∎

7 Constructing Hard Distributions

Although we have proven that most functions have large average case complexity even with respect to the uniform distribution we do not have any specific example. Simply from the property that a function has large worst case delay, let's say larger than $2\log n$, we cannot obtain much knowledge concerning its expected behaviour for the uniform distribution.

But it will be shown in the following that for every such function f there exist distributions μ_f that make 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. Since there is little known about distribution generating circuits, we introduce a special family of distributions. Each one gives weight to a particular set of the input domain and zero weight to the complement. A collection of such distributions, which can be approximated by different circuit types, will form the final distribution μ_f. Note that contrary to the linear depth bound for Boolean functions there is no a priori bound known for the depth of circuits that generate arbitrary distributions.

A lot of effort will be devoted in the following construction to keeping the complexity of the distribution μ_f as small as possible. It is not surprising that the complexity of μ_f will 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* **probability distribution μ_A** *by*

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

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 $a \in \mathbb{N}$ by $2^a \leq |A| < 2^{a+1}$. Probability distributions μ are called **nearly A-uniform** *iff $\mu(x)$ for $x \in A$ is either 2^{-a} or $2^{-(a+1)}$.*

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 $\log|A| + \log(\log|A|+1) + \log n + 3$.*

Proof: Let $A = \{a_1, \ldots, a_{|A|-1}\}$ and $m := \min\{2^k \mid 2^k \geq |A|\}$. We use a $\mathtt{RS}_{m,n}$-circuit and fix the input vectors by $x_i := a_i \bmod |A|$ for all $0 \leq i < m$. By Lemma 3 the depth of such a circuit C can be bounded by $\log|A| + \log(\log|A|+1) + 1$. ∎

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 10 *The* **(A,k)-uniform distribution** *is given by*

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

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

Proof: We use a $\mathtt{UCMX}_{k,n}$-circuit with n-bit random input strings $x_0, \ldots, x_{k-1}$. In parallel all strings are tested for membership in A using a circuit for f of depth d. The test results give the control inputs y_i.

The unary controlled multiplexer circuit outputs the string of the first successful test. In case that no test succeeds it outputs the last string.

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 $\overline{A}$. Thus the construction generates a (A, k)-uniform distribution. ∎

These almost uniform 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 $k \geq \ln(2t + 1)\ 2^n |A|^{-1}$. Further let $h : \{0, 1\} \to \mathbb{N}$ be a function with expectation bounded by t, that is $\mathrm{E}_\mu(\mathrm{h}) \leq \mathrm{t}$. Then for the set $\boldsymbol{A[h,t]} \ := \ \{x \in A \mid h(y) \leq t\}$ holds*

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

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

With respect to the bound on $\mathrm{E}_\mu(\mathrm{h})$ the set $A[h, t]$ is smallest possible, if for all $x \in A[h, t] \quad h(x) = 0$ and for all $x \in A \setminus A[h, t] \quad h(x) = t + 1$.

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

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

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

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

$$\begin{array}{rrcl} & t \cdot (2 \cdot |A_0| + |A_1|) & \geq & (|A_1 \setminus A[h, t]|\ (t + 1) \ = \ (|A| - |A[h, t]|)\ (t + 1) \\ \Longrightarrow & (t + 1) \cdot |A[h, t]| & \geq & (t + 1) \cdot |A| - t \cdot (2 \cdot |A_0| + |A_1|) \ \geq \ |A| - t \cdot |A_0| \\ & & \geq & |A| \cdot (1 - t/(2t + 2)) \ \geq \ |A|/2 \ . \end{array}$$

Now consider (A, k)-uniform distributions μ. Since $k \geq \ln(2t + 1)\ 2^n\ |A|^{-1}$ for $x \in A$ holds:

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

Then, we can conclude

$$
\begin{aligned}
t \;&\geq \sum_{x \in A\setminus A[h,t]} (t+1)\,\mu(x) \;\geq\; (|A| - |A[h,t]|)\,(t+1)\,\nu \qquad \text{and} \\
|A[h,t]| \;&\geq\; |A| - \frac{t}{\nu\cdot(t+1)} \;=\; |A|\left(1 - \frac{t(2t+1)}{(t+1)2t}\right) \;=\; \frac{|A|}{2(t+1)}\ . \qquad \blacksquare
\end{aligned}
$$

For a comparison of average time and depth of a circuit we have to transform time efficient circuits, which may be arbitrarily deep, into circuits with small depth, but producing erroneous results. The notion $I[C,t]$, the set of t-bad inputs of a circuit C, will help us to control the quality and quantity of errors.

Lemma 7 *Let $C \in \mathrm{Cir}(f)$ and $t \in \mathbb{N}$. Define functions $f_t^{\langle 1\rangle}$ and $f_t^{\langle 0\rangle}$ by $f_t^{\langle 1\rangle}(x) := f(x) \vee [x \in I[C,t]\ ]$, $f_t^{\langle 0\rangle}(x) := f(x) \wedge [x \notin I[C,t]\ ]$. 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 non-input gates at distance t to the constant 1. As before it holds $C'(x) = f(x)$ for $x \notin I[C,t]$. Since all internal gates of C' are monotone it is not hard to see that 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 path 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. ∎

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 $f_t^{\langle 1\rangle}$ defined in lemma 7 it holds $\chi_{C,t}(x) \Leftrightarrow (f_t^{\langle 0\rangle}(x) = f_t^{\langle 1\rangle}(x))$. So the circuit is given by

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

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

Theorem 3 *For any $t \in \mathbb{N}$ holds:*

$$
\begin{aligned}
&\mathrm{CirETime}\left(t, \mathcal{D}\mathrm{Depth}\left(\frac{t+n}{2} + \sqrt{n-t} + \log n + 3\right)\right) \\
&\qquad \subseteq\ \mathrm{CirDepth}(t + \log n + \log t + 3)\ .
\end{aligned}
$$

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

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

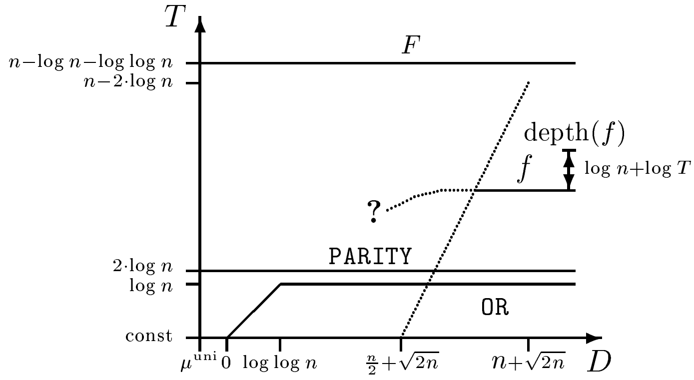

Figure 4: Lower bounds for the average complexity of OR, PARITY and an arbitrary function f with respect to the complexity of distributions $\mathcal{D}\mathrm{Depth}(D)$. F denotes a functions of maximal asymptotic complexity as described in Theorem 1.

2. Using Lemma 7 one can find circuits $S_1, \ldots, S_p$ of depth at most t that compute the functions $f_i(x) \ := \ f(x) \vee [x \notin X_i]$. Obviously, $f(x) = \bigwedge_i f_i(x)$. Thus f can be computed by combining the p circuits S_i by a binary tree yielding a circuit of total depth $t + \log p$.

For a definition of X_1 let μ_1 be the uniform distribution over the whole input space $Z_1 := \{0,1\}^n$. Since by assumption $f \in \mathrm{CirETime}(t, \mathcal{D}\mathrm{Depth}(\tau))$, there exists a circuit $C_1 \in \mathrm{Cir}(f)$ such that $\mathrm{E}_{\mu_1}(\mathrm{time}_{\mathrm{C}_1}) \ \leq \ \mathrm{t}$. Define $X_1 := \{x \mid \mathrm{time}_{C_1}(x) \leq t\}$ and $Z_2 := Z_1 \setminus X_1$.

To define X_m consider $Z_m := Z_{m-1} \setminus X_{m-1}$. If $\log |Z_m| \ \leq \ (t+n)/2 + \sqrt{n-t}$ we define μ_m as nearly Z_m-uniform distribution, otherwise as $(Z_m, \ln(2t+1)\frac{2^n}{|X|})$-uniform distribution. Let $C_m \in \underline{\mathrm{Cir}(f)}$ 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

$$|X_m| \ \geq \ \frac{|Z_{m-1}|}{2(t+1)} \qquad \text{and thus} \qquad |Z_m| \ \leq \ |Z_{m-1}| \left(1 - \frac{1}{2(t+1)}\right) .$$

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

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

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

8 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`, `THRESH` and `ADD` [JRS94] with substantially smaller average case complexity. On the other hand for other functions like `PARITY` it has been shown in [JRS94, JRSW94] that average case and worst case complexity are asymptotically identical for any distribution.

We have shown that most functions are hard in the average case even for the uniform distributions, thus the Shannon effect also applies for the average case. But the boundary is not as sharp as in the worst case. For every fixed distribution the number of functions with constant expected delay is quite large (larger than the number of functions with depth at most $n - 2\log n$).

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 3 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.

References

[BCGL92] Ben-David, S., Chor, B., Goldreich, O., and Luby, M. (1992), On the Theory of Average Case Complexity, *Journal of Computer and System Sciences*, Volume 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 Computer Architecture News*, 91-104.

[Gas78] Gaskov, (1978) The Depth of Boolean Functions, *Prob. Kybernet.*, Volume 34, 265-268.

[Gra90] Grape, P. (1990), Complete Problems with L-sampleable Distributions, *in* "Proceedings 2nd Scandinavian Workshop on Algorithm Theory", 360-367.

[Gur91] Gurevich, Y. (1991), Average Case Completeness, *Journal of Computer and System Sciences*, Volume 42, 346-398.

[Jak98] Jakoby, A. (1998), "Die Komplexität von Präfixfunktionen bezüglich ihres mittleren Zeitverhaltens," dissertation, Medizinische Universität zu Lübeck.

[JRS94] Jakoby, A., Reischuk, R., Schindelhauer, C. (1994), Circuit Complexity: From the Worst Case to the Average Case, *in* "Proceedings 26th ACM Symposium on Theory of Computation", 58-67.

[JRSW94] Jakoby, A., Reischuk, R., Schindelhauer C., Weis, S. (1994), The Average Case Complexity of the Parallel Prefix Problem, *in* "Proceedings 21st International Colloquium, Automata, Languages and Programming", 593-604.

[JS96] Jakoby, A., Schindelhauer, C. (1996) , On the Computational Complexity of Worst Case and Expected Time in a Circuit, *in* "Proceedings 13th Annual Symposium on Theoretical Aspects of Computer Science", 295-306.

[Kob93] Kobayashi, K. (1993), On Malign Input Distributions for Algorithms, *IEICE Transaction on Information and Systems*, Volume 76, 634-640.

[Kra78] Krapchenko, V. (1978), Depth and Delay in a Network, *Soviet Math. Dokl.*, Volume 19, No. 4, 1006-1009.

[LBS93] Lam, W., Brayton, R., Sangiovanni-Vincentelli, A. (1993), Circuit Delay Models and Their Exact Computation Using Timed Boolean Functions, *in* "Proceedings ACM/IEEE Design Automation Conference," 128-133.

[Lev86] Levin, L. (1986), Average Case Complete Problems, *SIAM Journal on Computing*, Volume 15, 285-286.

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

[LV92] Li, M., Vitanyi, P. (1992), Average Case Complexity under the Universal Distribution Equals Worst-Case Complexity, *Information Processing Letters*, Volume 42, 145-149.

[Mil91] Miltersen, P. (1991) , The Complexity of Malign Ensembles, *in* "Proceedings 6th Conference of Structure in Complexity Theory", 164-171, *see also* (1993) *SIAM Journal on Computing*, Volume 22, 147-156.

[RS96] Reischuk, R., Schindelhauer, C. (1996), An Average Complexity Measure that Yields Tight Hierarchies, *Comput. Complexity*, Volume 6, 133-173.

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

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

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