Medizinische Universität Lübeck[1]

Malign Distributions for
Average Case Circuit Complexity

(extended abstract)

Andreas Jakoby, Rüdiger Reischuk and Christian Schindelhauer

Abstract

In contrast to machine models like Turing machines or random access machines, circuits are a rigid computational model. The internal information flow of a computation is fixed in advance, independent of the actual input. Therefore, in complexity theory only worst case complexity measures have been used to analyse this model. In [JRS 94] 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.

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

Finally, worst case and average case complexity of Boolean will be compared. 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. We obtain a nontrivial bound on the complexity of such a distribution.

Classification: computational complexity, parallel algorithms.

1 Introduction

Complexity theory is traditionally based on worst case measures. Until recently, the average amount of resources necessary to solve a computational problems has been analysed only in a few cases. The average case measure considered there is mostly the expectation with respect to a uniform distribution over the input space. Levin observed that this simple analysis has serious drawbacks from a complexity theoretic point of view, and made a proposal how these difficulties could be overcome [Levi86]. One particular point 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 [Gure91, BCGL92, ReSc93] to define various average case complexity classes based on the time complexity of Turing machine computations. This was motivated by the question whether computationally hard problems – that means problems difficult in the worst case – might efficiently be solvable at least on the average (see also the discussion in [WaBe92]). However, it was shown that certain $\mathcal{NP}$-complete problems are likely to remain infeasible on the average. As in the worst case analysis this was 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

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

every $\mathcal{NP}$-complete problem has polynomial average case complexity with respect to any distribution that can be computed in polynomial time [Levi86].

In [JRS 94] we have developed an average case measure for parallel time, which is based on the Boolean circuit model. 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 circuits is fixed. Thus, it is not obvious how an average case consideration could decrease computational resources in the circuit model. 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 how 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 (see also [Gask78]).

We have shown how this timing information can be used to define the notion of delay for a circuit with respect to each input vector. This way, one obtains a meaningful average case measure for the circuit delay. 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 we have obtained an exponential speedup going from the worst case to the average delay measure, most significantly for the addition of two binary numbers [JRS 94]. On the other hand the parity function has been proved to require 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 [JRSW 94] we have investigated the average case complexity of the parallel prefix problem for arbitrary semigroups. By proving matching upper and lower bound it is shown that the complexity depends only on algebraic properties of the semigroup and that only two different situations are possible. The average delay is either of order $\log\log n$ as for the addition, or of order $\log n$ like for the parity function.

For the Turing machine as well as for the circuit model the average case complexity of computational problems depends to some extent on the set of distributions that may occur. Restricting an average case analysis to just the uniform distribution is of limited value. But it has been observed that allowing arbitrary distributions the average case complexity equals the worst case complexity for uniform computational models. Li/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 [LiVi92], see also [Koba93]. This distribution is closely related to the Kolmogorov complexity of strings [LiVi89], 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 one has considered time and space limits for a Turing machine to provide information about the individual distribution. The notion *computable* was proposed by Levin [Levi86]. It requires that the corresponding distribution function can be approximated in polynomial time. A weaker notion based on probabilistic machines called *sampleable* has been investigated by Ben-David/Chor/Goldreich/Luby in [BCGL92]. Reischuk/Schindelhauer have defined another natural notion called *rankable* [ReSc93], 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 [LiVi92] to subclasses $\mathcal{C}$ of machines with a fixed upper time bound. He calls a distribution with the property that the average case complexity of any machine in $\mathcal{C}$ is at most a constant factor smaller than its worst case complexity *malign* for $\mathcal{C}$. His main result in [Milt91] is that there exists a distribution computable in exponential time that is malign for the class $\mathcal{P}$ of all polynomial time bounded machines (see also Prop. 2 in [BCGL92] for a similar result). Furthermore, no distribution computable in polynomial time has this property if we make the technical assumption that the probabilities of the 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 [Milt91], and in a slightly different model even to a generator with an $\mathcal{NP}$-oracle [ReSc93].

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 [Grap90] 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. For any distribution there exist Boolean functions whose average delay is significantly smaller than their 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 [Gask78]. The situation for the average delay is more complicated. For a large portion of functions we can almost achieve this 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 result of this paper that for any Boolean function f with a given worst case complexity one can explicitely construct a distribution that is malign 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 [ReSc93] the tradeoff between average time resources and complexity of distributions has been studied for the Turing machine model. In the circuit model, for specific functions like the `OR` or `PARITY` the tradeoff could be determined for the whole range of distributions exactly [JRS 94]. 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.

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* $\boldsymbol{B_n}$ *denote the set of Boolean functions* $f : \{0,1\}^n \to \{0,1\}$. $\boldsymbol{\mathcal{D}_n}$ *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* $\mu_{n,\text{uni}}$.

Circuits will be defined over the standard basis of `AND`, `OR` *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 *[Weg87, Gask78]* $B_n = \mathit{CirDepth}_n(n - \log\log n + O(1))$. *This depth bound is best possible (up to a small additive term) for almost all functions in* B_n.

Information can be propagated faster in a circuit 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, we 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 $time_v(x) := 0$. For an internal nonconstant gate v with k direct predecessors $v_1, \ldots, v_k$ define*

$$\boldsymbol{time_v(x)} \quad := \quad 1 \;+\; \min\{t \mid \begin{array}{l} \textit{the values } res_{v_i}(x) \textit{ with } time_{v_i}(x) \leq t \\ \textit{uniquely determine } res_v(x)\} \; . \end{array}$$

For the circuit C itself with output gates $y_1, \ldots, y_m$ we define the global time function by $\boldsymbol{time_C(x)} := \max_i \; time_{y_i}(x)$.

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

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

This notion of time is so far defined only implicitly. In [JRS 94] 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.

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

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

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 OR-function. In [JRS 94] we have shown

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

The complexity class of all functions that can be computed within expected time at most t with respect a distribution in D will be denoted by

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

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, that is by the complexity of a circuit that starting with a vector of truly random bits generates the specific distribution. This may be considered as the circuit analog of Turing machine sampleable.

Definition 4 *A circuit C with r input gates and n output gates performs 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***, that generates the distribution of X.*

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

$$\boldsymbol{\mathcal{D}}\mathbf{Depth}_{\boldsymbol{n}}(\boldsymbol{d}) := \{\ \mu \in \mathcal{D}_n \mid \exists \textit{ an } r\textit{-input and } n\textit{-output DG-circuit } C \textit{ of depth } d \textit{ that transforms a random variable } Z \textit{ over } \{0,1\}^r \textit{ with distribution } \mu_{r,\mathrm{uni}} \textit{ into a random variable } X \textit{ over } \{0,1\}^n \textit{ with distribution } \mu\ \}.$$

In [JRS 94] the following precise estimations for the average complexity of some basic Boolean functions have been obtained. Let $\mathtt{AND}_n$ denote the n-ary AND-function, $\mathtt{EQUAL}_n$ the test for equality of 2 binary strings of length n, $\mathtt{ADDITION}_n$ the addition of 2 binary numbers of length n, etc.

$$\begin{gathered}
etime(f, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(2^d, \log n)) \quad \text{for } f = \mathtt{OR}_n,\ \mathtt{AND}_n,\ \mathtt{EQUAL}_n\ , \\
etime(f, \mu_{n,\mathrm{uni}}) = \Theta(\log n) \quad \text{for } f = \mathtt{PARITY}_n,\ \mathtt{MAJORITY}_n\ , \\
etime(\mathtt{ADDITION}_n, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(\log\log n + 2^d, \log n))\ , \\
etime(\mathtt{THRESHOLD}^a_n, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(\log a + 2^d, \log n))\ .
\end{gathered}$$

4 Circuits Do Not Have Malign Distributions

For a Boolean variable x and constant α let x^α equal x if $\alpha = 1$, and $\neg x$ else. $\mathtt{OR}(x_1, \ldots, x_n)$ denotes the n-ary Boolean OR-function over the variables $x_1, \ldots, x_n$. The following result can easily be shown by induction.

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* $\Pr[\mathtt{OR}(x_{i_1}^{\alpha_1}, \ldots x_{i_l}^{\alpha_l}) = 1] \ \geq\ 1 - 2^{-l}$.

Thus already for a small subset of variables one can select values that make the disjunction of these literals quite likely. Using this property we can prove

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

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

Proof: Given μ the function f is defined as follows. Let $l := \log n$ and apply the lemma above to the random subvector $x_1, \ldots, x_l$ of a random variable $X = x_1, \ldots, x_n$ that is distributed according to μ. Choose $g \in B_{n - \log n}$ with $g \notin \mathrm{CirDepth}(n - \log n - \mathrm{llog}\, n - 1)$. Due to the Shannon bound such a g exists (Fact 1). Then $f(x_1, \ldots, x_n) \ := \ \mathtt{OR}(x_1^{\alpha_1}, \ldots, x_l^{\alpha_l}, g(x_{l+1}, \ldots, x_n)\ )$. The $\mathtt{OR}$-subfunction is realized by the average case optimal design presented in [JRS 94] with average delay less than 2. ∎

Hence, the class B_n of all n-argument Boolean functions does not have malign distributions. The proof actually shows that already a small set of at most n functions has the property that for any distribution it possesses at least one example with a maximal difference between the average case and the worst case complexity, that is from constant to linear.

In the next section we will therefore consider an arbitrary Boolean function f and ask how complex a distribution μ has to be in order to make the average complexity of this specific f with respect to μ almost as large as its worst case complexity.

5 Asymptotic Bounds with respect to the Uniform Distribution

We will now study the question how far the Shannon effect remains valid for the average case complexity. For a circuit C and a natural number t define

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

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

Theorem 2 *Almost all functions $f \in B_n$ have average complexity larger than $n - \log n - \log\log n - 1$ w.r.t. the uniform distribution.*

Proof: Let $d := n - \log n - \log\log n - 1$. Assume that with respect to the uniform distribution $f \in B_n$ can be computed by an expected d-time bounded circuit C. We allow that input gates of C may also be set to the negation $\overline{x_i}$ of a variable. This way, internal gates can be restricted to AND and OR gates. Without increasing the delay C can be transformed into an equivalent circuit C' with fanout 1 — simply duplicate gates with larger fanout. Thus C' looks like a binary tree.

We extend the Boolean domain by a new symbol "?". The `OR` and `AND` function is then defined by the following tables:

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

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

Then 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 labelled 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 $2^d(1+\log(2n+1))$.

Comparing C and C'' the definition of *time* implies that on all inputs x that do not belong to $I[C,d]$ C'' yields the same result as C. For $x \in I[C,d]$, however, C'' yields the result "?". Thus the set $I[C,d]$ is uniquely determined by C''. Then 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

$$2^d(1+\log(2n+1)) \ + \ 2^n \frac{d}{d+1} \ \leq \ 2^n\left(1 - \frac{1}{\Theta(n)}\right) \ .$$

But it is well known that for almost all n-ary Boolean functions their Kolmogorov complexity is larger than the last bound. ∎

Because of the worst case depth upper bound $n - \log\log n$ this lower bound is guaranteed to be best possible up to at most 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 3 with respect to the uniform distribution. The proportion of such efficiently computable functions in the average case can be made even larger if we increase the delay bound moderately. 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 Constructing Hard Distributions

Although we have just proved 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 deduce much information 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 had only constant average delay with respect to this distribution. This function depended heavily 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 in the following construction will be devoted 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. Then is not too hard to find a distribution μ_f with complexity $n + O(d)$. 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 6 *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 $2^a \le |A| < 2^{a+1}$. 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 2 *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: For an integer n let $\hat{n}$ be the smallest power of 2 which is at least as large as n. Then, let $\alpha := \widehat{|A|}$. First consider the following circuit C with unbounded fan-out. Its inputs are inputs constants $y_0, \ldots, y_{a-1}$, for which holds $y_0, \ldots, y_{|A|-1}$ are all elements of A and $y_{|A|+i} := y_i$, for $i \ge 0$. Further we need random bits $r_1, \ldots, r_{\log|A|}$. The multiplexer circuit C with inputs r and $y_0, \ldots, y_{\alpha-1}$ outputs string $z = y_{\mathrm{bin}^{-1}(\mathrm{r})}$. This can be achieved by a design using

$$z[k] \quad := \quad \bigvee_{i=0}^{\alpha-1} \left(y_i[k] \wedge \bigwedge_{j=1}^{\log|A|} r_j^{\mathrm{bin(i, \log|A|)[j]}} \right) \quad ,$$

for all $k \le n$, where $s[i]$ for a string s and an integer i denotes the ith bit of s and $\mathrm{bin(i,n)}$ for integer $i \le 2^n - 1$ denotes the binary representation of i of length n. Clearly the depth of circuit C is $\log|A| + \log(\log|A|+1) + 1$.

But note that every random bit r_i is used $|A| \cdot n$ times. To obtain a circuit with constant fan-out we can duplicate these bits using trees. Such a straightforward modification increases the depth by $\log|A| + \log n$.

In this modified circuit the inputs y_i are used in depth $\log|A| + \log n + \operatorname{llog}|A| + 1$ for the first time. Alternatively we can construct a circuit that selects the elements in A iteratively using multiplexers of various depth. Recursively we duplicate random bits of a multiplexer and select elements of A using smaller multiplexers in parallel. It can be shown that this leads to a total depth of $\log|A| + \log(\log|A| + 1) + \log n + 3$. ∎

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 7 *The* $\boldsymbol{(A,k)}$**-uniform distribution** *is given by*

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

Lemma 3 *Let $C \in Cir(f)$ and $t \in \mathbb{N}$. With the help of the set $I[C,t]$ from above 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.*

Let $\chi_{C,t}$ be the characteristic function for the complement of $I[C,t]$. Therefore $\chi_{C,t}$ can be computed in depth $t+3$ using $f_t^{\langle 0\rangle}$ and $f_t^{\langle 1\rangle}$.

Lemma 4 *Let $A := f^{-1}(1)$ and $f \in CirDepth(d)$. 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: Generate k n-bit strings $R_0,\ldots,R_{k-1}$ at random and check for each of them whether they belong to A. If at least one test succeeds output the first such string. Otherwise output 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 constructions generates a (A,k)-uniform distribution.

In parallel all random strings are tested using a circuit for $\chi_{C,t}$ of depth d. It remains to select one of the strings. A straightforward implementation of a multiplexer circuit requires depth $2\log k$. Since we need the parameter k up to size 2^n the factor 2 implies a substantial loss of efficiency.

Define the n-bit strings $r_{\mathrm{bin(i,\log k)}} := R_{\mathrm{bin(i,\log k)}} := R_i$ for all $i \le \log k$, where $\mathrm{bin(i,\log k)}$ denotes the natural binary representation of i of length $\log k$. Now the result $r_\lambda[1],\ldots,r_\lambda[n]$ of the circuit will be defined recursively. For all $j \in \{1,\ldots,n\}$ and a constant $\delta \le \log k$

$$\forall p \in \{0,1\}^{\le \log k - 1} \quad e_p \;:=\; \bigvee_{q \in \{0,1\}^{n-|p|}} \chi_{C,t}(R_{pq})$$

$$\forall i \in [0;\log k/\delta - 1] \quad \forall p \in \{0,1\}^{\delta\cdot i} \quad \forall q \in \{0,1\}^{\delta} \quad s_{pq}[j] \;:=\; r_{pq}[j] \;\wedge\; \neg \bigvee_{u \in \{0,1\}^{\delta} \text{ and } u<q} e_{pu}$$

$$\forall i \ge 0 \quad \forall p \in \{0,1\}^{\delta\cdot i} \quad r_p[j] \;:=\; \bigvee_{q \in \{0,1\}^{\delta}} s_{pq}[j]$$

The nodes corresponding to $e_{\mathrm{bin(i)}}$ have depth $d + \log n$. For e_p there is a depth of $d + \log n + \log k - |p|$. A true value of e_p only occurs if a test $\chi_{C,t}(R_w)$ succeeds, such that p is a prefix of

w. A true value of the term $\bigvee_{u\in\{0,1\}^\delta \text{ and } u<q} e_{pu}$ indicates at least one successful test $\chi_{C,t}(R_w)$ for $w \in [p0^{n-|p|}, pq1^{n-|pq|}]$. If pq is the left most succesfull test, then s_{pq} is r_{pq}, otherwise zero. So r_p gives the left most successful string in $R_{p0^{n-|p|}}, \ldots R_{p1^{n-|p|}}$, if it exists.

We divide the circuit into $\lceil \log k/\delta \rceil + 2$ levels. For $p \in \{0,1\}^{i\cdot\delta}, q, r \in \{0,1\}^\delta$. In each level with $i \in \{1, \ldots, \lceil \log k/\delta \rceil\}$ there is an OR-tree of depth δ and a preceeding AND-gate that produces r_p. For $i \in \{2, \ldots, \lceil \log k/\delta \rceil + 1\}$ an OR-tree of depth δ with a negation following computes s_{pq}. For $i \in \{3, \ldots, \lceil \log k/\delta \rceil + 2\}$ we use a fan-out tree starting with inputs e_{pqr} of depth δ.

Note that the computation of all parts can be done independently. Therefore we get a total depth

$$d + \log n + (\delta+1)(\lceil \frac{\log k}{\delta} \rceil + 1) + \delta \;=\; d + \log k + \log n + (\delta+1)\lceil \frac{\log k}{\delta} \rceil + 2\cdot\delta + 1 \;.$$

If we choose $\delta := \lceil \sqrt{\log k/2} \rceil$ the depth is bounded by $d + \log k + \lceil \sqrt{8 \log k} \rceil + \log n + 1$. ∎

These almost uniform distributions share the following property.

Lemma 5 *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 $\mathrm{E}_\mu(h) \leq t$. Then for the set $\boldsymbol{A[h,t]} := \{x \in A \mid h(y) \leq t\}$ holds $|A[h,t]| \geq |A| \cdot (2\cdot(t+1))^{-1}$.*

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

$$\mathit{CirETime}\left(t, \mathcal{D}\mathit{Depth}\left(\frac{(t+n)}{2} + \sqrt{n-t} + \log n + 4\right)\right) \;\subseteq\; \mathit{CirDepth}(t + \log n + \log t + 3) \;.$$

Proof: Let $\tau := \frac{1}{2}(t+n) + \sqrt{n-t} + \log(n+t) + 3$ and $f \in \mathrm{CirETime}(T, \mathcal{D}\mathrm{Depth}(\tau))$. 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 $\boldsymbol{X_i} := \overline{I[C_i,t]} = \{x | \mathit{time}_{C_i} \leq t\}$ completely cover the input set $\{0,1\}^n$.
2. Using lemma 3 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 $t + \log p$.

To get 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 $\mathrm{E}_{\mu_1}(\mathit{time}_{C_1}) \leq t$. Define $X_1 := \{x \mid \mathit{time}_{C_1}(x) \leq t\}$ and $Z_2 := Z_1 \setminus X_1$.

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

Applying lemma 2 and lemma 4 the distributions μ_m can be generated in depth τ. Using lemma 5 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. ∎

Corollary 1 *For all Boolean functions $f \in \mathit{CirDepth}(d)$ 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.*

7 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.

For simple probability distributions there are functions like `OR`, `AND`, `THRESHOLD` and `ADDITION` [JRS 94] with substantially smaller average case complexity. On the other hand for other functions like `PARITY` it has been shown in [JRS 94, JRSW 94] that average case and worst case complexity are asymptotically identical for any distribution.

In this paper we have shown that most functions are hard in the average case even for the uniform distributions, thus the Shannon effect also holds 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 worst case behavior is substantially worse than the average case behavior 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 [JRS 94] and Theorem 3 it follows that it is between llog n and $n + O(\log n)$. Narrowing this gap would yield a better understanding of average case complexity for a broader class of functions and distributions.

References

[BCGL92] S. Ben-David, B. Chor, O. Goldreich, M. Luby, *On the Theory of Average Case Complexity*, J. CSS 44, 1992, 193-219.

[DGY89] I. David, R. Ginosar, M. Yoelli, *An Efficient Implementation of Boolean Functions and Finite State Machines as Self-Timed Circuits*, ACM SIGARCH, 1989, 91-104.

[Gask78] Gaskov, *The Depth of Boolean Functions,* Prob. Kybernet. 34, 1978, 265-268.

[Grap90] P. Grape, *Complete Problems with L-sampleable Distributions,* Proc. 2. SWAT 90, 360-367.

[Gure91] Y. Gurevich *Average Case Completeness*, J. CSS 42, 1991, 346-398.

[JRS 94] Andreas Jakoby, R"udiger Reischuk, Christian Schindelhauer, *Circuit Complexity: From the Worst Case to the Average Case*, Proc. 26. SToC, 1994, 58-67.

[JRSW 94] Andreas Jakoby, R"udiger Reischuk, Christian Schindelhauer, Stephan Weis, *The Average Case Complexity of the Parallel Prefix Problem* 21. ICALP, 1994, 593-604.

[Koba93] K. Kobayashi, *On Malign Input Distributions for Algorithms,* IEICE Trans. Inf. & Syst. 76, 1993, 634-640.

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

[Levi86] L. Levin, *Average Case Complete Problems*, SIAM J. Computing 15, 1986, 285-286.

[LiVi89] M. Li, P. Vitanyi, *Inductive Reasoning and Kolmogorov Complexity,* Proc 4. Structure, 1989, 165-185.

[LiVi92] M. Li, P. Vitanyi, *Average Case Complexity under the Universal Distribution Equals Worst-Case Complexity,* IPL 42, 1992, 145-149.

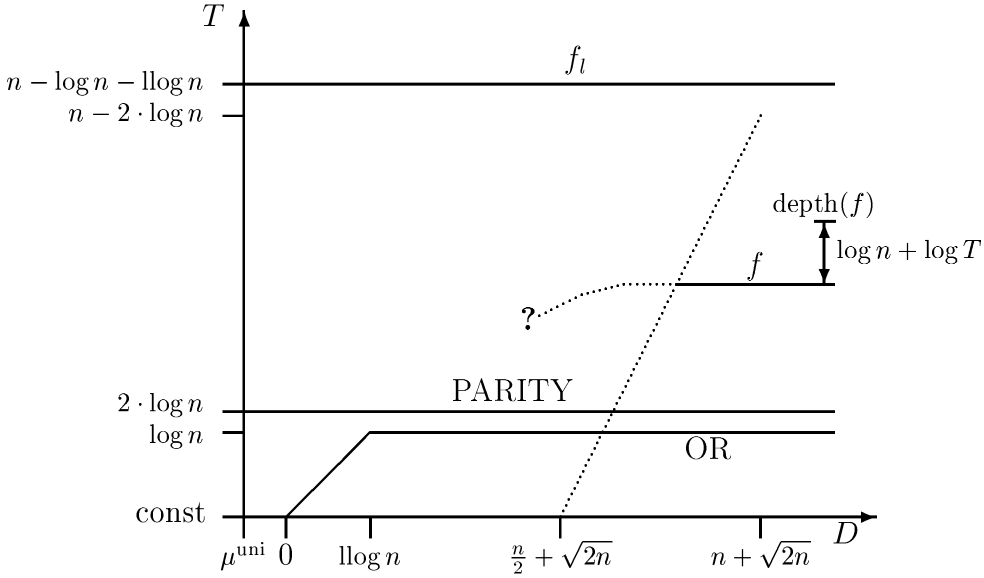

Figure 1: Lower bounds for the average complexity of OR, PARITY and an arbitrary function f with respect to the complexity of the distributions. F denotes a functions of maximal asymptotic complexity as described in theorem 1.

[Milt91] P. Miltersen, *The Complexity of Malign Ensembles,* Proc. 6. Structure in Complexity Theory, 1991, 164-171, see also SIAM. J. Comput. 22, 1993, 147-156.

[ReSc93] R. Reischuk, C. Schindelhauer, *Precise Average Case Complexity,* Proc. 10. GI-AFCET Symposium on Theoretical Aspects of Computer Science, STACS 1993, Springer Lecture Notes, 650-661, see also Technical Report 93-049 ICSI Berkeley, 1993.

[Weg87] Ingo Wegener, *The Complexity of Boolean Functions*, Wiley–Teubner, 1987.

[WaBe92] Jie Wang, Jay Belanger, *On Average P versus Average NP*, Proc. of 7. IEEE Conference on Structure in Complexity Theory, 1992, 318-326.

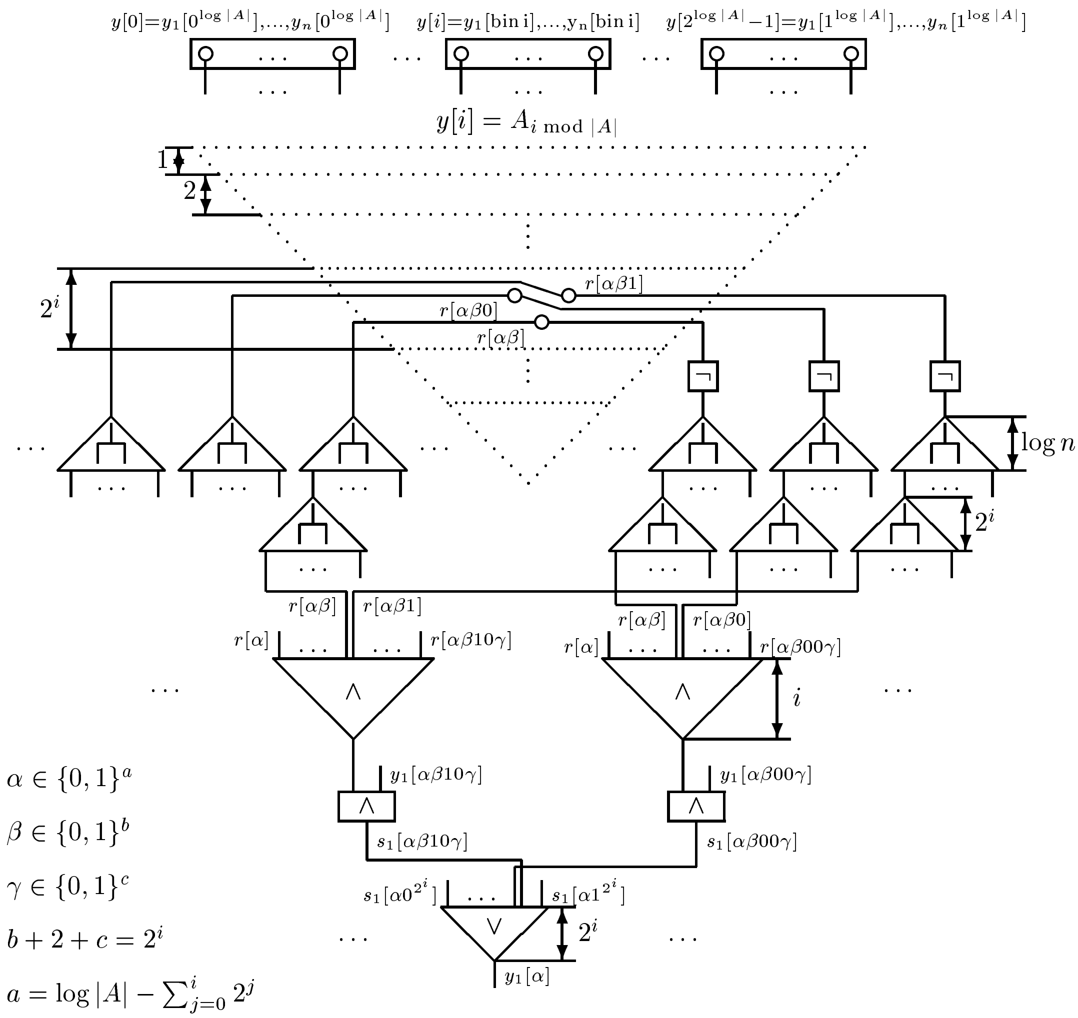

Figure 2: A circuit generating a nearly A-uniform distribution

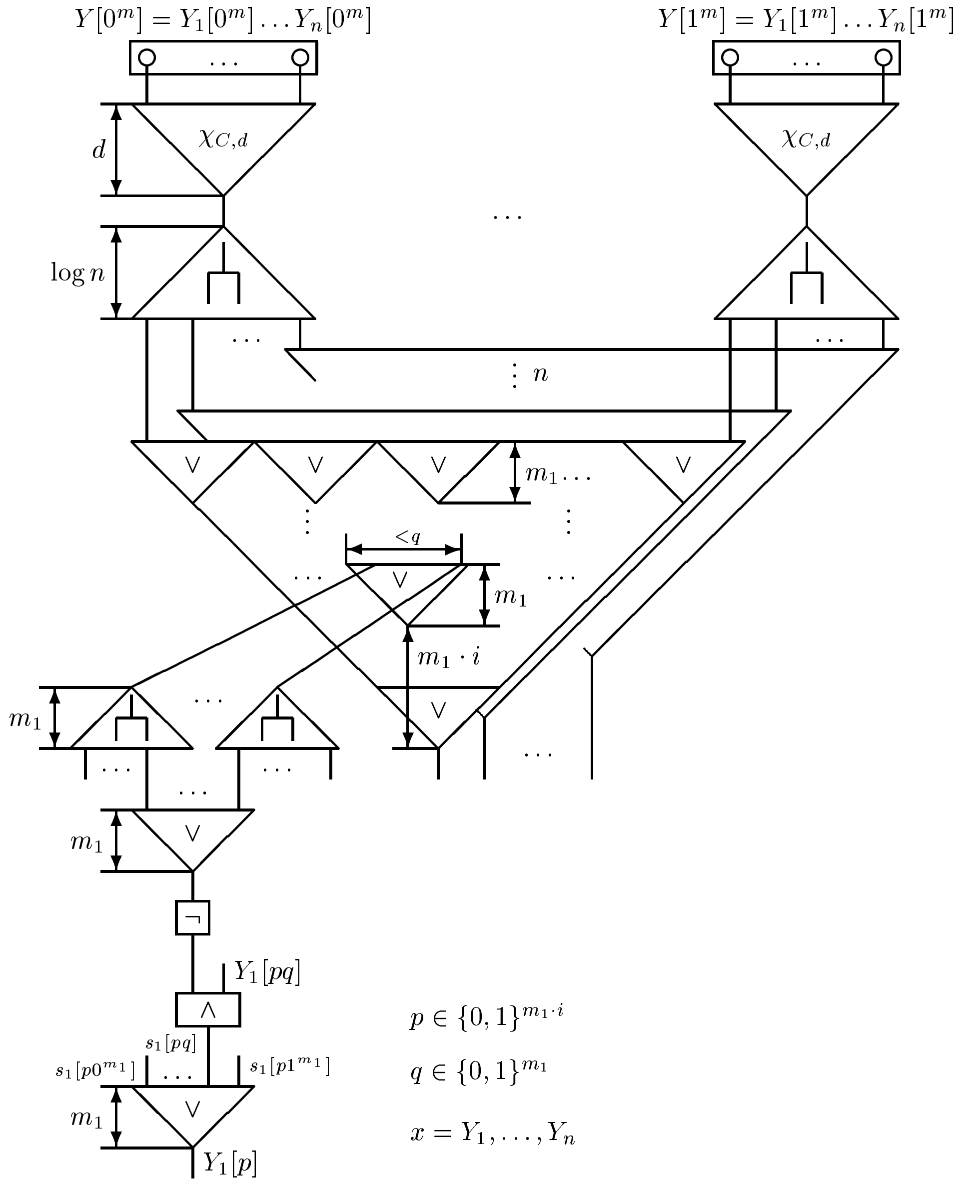

Figure 3: A circuit generating a (A, k)-uniform distribution

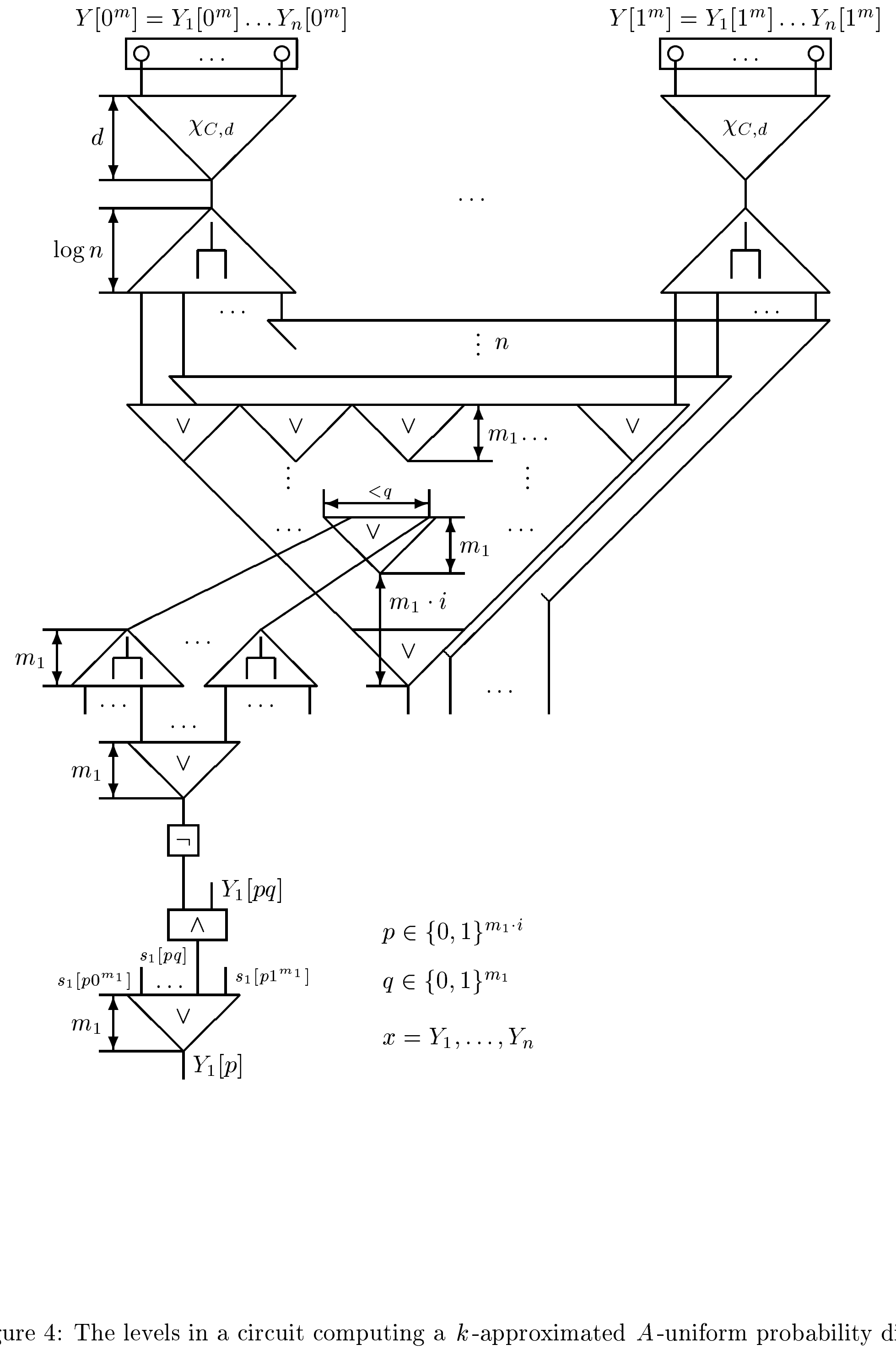

Figure 4: The levels in a circuit computing a k-approximated A-uniform probability distribution