On the Complexity of Worst Case and Expected Time in a Circuit

Andreas Jakoby

Christian Schindelhauer*

Medizinische Universität zu Lübeck

Abstract. The computational delay of a circuit can be described by the natural concept of time [Jakoby et al. STOC94]. We show that for a given input x and circuit C the computation of $\text{time}_C(x)$ is $\mathcal{P}$-complete. Moreover, we show that it is $\mathcal{NP}$-complete to decide whether there exists an input x such that $\text{time}_C(x) \leq t$ for a given time bound t.

We introduce the notion of worst time of a circuit and show that to decide whether a given time bound is the worst time of a circuit is $\mathcal{BH}_2$-complete. We also prove that the computation of an arbitrary worst case input is $\mathcal{FP}_{\text{tt}}^{\mathcal{NP}}$-hard, whereas the search of the lexicographically minimal worst case input is $\mathcal{FP}^{\mathcal{NP}}$-complete and of the lex. middle worst case input is $\mathcal{FP}^{\#\mathcal{P}}$-complete.

Computation of the expected time $E_{\mu_D}(\text{time}_C)$ of a circuit C with respect to a distribution μ_D generated by circuit D is $\#\mathcal{P}$-complete under metric reducibility. Nevertheless we show that a polynomial time bounded probabilistic Turing machine approximates $E_{\mu_D}(\text{time}_C)$ up to an arbitrary additive constant with high probability.

Key words: theory of parallel and distributed computation, computational complexity, average case analysis, expected time, worst case, timed circuits.

1 Introduction

Most research in circuit theory concentrates on the complexity measures *size* and *depth*. In [JRS93] a natural concept for a measure *time* is presented. Contrary to depth the time measure takes into account that gates may be evaluated before all inputs are known.

Two models for timing in a circuit are investigated in [JRS93]: The *explicit time* model works with a 3-valued logic where in addition to Boolean values 0 and 1 the sign **?** indicates an undefined value. This model is related to the notion of self-timed circuits [DGY89]. In the beginning all gates are initialized

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

with ? and all inputs with Boolean values. E.g. in an OR the question mark will be replaced (which means computation continues) when an input has value 1 or at the latest when both inputs are given.

In this paper the *implicit* definition of time will be used. It is proved in [JRS93] that both definitions are equivalent. Furthermore, it is shown that up to a constant factor, the time is independent of the choice of a complete base.

In Computer systems Boolean circuits are used as elementary computation units. For a combination of circuits in synchronous computer models it is essential to know the maximum time the circuits need for computation. We call this time the *worst time*. Of course the depth of a circuit gives an upper bound for the worst time. But Krapchenko has shown the existence of a function whose circuits with minimal size are deeper than the worst time (see [Krap78]). In [LBS93] an approach for computing the delay is presented for several delay models. The authors examine the worst time and offer an algorithmic solution. In section 2 we will present upper and lower bounds for these and related problems. In particular, we show that for a given circuit C and a bound t the question $\exists x \; \mathrm{time}_C(x) \leq t$ is $\mathcal{NP}$-complete. This also holds for logarithmic depth bounded circuits.

The time of a circuit depends on the input. This way it is possible to investigate the average behavior of circuits. In [Reif93] a circuit for addition of n-bit numbers ($\mathtt{ADD}_n$) is presented which is expected $\log\log n$ time bounded with respect to the uniform distribution.

Broader classes of distributions are investigated in [JRS93]. Here special depth-bounded *distribution-generating* circuits with coin-tossed inputs are used to define a complexity class of distributions. Basic functions like OR, PARITY, ADD, THRESHOLD and others are asymptotically exactly classified. For many functions a tremendous speedup is possible, even with respect to classes of distributions generated by DG-circuits. In this model for a given distribution the circuit can be chosen appropriately.

On the other hand for functions like OR (see [BHPS94]), ADD and some other parallel prefix problems (see [JRSW93]) there exist *robust* circuits which have asymptotically optimal expected time behavior with respect to broad sets of probability distributions. In section 4 we will examine the complexity of computing the expected time of a given circuit.

For a complexity class a probability distribution is called *malign* [Milt93, LiVi92], if each of its problems cannot provide a substantial speedup in the expected time measure with respect to this distribution. In [JRS95] it is shown that circuit complexity classes do not have malign probability distributions, as long as they contain a small set of basic functions. On the other hand it is proved that for every Boolean function exist malign distributions.

For a circuit the *worst distribution* only weights worst case inputs. Here we show, that its computation is as hard as to compute a worst case input for C, which we will classify as $\mathcal{FP}_{\mathtt{tt}}^{\mathcal{NP}}$-hard.

2 Time in a Circuit

Depth and size are the most often investigated complexity measures of Boolean circuits. Here depth is usually taken as a measure for the computational delay. Depth as well as the parallel time considered so far are worst case measures.

Definition 1. Let $\mathcal{B}_n$ denote the set of Boolean functions $f : \{0,1\}^n \to \{0,1\}$. 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.

An important parameter of a circuit is the amount of time needed for evaluation. 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 $\mathbf{time} : \{0,1\}^n \to \mathbb{N}$ for each gate v of a circuit C. It specifies for each input x the step when v can compute its result $\mathbf{res_v(x)}$ using the values of its predecessors. The result of the circuit's output is called $\mathbf{res_C(x)}$.

This leads to the definition of **implicit** time measure of a circuit [JRS93].

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

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

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

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

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

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

Note that for a given circuit its depth is computable in linear time and in space $\log \mathrm{size}(C) \cdot \log \mathrm{depth}(C)$. Furthermore, it can also be determined in $\mathcal{NC}$ which means it can efficiently be parallelized. As an upper bound for the computation of the time we can show:

Lemma 3. *The time of a circuit C on input x $\mathrm{time}_C(x)$ can be computed in linear time and space; i.e.* $\mathrm{O}(\mathrm{size}(C))$.

For a lower bound we define the **circuit time problem** (CTP): For a given Boolean circuit C, an input string x and a bound t decide whether $\mathrm{time}_C(x) \le t$.

Theorem 4. CTP *is $\mathcal{P}$-complete.*

Proof. From lemma 3 it follows that the CTP is in $\mathcal{P}$.

The $\mathcal{P}$–hardness follows from a reduction of the ***Circuit Value Problem (CVP)*** defined as: For a given Boolean circuit C and an input string x decide whether C on input x outputs 1, i.e. compute $\mathrm{res}_C(x)$. It is well known that the CVP problem is $\mathcal{P}$-complete (see [GHR91]).

Define the circuit $\mathtt{CHAIN}_l(b)$ as a circuit with depth and size l that consists of l successing OR-gates with input b. Let C and x be an instance of the CVP problem. Now define circuit $C' := C\ \mathtt{OR}\ \mathtt{CHAIN}_{\mathrm{size}(C)+1}(0)$. Note that

$$\mathrm{time}_{C'}(x) := \begin{cases} 1 + \mathrm{time}_C(x) \text{ if } \mathrm{res}_C(x) = 1 \\ 2 + \mathrm{size}(C) \quad \text{else .} \end{cases}$$

Let $t := 1+\mathrm{size}(C)$. So to decide whether $\mathrm{time}_{C'}(x) \leq t$ is as hard as to compute $\mathrm{res}_C(x) = 1$. □

In a synchronous computer any partial circuit with evaluation time greater than the clock would cause fatal errors. But the question whether $\exists x\ \mathrm{time}_C(x) > t$ or the related question $\exists x\ \mathrm{time}_C(x) \leq t$ leads to $\mathcal{NP}$-completeness.

Theorem 5. *For a given circuit C and a bound t the decision whether there exists an input x such that* $\mathrm{time}_C(x)$ *rel* t *is $\mathcal{NP}$-complete for* rel $\in \{\leq, =, \geq\}$.

This can be proved by straightforward reduction of the satisfiability problem of Boolean functions (SAT). In the same way the circuits can be restricted such that the range of time_C is $\{t_1, t_2\}$ with $t_2 = 2^{t_1}$ which prevents polynomial time approximation within any constant factor, unless $\mathcal{P} = \mathcal{NP}$.

3 Worst Case in a Circuit

The depth of a circuit must not be mixed up with the worst time, since in [Krap78] it is shown, that there exists a circuit, the depth of which is greater than the worst time even if the circuits have minimal size. That means that depth of a circuit and worst case time of parallel algorithms may not correlate.

Definition 6. We call $t_{\mathtt{wc}}(C) := \max_x \mathrm{time}_C(x)$ the ***worst time*** of a circuit C and an input x a ***worst case*** of C if $\mathrm{time}_C(x) = t_{\mathtt{wc}}(C)$.

Since it is $\mathcal{NP}$-hard to decide whether there exists an input x with $\mathrm{time}_C(x) \geq t$, it is co-$\mathcal{NP}$-hard to decide whether t is greater than the worst time.

Theorem 7. *To decide for a given circuit C and a bound t whether t is the worst time of C is complete for $\mathcal{BH}_2$.*

Proof. Note that this problem is equivalent to the following predicate.

$$\exists x\ :\ \mathrm{time}_C(x)\ =\ t \qquad \text{and} \qquad \forall x\ :\ \mathrm{time}_C(x)\ <\ t+1\ .$$

So this problem belongs to the second stage of the Boolean hierarchy.

The completeness follows by a reduction to SAT-UNSAT [PaYa82] defined as: Given two Boolean formulas F, F'. Decide whether F is satisfiable, whereas F' is not.

For a given instance F, F' of the SAT-UNSAT-problem generate a circuit C that computes F and C' for F'. Let $\tilde{C} := C \,\mathtt{AND}\, \mathtt{CHAIN}_{\mathrm{size}(C)+\mathrm{size}(C')}(0)$ and $\tilde{C}' := C' \,\mathtt{AND}\, \mathtt{CHAIN}_{2\,\cdot\,(\mathrm{size}(C)+\mathrm{size}(C'))}(1)$. Futhermore let $C_{\mathrm{final}} := \tilde{C} \,\mathtt{OR}\, \tilde{C}'$. It is easy to see that $\mathrm{size}(C) + \mathrm{size}(C') + 2$ is the worst time of C_{final} if and only if F is satisfiable whereas F' is not. □

Definition 8. For a circuit C let $\mathcal{WC}_C$ be the set of all worst case inputs for C. Further $\mathtt{WC}_C(k)$ denotes the lexicographically k-th worst case input.

Circuit Worst Case Problem (CWC):
The relation CWC is defined as $\mathtt{CWC} := \{(C, x) \mid \mathrm{time}_C(x) = t_{\mathtt{wc}}(C)\}$. An algorithm solves the CWC-problem if for every given circuit C it computes a binary string x with $(C, x) \in \mathtt{CWC}$.

For $k \in \mathbb{N}$: ***Lex. k-th Circuit Worst Case Problem*** (lex-kth-CWC):
For a given circuit C compute the lexicographically k-th worst case input, or if it does not exist, the largest one, i.e. compute $\mathtt{WC}_C(\min(|\mathcal{WC}_C|, k))$.

Lex. Middle Circuit Worst Case Problem (lex-middle-CWC):
For a given circuit C compute the lexicographically middle worst case input, i.e. compute $\mathtt{WC}_C(\lceil |\mathcal{WC}_C|/2 \rceil)$.

Lex. Indexed Circuit Worst Case Problem (lex-index-CWC):
For a given circuit C and a natural number m compute the lexicographically mth worst case input, or if it does not exist, the largest one, i.e. compute $\mathtt{WC}_C(\min(|\mathcal{WC}_C|, m))$.

The construction of a worst case input seems to require linear many questions to an $\mathcal{NP}$-oracle.

Lemma 9. $\mathtt{lex\text{-}1st\text{-}CWC} \in \mathcal{FP}^{\mathcal{NP}}$.

Proof. Note that the worst time t of an arbitrary circuit C can be determined in polynomial time using an $\mathcal{NP}$-oracle via binary search.

Let $\mathcal{O}$ be a polynomial time bounded nondeterministic Turing machine that verifies for a given circuit C and a number t whether there exists an input x of C such that $\mathrm{time}_C(x) = t$. Using $\mathcal{O}$ as an oracle, a TM M can compute a worst case input of length n for C by fixing the input variables of C in the oracle questions sequentially. Clearly, M is in $\mathcal{FP}^{\mathcal{NP}}$. □

Theorem 10. lex-1st-CWC *is* $\mathcal{FP}^{\mathcal{NP}}$*-complete.*

Proof. From lemma 9 it follows that the problem to generate the lexicographically maximum worst case input is in $\mathcal{FP}^{\mathcal{NP}}$.

The $\mathcal{FP}^{\mathcal{NP}}$–hardness follows from a reduction of the MAXSAT problem (i. e. to find the lex. maximum satisfying assignment), which is known to be $\mathcal{FP}^{\mathcal{NP}}$–complete (see [Kren86]). For a Boolean function F generate a circuit C' using the strategy of theorem 5. Note that the lex. maximum worst case input of C' gives also the lex. maximum satisfying assignment of F. □

The more general problem to find the lex. indexed worst case of circuit seems to be more difficult.

Theorem 11. `lex-index-CWC` *and* `lex-middle-CWC` *are* $\mathcal{FP}^{\#\mathcal{P}}$*–complete under* $\leq^{\mathcal{FP}}_{1\text{-}T}$ *reducibility.*

This follows by a $\leq^{\mathcal{FP}}_{1\text{-}T}$-reduction to `lex k-th SAT` (see [Toda90]).

Theorem 12. *It is* $\mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$*–hard to compute a worst case input of a given circuit.*

Proof. In the following we will consider nondeterministic Turing machines with canonical computation trees. Therefore we order the nondeterministic choices at each state of these machines. Furthermore we restrict these machines to binary choices. So we get a left (resp. 0) and a right (resp. 1) choice for every state.

Let $f \in \mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$ and D be a polynomial time bounded Turing machine that computes $f(y)$ on input y using polynomial many queries to an $\mathcal{NP}$-oracle $\mathcal{O}$ where all must be asked before any of them is answered. Let $q(y)$ denote the number of oracle queries of D.

Note that we can construct a NTM N that works in three phases: In the first phase N chooses nondeterministically the answer string $w_{\mathtt{oracle}} \in \{0,1\}^{q(y)}$ of $\mathcal{O}$. In the second phase it verifies the positive answers ($w_{\mathtt{oracle}}[i] = 1$) and simulates D where the oracle answers are given by the corresponding binary digits of $w_{\mathtt{oracle}}$. For $w \in \{0,1\}^{q(y)}$ let value(w) be the result of D with answer string w. Finally N chooses a binary string of length d where d denotes the length of the phases 1 and 2. For a computation p of N let $w_{\mathtt{comp}}(p) \in \{0,1\}^{2d}$ describe the nondeterministic choises of N on p. Let p be an accepting computation of N iff the substring $w_{\mathtt{oracle}}(p) := w_{\mathtt{comp}}(p)[0]..w_{\mathtt{comp}}(p)[q(y)-1]$ describes the answer string of $\mathcal{O}$ where all positive answers are chosen correctly and it holds $w_{\mathtt{comp}}(p)[d]..w_{\mathtt{comp}}(p)[2d-1] = \text{value}(w_{\mathtt{oracle}}(p))$ if leading zeros are ignored.

For a path p let $\mathcal{T}(p)$ be the set of positive oracle answers of $w_{\mathtt{oracle}}(p)$. Since each query to the oracle must be asked before any of them is answered there are accepting computations p of N such that for each accepting computation p' it holds $\mathcal{T}(p') \subseteq \mathcal{T}(p)$. Note that for such a computation p it holds $\text{value}(w_{\mathtt{oracle}}(p)) = D^{\mathcal{O}}(y)$.

Since N is a polynomial time bounded NTM we can construct a circuit C with input gates $x_0, \ldots, x_{2d-1}$ where x_i describes the i-th nondeterministic choice of N (see [Cook71] and [Pa94]). So, w describes an accepting path iff $\text{res}_r(w) = 1$.

Note that there exists circuit Δ with input gates $x_0, \ldots, x_{q(y)-1}$ such that $\text{time}_\Delta(x_0, \ldots, x_{q(y)-1}) := \text{depth}(C) + |\{x_i = 1 \mid i \in \{0, \ldots, q(y)-1\}\}|$. Δ can by generated in polynomial time.

Finally let $\tilde{C}(x_0, \ldots, x_{2d-1}) := C(x_0, \ldots, x_{2d-1})\ \mathtt{AND}\ \Delta(x_0, \ldots, x_{q(y)-1})$. It can be shown that $\text{time}_{\tilde{C}}(x_0, \ldots, x_{2d-1}) := 1 + \text{time}_\Delta(x_0, \ldots, x_{q(y)-1})$ if the result of C on input $x_0, \ldots, x_{2d-1}$ is 1 and $\text{time}_{\tilde{C}}(x_0, \ldots, x_{2d-1}) \leq 1 + \text{depth}(C)$ otherwise. So, it holds that $\text{time}_{\tilde{C}}(x_0, \ldots, x_{2d-1})$ is the worst time of $\tilde{C}$ if $\text{res}_C(x_0, \ldots, x_{2d-1}) = 1$ and $|\{x_i = 1 \mid i \in \{0, \ldots, q(y)-1\}\}|$ is maximal. From the construction of N and C follows that for maximum $\text{time}_{\tilde{C}}(x_0, \ldots, x_{2d-1})$ the Boolean values of $x_d, \ldots, x_{2d-1}$ give the result of $D^{\mathcal{O}}(y)$. □

After we have discussed the complexity of worst case inputs, we start to investigate sets of inputs or more generally probability distributions.

Definition 13. The set of all probability distributions μ on $\{0,1\}^n$ is denoted by $\boldsymbol{\mathcal{D}_n}$. The uniform distribution on $\{0,1\}^n$ that gives equal probability to each of the 2^n possible input vectors is denoted by μ_n^{uni}. A ***worst case probability distribution*** μ of circuit C gives positive weight only to worst case inputs of C.

The worst probability distribution for the expected time of a circuit is without any doubt a worst case distribution.

Now we want to characterize the complexity of worst case distributions. Hence we have to find an encoding for probability distributions. In [JRS93] for the average case analysis of circuits a complexity measure is defined for distributions that generate the inputs. The complexity is 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 [BCGL92].

Definition 14. 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 to 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. For a DG-circuit D the distribution of the transformed random variable is denoted by $\boldsymbol{\mu_D}$.

Using this encoding we can characterize the complexity to decide whether a given DG-circuit generates a worst case distribution.

Theorem 15. *For a given circuit C and a DG-circuit D it is* co-$\mathcal{NP}$*-complete to decide whether μ_D is a worst case distribution of C.*

Proof. Given a circuit $C \in \text{Cir}(\mathcal{B}_n)$ and a DG-circuit $D \in \text{Cir}(\mathcal{B}_r^n)$ the decision whether μ_D is a worst case distribution of C is equivalent to

$$\begin{aligned} &\forall z \in \{0,1\}^r \ : \ \text{time}_C(\text{res}_D(z)) \ = \ \text{time}_C(\text{res}_D(0^r)) \\ \text{and} \quad &\forall x \in \{0,1\}^n \ : \ \text{time}_C(x) \ \leq \ \text{time}_C(\text{res}_D(0^r)) \ . \end{aligned}$$

So we can solve this problem in co-$\mathcal{NP}$.

On the other hand we can reduce the `TAUTOLOGY` problem to this problem using the construction in the proof of theorem 5. So even the decision whether μ^{uni} is a worst case distribution of C, is co-$\mathcal{NP}$-complete. □

Note that a worst case distribution for a circuit C can constantly output one worst case input for C. Such a distribution can be achieved by the algorithm of lemma 9. On the other hand if we have a strategy to compute a DG-circuit $D \in \text{Cir}(\mathcal{B}_r^n)$, which generates a worst case distribution for $C \in \text{Cir}(\mathcal{B}_n)$, we can easily use it to determine the worst case input $D(0^r)$.

Corollary 16. *For a given C the computation of a DG-circuit, which generates a worst case distribution for C, is in $\mathcal{FP}^{\mathcal{NP}}$ and $\mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$-hard.*

4 Expected Time of a Circuit

Definition 17. 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 S is a set of probability distributions we define

$$\boldsymbol{etime(f,S)} := \max_{\mu \in S} \min_{C \in Cir(f)} \mathrm{E}_\mu(\mathrm{time}_C)$$

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

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 [JRS93] it is shown that

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

If DG-circuits are restricted to constant fan-out 2 for the set $\mathcal{D}\mathrm{Depth}(d)$ of all probability distributions with d-depth bounded DG-circuits it holds [JRS93]:

$$etime(\mathtt{OR}_n, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(2^d, \log n)) ,$$
$$etime(\mathtt{THRESHOLD}_a^n, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(\log a + 2^d, \log n)) ,$$
$$etime(\mathtt{ADD}_n, \mathcal{D}\mathrm{Depth}(d)) = \Theta(\min(\log\log n + 2^d, \log n)) .$$

DG-circuits simulate the context of the circuits. The question is whether an average analysis of the time of circuit C with respect to μ_D generated by circuit D can be reduced to the analysis of the combined circuit $C \circ D$. This is not the case, since $\mathrm{time}_{C\circ D}(x) = \mathrm{time}_C(\mathrm{res}_D(x)) + \mathrm{time}_D(x)$. So the time behavior of D influences the behavior of $C \circ D$.

Lemma 18. *For a given circuit $D \in Cir(\mathcal{B}_r^n)$, there exists a circuit D' with $\forall x\ \mathrm{res}_{D'}(x) = \mathrm{res}_D(x)$ and $\mathrm{size}(D') \le \mathrm{size}(D) + \mathrm{depth}(D) + 7n$ and $\mathrm{time}_{D'}(x) = \mathrm{depth}(D) + 3$.*

Proof. Let $D' := \mathtt{EQUAL}(D, \mathtt{CHAIN}_{\mathrm{depth}(D)}(1))$, where EQUAL is implemented as $\mathtt{EQUAL}(x,y) := ((x\ \mathtt{AND}\ x)\ \mathtt{AND}\ (y\ \mathtt{AND}\ y))\ \mathtt{OR}\ ((\mathtt{NOT}\ x)\ \mathtt{AND}\ (\mathtt{NOT}\ y))$. □

Now if we combine C with the synchronized circuit D' of D, we only have to analyze $E_{\mu_D}(time_C) + \mathrm{depth}(D')$.

Theorem 19. *For a given circuit $C \in Cir(\mathcal{B}_n)$ and a DG-Circuit D, there exists a circuit C' with $\forall x\ \mathrm{res}_{C'}(x) = \mathrm{res}_C(\mathrm{res}_D(x))$ and $\mathrm{size}(C') \le \mathrm{size}(C) + \mathrm{size}(D) + \mathrm{depth}(D) + 7n$ and $\mathrm{E}_{\mu^{\mathrm{uni}}}(\mathrm{time}_{C'}) = \mathrm{E}_{\mu_D}(\mathrm{time}_C) + \mathrm{depth}(D) + 3$.*

If we encode probability distributions by DG-circuits using r random bits every probability is a multiple of 2^{-r}. So the number $E_{\mu_D}(\text{time})$ can easily transformed into a natural number by multiplication with 2^r.

Definition 20. ***Circuit Expected Time Problem (*** $\texttt{CET}$***)***
For a given circuit C and a given DG-circuit D with r random input bits, that defines the probability distribution μ_D, compute $\mathrm{E}_{\mu_D}(\text{time}_C) \cdot 2^r$.

Lemma 21. $\texttt{CET} \in \#\mathcal{P}$.

Proof. Consider $\texttt{CET}$ restricted to uniform distributions. A nondeterministic Turing machine N chooses a bit string $x \in \{0,1\}^r$ and computes $t := \text{time}_C(x)$. N is constructed such that at this configuration t accepting leaves are following. Therefore the number of accepting paths is $\sum_{x\in\{0,1\}^r} \text{time}_C(x) = 2^r \cdot \mathrm{E}_{\mu^{\text{uni}}}(\text{time}_C)$. By the same technique we get $\texttt{CET} \in \#\mathcal{P}$. □

On the other side, $\texttt{CET}$ is $\#\mathcal{P}$-hard under metric reducibility. To get a problem which is $\#\mathcal{P}$-hard even under many-one reducibility we enhance the linear mapping.

Theorem 22. *For a given circuit $C \in Cir(\mathcal{B}_r)$ and $k \in \mathbb{N}$, the computation of $2^r \cdot \mathrm{E}_{\mu^{\text{uni}}}(\text{time}_C) - 2^k$ is $\#\mathcal{P}$-hard.*

Proof. Let M be a polynomial time bounded NTM. At first we transform this machine into a polynomial time bounded machine M' with the same number of accepting states and a complete binary computation tree of depth $p(n) \in \mathrm{POL}$.

Now let C_i be the polynomial time bounded circuit, that realizes the i-th computation step of M' as follows. It has $i-1$ input bits $g_1,\ldots g_{i-1}$ which represents the choices M' has made in previous computations steps. We add an input bit g_i that indicates which choice M' is doing in the i-th step. In addition the configuration of M' at the $(i-1)$-th step is given in the input. Note that the input length is bounded by $\mathrm{O}(p(n))$.

The circuit now simulates one step of M' and outputs the succeding configuration of M'. This circuit has constant depth c and size $\mathrm{O}(p(n))$.

Now let $k = 1 + \log(c \cdot p(n) + 4)$ and $r = p(n)$. We construct the circuit C as follows. $g_1,\ldots g_r$ are the inputs of C. Every circuit C_{i+1} is fed with the configuration output of C_i and the inputs $g_1,\ldots,g_{i+1}$. The result of C_r is a bit indicating whether M' accepts or not.

Now $\texttt{CHAIN}_l(b)$ and $\texttt{EQUAL}$ are defined as in theorem 4 and lemma 18. The wanted circuit C is defined for input gates $x = x_0,\ldots,x_n$ as

$$C \;:=\; (C_p(C_{p-1}(\ldots C_0(x)\ldots))\ \texttt{EQUAL}\ \texttt{CHAIN}_{2^k-4}(1))\ \texttt{AND}\ \texttt{CHAIN}_{2^k}(0)\ .$$

For the time of C holds:

$$\text{time}_C(g_1,\ldots,g_r) \;=\; \begin{cases} 2^k+1\ , \text{if } M' \text{ accepts } x \text{ with choices } g_1,\ldots,g_{p(n)} & , \\ 2^k\ , \quad \text{if } M' \text{ rejects } x \text{ with choices } g_1,\ldots,g_{p(n)} & . \end{cases}$$

Therefore for the expected time of C and for $a(x)$ defined as the number of accepting configurations of M (or M') on input x, it holds

$$\mathrm{E}_{\mu^{\text{uni}}}(\text{time}_{C_x}) \;=\; (a(x) + 2^k) \cdot 2^{-r}\ .$$

□

For the exact computation of the expected time of a given circuit no efficient deterministic algorithm is in sight. So we try to approximate this problem with a probabilistic Turing machine.

We already showed a polynomial time bounded algorithm to compute the time. Furthermore we model the probability distributions of inputs by DG-circuits, that can be simulated in polynomial time, too. The idea is now to make several experiments on random inputs to compute the statistical expectation.

Theorem 23. *For each constant $k > 0$ there exists a polynomial time bounded probabilistic TM M such that* $\Pr[|M(C) - \mathrm{E}_{\mu^{\mathrm{uni}}}(\mathrm{time}_C)| \leq k] \geq \frac{2}{3}$.

Proof. Let T be the random variable of time_C with respect to the uniform probability distribution μ_n^{uni}. Further let $v \geq \mathrm{Var}(T)$ be an upper bound for the variance of T. For example $v = \mathrm{depth}(C)^2$ is a trivial upper bound for the variance, since $0 \leq \mathrm{time}_C(x) \leq \mathrm{depth}(C)$. Let M be a $\mathrm{O}(\mathrm{size}(C) \cdot m)$-time bounded prob. TM that computes the statistical expected value of $m := \frac{3 \cdot v}{k^2}$ independent random variables $\mathrm{time}_C(x_1), \ldots, \mathrm{time}_C(x_m)$.

For the variance of the output it holds: $\mathrm{Var}(M(C)) = \frac{\mathrm{Var}(T)}{m}$. Using the inequality of Chebyshev we get

$$\Pr[\,|M(C) - E(T)| \leq k] \geq 1 - \frac{\mathrm{Var}(T)}{m \cdot k^2} \geq \frac{2}{3}\,.$$

□

Corollary 24. *For each constant $k > 0$ there exists a polynomial time bounded probabilistic TM M such that* $\Pr[|M(C, D) - \mathrm{E}_{\mu_D}(\mathrm{time}_C)| \leq k] \geq \frac{2}{3}$.

Proof. The claim follows similarly to the proof of theorem 23 by replacing the uniformly chosen input in M with a μ_D distributed input.

Note that the time of this algorithm is $\mathrm{O}((\mathrm{size}(D) + \mathrm{size}(C)) \cdot v/k^2)$. If we choose for $v = \mathrm{depth}(C)^2$ we get $\mathrm{O}((\mathrm{size}(D) + \mathrm{size}(C)) \cdot \mathrm{depth}(C)^2/k^2)$. □

These algorithms approximate with high probability up to an arbitrary additive constant in polynomial time. Such a approximation rate is extraordinary, since normally approximation rate is measured by a factor.

5 Conclusion

Recent research in average compexity theory of circuit was directed towards the classification of Boolean functions. First results [JRS93, Reif93] analyzed upper and lower bounds for basic functions. Later more sophisticated analysis was done to find more general bounds for the expected time [JRSW93, JRS95]. Even for simple functions these analyses turned out to be quite complicated. For many functions this approach seems hopeless.

For a fixed circuit this analysis is not trivial at all, since we have proved that the complexity to determine the expected time of a circuit can be completely classified by $\#\mathcal{P}$.

Given	Question	Upper Bound	Lower Bound
C, x, t	$\text{time}_C(x) \leq t$	$\mathcal{P}$-complete	
C, t	$\exists x\ \text{time}_C(x) \leq t$	$\mathcal{NP}$-complete	
C, t	$t_{\mathtt{wc}}(C) = t$	$\mathcal{BH}_2$-complete	
C	x such that $\text{time}_C(x) = t_{\mathtt{wc}}(C)$	$\mathcal{FP}^{\mathcal{NP}}$	$\mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$-hard
C	lex. max. x with $\text{time}_C(x) = t_{\mathtt{wc}}(C)$	$\mathcal{FP}^{\mathcal{NP}}$-complete	
C, D	μ_D = worst case distr. of C	co-$\mathcal{NP}$-complete	
C, D	$E_{\mu_D}(\text{time}_C)$	$\#\mathcal{P}$-complete	

Table 1. Upper and lower bounds for problems investigated in this paper. C denotes a circuit, D a DG-circuit, x an input and t a time bound.

This concerns exact computation of the expectation, which in practice is not necessary. In this paper, an approximation algorithm is presented that solves this problem in polynomial time. Since it is a Monte-Carlo algorithm a standard technique can decrease the error probability; e.g. using this algorithm the expected time ± 1 of the circuit $\mathtt{ADD}_n$ presented in [JRSW93] can be computed within $O(n^2 \cdot \log^2 n)$ with probability $1 - \frac{1}{n}$.

When considering standard computer systems, the worst case also is of importance. Synchronized computation in particular needs the complete evaluation of all partial circuits within the clock. That means that the calculation of the minimal clock needs the computation of the worst time of all partial circuits. This problem is $\mathcal{BH}_2$-complete.

A trivial upper bound for the worst time is the depth, which can be determined in $\mathcal{NC}$. But depth and worst time can differ arbitrarily without loss of hardness.

So we propagate the concept of asynchronous computation, since it is not necessary to precompute such hard parameters like worst time. In this model for many functions there exist special designed circuits which provide a significant speedup in the average.

Note that the worst time of an asynchronous circuit gives a lower bound of the clock of a synchronized version. But the worst time requires a given worst case input. Since we have proved computing a worst case input is $\mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$-hard this implies high computational effort. Moreover, computation of the lexicographically maximum worst case input is even $\mathcal{FP}^{\mathcal{NP}}$-complete. Therefore we expect that asynchronous computer systems will compute faster than synchronous systems in general.

Table 1 gives an overview of the upper and lower bounds concerning time, worst time and expected time of circuits.

Open Problem Remember that CWC is defined as the search problem for the worst case input. We showed that it is $\mathcal{FP}^{\mathcal{NP}}_{\mathtt{tt}}$-hard. An upper bound is $\mathcal{FP}^{\mathcal{NP}}$. So the question remains whether CWC is $\mathcal{FP}^{\mathcal{NP}}$-complete.

Acknowledgement We thank our common teacher Rüdiger Reischuk for his advisement. Further we thank an unkown referee for very helpful suggestions, Barbara Goedecke for the right time, Gerhard Buntrock, Ulrich Hertrampf, Maciej Liskiewicz and Stephan Weis for discussions and pointers to the literature.

References

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

[BHPS94] B. Bollig, M. Hühne, S. Pölt, P. Savický, *On the Average Case Circuit Delay of Disjunction*, Technical Report, University of Dortmund, 1994.

[Cook71] S. A. Cook, *The Complexity of Theorem-Proving Procedures*, Proc. of the 3rd IEEE Symp. on the Foundations of Computer Science, 151-158, 1971.

[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.

[GHR91] R. Greenlaw, H. J. Hoover, W. L. Ruzzo, *A Compendium of Problems Complete for $\mathcal{P}$*, Technical Report 91-05-01, University of Washington, 1991.

[JRS93] A. Jakoby, R. Reischuk, C. Schindelhauer, *Circuit Complexity: from the Worst Case to the Average Case,* Proc. 26th SToC, 1994, 58-67; see also Technical Report, TH Darmstadt, 1993.

[JRS95] A. Jakoby, R. Reischuk, C. Schindelhauer, *Malign Distributions for Average Case Circuit Complexity,* Proc. 12th STACS, 1995, 628-639.

[JRSW93] A. Jakoby, R. Reischuk, C. Schindelhauer, S. Weis *The Average Case Complexity of the Parallel Prefix Problem,* Proc. 21st ICALP, 1994, 593-604; Technical Report, TH Darmstadt, 1993.

[Krap78] V. Krapchenko, Depth and Delay in a Network, Soviet Math. Dokl. 19, 1978, 1006-1009.

[Kren86] M. W. Krentel *The complexity of optimization problems,* Proc. 18th SToC, 1986, 79-86; see also J. CSS 36, 1988, 490-509.

[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.

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

[Milt93] P. Miltersen, *The Complexity of Malign Ensembles,* SIAM. J. Comput. 22, 1993, 147-156.

[PaYa82] C. H. Papadimitriou, M. Yannakakis, *The Complexity of Facets (and some facets of complexity)*, Proc. 14th SToC, 1982, 255-260; also, J.CSS 28, 1984, 244-259.

[Pa94] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1994.

[Reif93] J. Reif, *Probabilistic Parallel Prefix Computation,* Comp. Math. Applic. 26, 1993, 101-110.

[Toda90] S. Toda, *The Complexity of Finding Medians,* 31th FoCS, 1990, 778-787.