Relations among Parallel and Sequential Computation Models

Heribert Vollmer

Theoretische Informatik, Universität Würzburg, Am Exerzierplatz 3,
D-97072 Würzburg, Germany, vollmer@informatik.uni-wuerzburg.de

Abstract. This paper relates uniform small-depth circuit families as a parallel computation model with the complexity of polynomial time Turing machines. Among the consequences we obtain are: (a) a collapse of two circuit classes is equivalent to a relativizable collapse of the two corresponding polynomial time classes; and (b) a collapse of uniformity conditions for small depth circuits is equivalent to a related absolute (i.e., unrelativized) collapse in the polynomial time world.

1 Introduction

Small depth polynomial size circuit families are a very convenient computation model for abstract examinations of the power of *parallelism* [Ruz81]. AC^0, ACC^0, TC^0, and NC^1 are the most prominent complexity classes arising in this model. For example NC^1 corresponds to logarithmic time on parallel machine models (the PRAM for instance). The standard model for *sequential computation*, the Turing machine, has given rise to the definition of several complexity classes, but in the most interesting case, the restriction to polynomial time machines, no separations between classes are known. In contrast to this, one succeeded in proving certain circuit classes to be different from one another. E.g. it is known that $\mathrm{AC}^0 \subset \mathrm{ACC}^0 \subseteq \mathrm{TC}^0 \subseteq \mathrm{NC}^1$.

What is the connection between these two worlds? It is known that equality of two circuit classes translates upwards to equality of two polynomial time classes. However, only a partial converse is known [AW90]: In a number of cases a separation of *quasipolynomial size* circuit classes lead to an oracle separation of polynomial time classes [FSS84, Yao85, Hås88]. Quasipolynomial size circuits are getting increasingly popular among complexity theorists (see e.g. [AH94, Bar92]). One of the goals of this paper is to make the above connection precise. We relate quasipolynomial size small-depth circuits to polynomial time classes using the concept of the so called *leaf languages* [BCS92, Ver93]. This will allow us to prove an equivalence between (a) a collapse of quasipolynomial size circuit classes, and (b) a collapse of polynomial time classes holding for all oracles. This means that generally a separation of circuit classes is as difficult as constructing a separating oracle in the polynomial time world.

A second issue we address is that of the power of precomputation. In the circuit model this translates to the question how much computational power is needed to *construct* the circuit family. Again, we show how this relates to polynomial time classes. We prove that separability of polynomial time classes (in the absence of an oracle) is equivalent to separability of different precomputation modes for circuits.

In the next section, we will introduce the reader to circuit classes. We extensively discuss uniformity issues. In Sect. 3 we prove our general relationship between circuit and polynomial time classes. In Sect. 4 we consider the power of precomputation. We end with a short conclusion.

2 Preliminaries and Some Basic Facts

We assume the reader is familiar with basic complexity theory notions, see e.g. the textbooks [BDG95, BC94, Pap94].

As a computation model for parallel computation, very often families of boolean circuits are used. One single boolean circuit has of course a fixed number of input bits, say n, thus it can only work on input words of length exactly n. To define acceptance of languages via circuits, we have to consider circuit families $\{C_n\}_{n\in\mathbb{N}}$ where C_n is a circuit with n inputs. A language A is accepted by such a family if for every x, $x \in A$ if and only if x is accepted by circuit $C_{|x|}$. $\{C_n\}$ has size s and depth d, if for every n the number of gates in C_n is bounded by $s(n)$ and the length of the longest path from an input gate to an output gate is bounded by $d(n)$.

Such defined circuit families are of course a nonuniform computation model since for inputs of different lengths we have different acceptance devices. To make a circuit family uniform, i.e., to have one finite description of the whole infinite family, several different approaches have been considered in the literature: The first is to require that there is a deterministic Turing machine M that constructs the circuit family in the following sense: Given an input of length n, M outputs an encoding $\bar{C}_n$ of C_n. We say that $\{C_n\}$ is *logspace-uniform*, if this machine M is space bounded by $\log n$. Of course, logspace bounded machines are time-bounded by a suitable polynomial; thus this definition makes sense for at most polynomial size circuit families. For larger families, we need other space bounds. In this paper, we use *polylogspace-uniformity*, which requires simply that the space used by the constructing machine is bounded by $(\log n)^k$ for some $k \in \mathbb{N}$.

Another way of making circuit families uniform is the following: First define the so called *direct connection language* L_{DC} of $\{C_n\}$ to consist of those quadruples $\langle g, t, h, y\rangle$, where for $n = |y|$, g and h are numbers of gates in C_n, t is the type of g, and either h is an input gate for g or g is a gate for an input variable x_i, $1 \le i \le n$ [Ruz81]. Now we say that the circuit family is *dlogtime-uniform* (or simply, logtime-uniform), if there is a deterministic Turing machine M which accepts L_{DC} in time bounded by $O(\log n)$ [BIS90]. Similarly we speak of *polylogtime-uniformity*, when the run time is bounded by some $O((\log n)^k)$ ($k \in \mathbb{N}$) [Bar92], and of *P-uniformity*, if the run time is bounded by some polynomial. (This latter uniformity condition was considered by Allender in [All89].)

Some connections between these different conditions are known, e.g. for polynomial size and depth at least the square of the logarithm, logspace and logtime uniformity coincide [Ruz81]. But in general, different conditions may yield different classes.

Uniformity conditions directly reflect the intuitive concept of precomputation. Thus, different uniformity conditions give rise to different precomputation modes. It will be one of our goals in this paper to compare different such modes.

Let us say that a circuit family is of quasipolynomial size, if its size is bounded by some $2^{(\log n)^k}$ for some $k \in \mathbb{N}$. The following analogue of a result of Ruzzo [Ruz81] can be proved:

Proposition 1. *A quasipolynomial size circuit family is polylogspace-uniform, if and only if its direct connection language can be recognized within polylog space bounds.*

Now we define the following classes: AC^0 consists of all languages accepted by polynomial size, constant depth circuit families with unbounded fan-in AND and OR gates. ACC^0 is defined by the same resource bounds but now allowing arbitrary Mod_m gates, i.e. gates whose output is one if the number of its inputs that are 1 is divisible by m. For TC^0 we allow majority gates, i.e. gates whose output is 1 iff the majority of its inputs are 1. NC^1 is the class of languages accepted by polynomial size, logarithmic depth circuits with fan-in 2 AND and OR gates. $\mathrm{AC}^0 \subset \mathrm{ACC}^0 \subseteq \mathrm{TC}^0 \subseteq \mathrm{NC}^1$ is known, and all classes can be simulated within logarithmic space on Turing machines. The classes qAC^0, qACC^0, qTC^0 are defined analogously, but now allowing quasipolynomial circuit size. In the case of qNC^1, a straightforward translation of the definition of NC^1 to quasipolynomial size does not lead to a new class, because logarithmic depth bounded fan-in circuits cannot have more than polynomially many gates. For the purpose of this paper, we define qNC^1 to be the qAC^0 closure of NC^1. Again, $\mathrm{qAC}^0 \subset \mathrm{qACC}^0 \subseteq \mathrm{qTC}^0 \subseteq \mathrm{qNC}^1$ is known, furthermore it is not too hard to see that qNC (the class of quasipolynomial size polylog depth circuits) corresponds exactly to polylogarithmic space on Turing machines. For this and more on the subtle relations between qNC^1, qNC and their polysize counterparts see [Bar92].

To make precise, which uniformity we are talking about, we will write for example logspace-uniform AC^0 or polylogspace-uniform qAC^0. If we use no prefix of the form "...-uniform", then we implicitly assume logtime uniformity in case of polynomial size, and polylogtime uniformity in case of quasipolynomial size.

From the polynomial time context, we will need the following classes: The polynomial time hierarchy is the hierarchy of classes one obtains starting with P (deterministic polynomial time) and applying to this class the polynomially bounded quantifiers $\exists^p$ and $\forall^p$, see [Wra77] or the textbook [BC94]. PH is the union of all classes in the polynomial time hierarchy. The counting hierarchy is the class obtained in the same way but now allowing additionally the polynomially bounded counting quantifier C^p, essentially a majority quantifier over an exponential range [Wag86, Tor91]. CH is the union of all classes in the counting hierarchy [Tor91]. The polynomial time mod hierarchy is the hierarchy of classes obtained as above, but now allowing polynomially bounded existential, universal, Mod_2, Mod_3, ... quantifiers. (Mod quantifiers which are an adaption of the above defined Mod gates to the context of polynomial time were defined in [BGH90].) MOD-PH is the union of all classes in the polynomial time mod hierarchy.

The adaptions of the above defined classes to the logarithmic time context (where we have quantifiers ranging over logarithmically long strings, and apply these quantifiers to the base class DLOGTIME) are the logarithmic time hierarchy LH [CKS81], the logarithmic time mod hierarchy MOD-LH, and the logarithmic time counting hierarchy LCH [Tor91]. The notations we use for the polylogarithmic time case (quantifiers ranging over polylogarithmically long strings) are PLH, MOD-PLH, and PLCH.

A remark now is in order about Turing machines working in sublinear time: To give these machines the possibility to access all their input bits within their runtime, we cannot use the usual input tape with a read head which can be moved to the left or right in one time step. Instead, our machines have a special index tape on which they write in binary numbers of input bits. The input head moves always in one time step to the bit whose position can be found on the index tape. It should be remarked that there are several similar but slightly different ways to define sublinear time Turing machines with in general different power, but in the case of recognition of direct connection languages, these are all equivalent as was observed in [RV95].

The following connection between circuit classes and classes of Turing machines operating in sublinear time are known [BIS90]: A set is in AC^0 iff it is in LH; a set is in ACC^0 iff it is in MOD-LH; and finally a set is in TC^0 iff it is in LCH. It can be seen that these characterizations carry over to the case of quasipolynomial size circuit classes ($qAC^0 = PLH$, $qACC^0 = MOD\text{-}PLH$, and $qTC^0 = PLCH$).

The question which now arises is of course: Can logspace (or polylogspace) uniform circuit classes also be characterized on sublinear time Turing machines? The following proposition can be proved with methods similar to those used by Allender in [All89], which in turn rely on simulations given in [Ruz81].

Proposition 2. *All logspace-uniform circuit classes (polylogspace-uniform circuit classes, resp.) mentioned above can be characterized by logspace-bounded machines (polylogspace-bounded machines, resp.) of the same type as in the corresponding logtime-uniform case, which are allowed to branch and read their input only during a first phase of their computation which is time-bounded by some logarithmic (polylogarithmic, resp.) function, and afterwards behave deterministically without reading any more input bits.*

What we want to do in the upcoming sections of our paper is to relate quasipolynomial size circuit classes to polynomial time Turing machine classes. For this, we will use padding techniques (see e.g. [Boo74]) in a slightly elaborated form, which has become known under the name *leaf language approach* to the examination of complexity classes. Leaf languages were first defined and examined independently in [BCS92] and [Ver93] and later considered in a number of papers, e.g. [HLS+93, JMT94, HVW96], cf. also the recent textbook [Pap94, pp. 504f]. We use the following definition: Let A, B be two sets. We say that A is polynomial time bit-reducible to B, $A \le_m^{\mathrm{p,bit}} B$, iff there exist two functions f, g computable deterministically in polynomial time where f is zero-one-valued such that $x \in A \iff f(x,1)f(x,2)\cdots f(x,g(x)) \in B$. Analogously, we define polylogarithmic time bit-reducibility, $\le_m^{\mathrm{plt,bit}}$, requiring that in the above f and g are computable in polylogarithmic time. Let $\mathcal{R}_m^{\mathrm{p,bit}}(B) =_{\mathrm{def}} \{\, A \mid A \le_m^{\mathrm{p,bit}} B \,\}$ and for a class of sets $\mathcal{C}$ let $\mathcal{R}_m^{\mathrm{p,bit}}(\mathcal{C}) =_{\mathrm{def}} \bigcup_{B \in \mathcal{C}} \mathcal{R}_m^{\mathrm{p,bit}}(B)$. The class $\mathcal{R}_m^{\mathrm{p,bit}}(B)$ can best be visualized as the class of those sets A accepted by nondeterministic Turing machines producing exactly for words in A a *leaf string* from B; that is take the computation tree which is defined by the nondeterministic machine, order the paths according to, say, the order of entries in the Turing table of the machine. Then form a word of zeroes and ones, one symbol for each path, zero for rejecting and one for accepting. This word, the leaf string of the machine, has to be in B iff the input is in A. For more on this (especially the issue of balanced versus unbalanced computation trees which we did not address here), consult the above cited papers. The notation for $\mathcal{R}_m^{\mathrm{p,bit}}(B)$ in [BCS92] is $\mathcal{C}(B, \overline{B})$. The notation in [JMT94] is $\text{Balanced-Leaf}^{\mathrm{P}}(B)$.

The fundumantal result in connection with these classes, proved by Bovet, Crescenzi, and Silvestri in [BCS92] and independently by N. Vereshchagin in [Ver93], is the following:

Proposition 3. *For sets A, B we have: $A \le_m^{\mathrm{plt,bit}} B$ if and only if for all oracles Y, $\mathcal{R}_m^{\mathrm{p,bit}}(A)^Y \subseteq \mathcal{R}_m^{\mathrm{p,bit}}(B)^Y$.*

In connection with circuits, we will need a form of many-one reducibility, which we define as follows: $A \le_m^{\mathrm{qAC}^0} B$ iff A is many-one reducible to B via a function f which can be computed by qAC^0 circuits. Obviously all constant depth quasipolynomial size classes are closed under this reducibility, but also qNC^1 since, as noted in [Bar92], qNC^1 can be characterized by constant depth circuits with gates for any regular language. Thus we proved:

Proposition 4. *All considered quasipolynomial size circuit classes are closed under $\le_m^{\mathrm{qAC}^0}$.*

3 Comparing Different Gate Types

In [HLS+93] the time complexity of a complexity class was related to the time complexity of its closure under $\mathcal{R}_m^{\mathrm{p,bit}}(\cdot)$. However as stated the result holds only for at least linear time bounds. Some extensions for lower time bounds were given in [JMT94]. Here we need the following result:

Lemma 5. *For $\mathcal{X} \in \{\mathrm{D}, \mathrm{N}, \Sigma_k, \mathrm{C}, \mathrm{Mod}_2, \mathrm{Mod}_3, \ldots\}$,*

$$\mathcal{X}\mathrm{TIME}(n^{O(1)}) = \mathcal{R}_m^{\mathrm{p,bit}}(\mathcal{X}\mathrm{TIME}(\log^{O(1)} n)).$$

Proof. The proof follows closely those given for the related results in [HLS+93, JMT94, HVW96]. In order to keep the paper self-contained we give a brief sketch.

For the direction from left to right suppose that we are given a set $A \in \mathcal{X}\mathrm{TIME}(n^k)$. Then

$$A' =_{\mathrm{def}} \left\{ a_1 a_2 \cdots a_{2^n} \;\middle|\; a_2 a_4 a_8 \cdots a_{2^n} \in A \right\}$$

is in $\mathcal{X}\mathrm{TIME}((\log n)^k)$. The transformation from A to A' can certainly be performed by a $\leq_m^{\mathrm{p,bit}}$ reduction.

For the direction from right to left, it is sufficient to note that we simply have to compose the $\leq_m^{\mathrm{p,bit}}$ reduction with the polylogtime decision procedure with acceptance type $\mathcal{X}$ which yields polynomial time algorithms again working according to acceptance type $\mathcal{X}$. □

Recall the machine characterizations of our circuit classes mentioned above in Sect. 2. We thus get:

Corollary 6.

1. $\mathcal{R}_m^{\mathrm{p,bit}}(\mathrm{qAC}^0) = \mathrm{PH}$.
2. $\mathcal{R}_m^{\mathrm{p,bit}}(\mathrm{qACC}^0) = \mathrm{MOD\text{-}PH}$.
3. $\mathcal{R}_m^{\mathrm{p,bit}}(\mathrm{qTC}^0) = \mathrm{CH}$.
4. $\mathcal{R}_m^{\mathrm{p,bit}}(\mathrm{qNC}^1) = \mathrm{PSPACE}$.

What we would like to do next is use Proposition 3 to relate inclusions between quasipolynomial size circuit classes to relativizable inclusions between polynomial time Turing machine classes. To do so, we first have to establish a certain closure property of the former classes:

Theorem 7. *Let A, B be sets such that $A \leq_m^{\mathrm{plt,bit}} B$. Then $A \leq_m^{\mathrm{qAC}^0} B$.*

Proof. Let $A \leq_m^{\mathrm{plt,bit}} B$ via f. Then there exist functions g, h that are computable deterministically in polylogarithmic time such that for all x, $f(x) = g(x,1)g(x,2)\cdots g(x,h(x))$.

Consider some position i, $1 \leq i \leq h(x)$, in the output of $f(x)$. The bit $g(x,i)$ depends on at most $(\log n)^k$ bits of x (for some $k \in \mathbb{N}$). This means that we can set up a truth table for $g(x,i)$ of size $2^{(\log n)^k}$. Such a truth table corresponds immediately to a depth 2 circuit with unbounded fan-in AND and OR gates.

Combining these circuits for all output bits yields the desired $\leq_m^{\mathrm{qAC}^0}$ reduction. Uniformity of the construction follows by the following considerations: Choose a gate numbering such that from the gate number it is possible to recover the line in the truth table the gate corresponds to. This line number in binary is identical to the sequence of the bits which have to be read to evaluate $g(x,i)$. Thus, given the line number we can follow an at most polylogarithmically long computation path of the machine computing g and see, if the line in the truth table evalues to one given the input x. In such a way the connections between the AND gates on level 1 and the inputs can be checked. The next layer of our depth two circuit is as follows: At the top we have an OR gate which is connected to those AND gates that correspond to input combination which make $g(x,i)$ one. This can also be checked following paths in the machine.

Thus it is possible in polylogarithmic time to check connections in the resulting circuit, which makes the reduction polylogtime-uniform. $\square$

Corollary 8. *All quasipolynomial size circuit classes that we consider are closed under* $\leq_m^{\mathrm{plt,bit}}$.

Proof. Immediate from Proposition 4 and the just given theorem. $\square$

Corollary 9.

1. $\mathrm{qTC}^0 = \mathrm{qNC}^1$ *if and only if for all oracles* Y, $\mathrm{CH}^Y = \mathrm{PSPACE}^Y$.
2. $\mathrm{qACC}^0 = \mathrm{qNC}^1$ *if and only if for all oracles* Y, $\text{MOD-PH}^Y = \mathrm{PSPACE}^Y$.
3. $\mathrm{qACC}^0 = \mathrm{qTC}^0$ *if and only if for all oracles* Y, $\text{MOD-PH}^Y = \mathrm{CH}^Y$.

Proof. The implication from left to right follows in all three cases immediately from Corollary 6. The implication from right to left can be seen as follows: Suppose e.g. the right hand side of the first statement holds. Then by Proposition 3 every set $A \in \mathrm{qNC}^1$ is $\leq_m^{\mathrm{plt,bit}}$ reducible to a set in qTC^0, thus by Corollary 8, $A \in \mathrm{qTC}^0$. $\square$

4 Comparing Uniformity Conditions

In [CMTV96] it was shown that $\mathcal{R}_m^{\mathrm{p,bit}}(\text{logspace-uniform}\,\mathrm{AC}^0) = \mathrm{PSPACE}$. Thus we have immediately:

Lemma 10. *If logspace-uniform* $\mathrm{AC}^0 \subseteq \mathcal{C}$, *then* $\mathrm{PSPACE} \subseteq \mathcal{R}_m^{\mathrm{p,bit}}(\mathcal{C})$.

In the above take for example $\mathcal{C} = \mathrm{AC}^0$. In other words, we assume that logspace-uniform $\mathrm{AC}^0 \subseteq \mathrm{AC}^0$. Then we get a collapse of PH to PSPACE. The question which now arises naturally is if the converse also holds[1]. It turns out that in the case of quasipolynomial circuit classes, i.e. choosing e.g. $\mathcal{C} = \mathrm{qAC}^0$, the answer is yes.

Theorem 11.

1. *If* $\mathrm{PH} = \mathrm{PSPACE}$, *then logspace-uniform* $\mathrm{AC}^0 \subseteq \mathrm{qAC}^0$.
2. *If* $\text{MOD-PH} = \mathrm{PSPACE}$, *then logspace-uniform* $\mathrm{ACC}^0 \subseteq \mathrm{qACC}^0$.
3. *If* $\mathrm{CH} = \mathrm{PSPACE}$, *then logspace-uniform* $\mathrm{TC}^0 \subseteq \mathrm{qTC}^0$.

[1] In [CMTV96] the authors (including the present author) claimed that PH = PSPACE iff AC^0 = logspace-uniform AC^0. However, their proof relied on a wrong assumption.

Proof. We only give the proof for the first statement since the remaining two proofs are very similar. The proof follows that of Theorem 12 in [AG91].

Assume PH = PSPACE and let A be in logspace-uniform AC^0. Then there is a circuit family $\{C_n\}$ for A such that the set

$$\bar{C} =_{\mathrm{def}} \{1^n 0^i \mid \text{bit } i \text{ in the description of } C_n \text{ is } 1\}$$

is in logspace. Thus,

$$\bar{C}' =_{\mathrm{def}} \{\langle n,i\rangle \mid \text{bit } i \text{ in the description of } C_n \text{ is } 1\} \in \mathrm{PSPACE}$$

and thus by assumption in PH, say in Σ_j-TIME(n^k). But then,

$$\{\langle n,i\rangle 10^{|\langle n,i\rangle|^k} \mid \text{bit } i \text{ in the description of } C_n \text{ is } 1\} \in \Sigma_j\text{-TIME}(n)$$

and thus,

$$\{1^n 0^i 10^{2^{|\langle n,i\rangle|^k}} \mid \text{bit } i \text{ in the description of } C_n \text{ is } 1\}$$

is in the logtime hierarchy which is known to be equal to (dlogtime-uniform) AC^0. Observe, that $|1^n 0^i 10^{2^{|\langle n,i\rangle|^k}}|$ is quasipolynomial in n.

It was shown in [AG91] that for every k the set L_k consisting of all pairs $\langle C,x\rangle$ such that C is a depth k circuit consisting of AND and OR gates that evaluates to 1 on input x is in AC^0. Thus, $A \in \mathrm{qAC}^0$ by the circuit family consisting of subcircuits constructing on input x, $|x| = n$, a description of C_n and then evaluating it on input x. This family is polylogtime-uniform. □

Thus, we have the following equivalences:

Corollary 12.

1. PH = PSPACE *if and only if logspace-uniform* $\mathrm{AC}^0 \subseteq \mathrm{qAC}^0$.
2. MOD-PH = PSPACE *if and only if logspace-uniform* $\mathrm{ACC}^0 \subseteq \mathrm{qACC}^0$.
3. CH = PSPACE *if and only if logspace-uniform* $\mathrm{TC}^0 \subseteq \mathrm{qTC}^0$.

Above, we related polynomial size circuits with powerful precomputation to quasipolynomial size circuits with a less powerful precomputation. Next, we concentrate on quasipolynomial circuit size and compare uniformity conditions within that model. It turns out that the above results go through.

Corollary 13.

1. PH = PSPACE *if and only if polylogspace-uniform* $\mathrm{qAC}^0 = \mathrm{qAC}^0$.
2. MOD-PH = PSPACE *if and only if polylogspace-uniform* $\mathrm{qACC}^0 = \mathrm{qACC}^0$.
3. CH = PSPACE *if and only if polyglogspace-uniform* $\mathrm{qTC}^0 = \mathrm{qTC}^0$.

Proof. The direction from right to left is alway obvious. For the other direction, it is sufficient to observe that reproducing the proof of Theorem 11 the set $\bar{C}$ is now in POLYLOGSPACE, but again $\bar{C}' \in$ PSPACE. □

Combining the just given two corollaries, we get an interesting connection relating different uniformity conditions for differenct circuit classes (this result could of course also be proved in a more direct way):

Corollary 14.

1. *If polylogspace-uniform* $\mathrm{qAC}^0 \subseteq \mathrm{qACC}^0$*, then polylogspace-uniform* $\mathrm{qACC}^0 = \mathrm{qACC}^0$.
2. *If polylogspace-uniform* $\mathrm{qAC}^0 \subseteq \mathrm{qTC}^0$*, then polylogspace-uniform* $\mathrm{qTC}^0 = \mathrm{qTC}^0$.

5 Conclusion

We made the connection between quasipolynomial size circuit classes and polynomial time classes, exploited already a few times [FSS84, Yao85, Hås88], precise by giving a general theorem relating the two worlds. For this relation we used the concept of leaf languages for polynomial time machines. We showed that constructing an oracle separating polynomial time classes is *equivalent* to an (absolut) separation of circuit classes. As a second application of our general relation, we examined the power of precomputation by examining different uniformity conditions. We showed that to prove different precomputation modes to be of different power, one has to separate related polynomial time classes absolutely, that is in the absence of an oracle.

Our results are interesting since they reflect the current state of the art in separating circuit classes. Separations of circuit classes are often easier than separations of polynomial time classes, since by the above one "only" has to construct an oracle separating polynomial time classes to get a circuit separation. On the other hand, oracle constructions can be very hard, and e.g. no oracle separating the counting hierarchy from PSPACE is known. Translated into questions about circuits, the question $\mathrm{TC}^0 \stackrel{?}{=} \mathrm{NC}^1$ (and $\mathrm{qTC}^0 \stackrel{?}{=} \mathrm{qNC}^1$) is still open.

Acknowledgement. Thanks are due to Eric Allender, New Brunswick, and Klaus-Jörn Lange, Tübingen, for helpful discussions.

References

[AG91] E. Allender and V. Gore. On strong separations from AC^0. In *Proceedings 8th Fundmentals of Computation Theory*, volume 529 of *Springer Lecture Notes in Computer Science*, pages 1–15, 1991.

[AH94] E. Allender and U. Hertrampf. Depth reduction for circuits of unbounded fan-in. *Information & Computation*, 112:217–238, 1994.

[All89] E. Allender. P-uniform circuit complexity. *Journal of the Association for Computing Machinery*, 36:912–928, 1989.

[AW90] E. Allender and K. W. Wagner. Counting hierarchies: polynomial time and constant depth circuits. *Bulleting of the EATCS*, 40:182–194, 1990.

[Bar92] D. A. Mix Barrington. Quasipolynomial size circuit classes. In *Proceedings 7th Structure in Complexity Theory*, pages 86–93. IEEE Computer Society Press, 1992.

[BC94] D. P. Bovet and P. Crescenzi. *Introduction to the Theory of Complexity.* International Series in Computer Science. Prentice Hall, 1994.

[BCS92] D. P. Bovet, P. Crescenzi, and R. Silvestri. A uniform approach to define complexity classes. *Theoretical Computer Science*, 104:263–283, 1992.

[BDG95] J. L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* Springer Verlag, 2nd edition, 1995.

[BGH90] R. Beigel, J. Gill, and U. Hertrampf. Counting classes: thresholds, parity, mods, and fewness. In *Proceedings 7th Symposium on Theoretical Aspects of Computer Science*, volume 415 of *Lecture Notes in Computer Science*, pages 49–57. Springer-Verlag, 1990.

[BIS90] D. A. Mix Barrington, N. Immerman, and H. Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306, 1990.

[Boo74] R. V. Book. Tally languages and complexity classes. *Information and Control*, 26:186–194, 1974.

[CKS81] A. K. Chandra, D. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28:114–133, 1981.

[CMTV96] H. Caussinus, P. McKenzie, D. Thérien, and H. Vollmer. Nondeterministic NC^1 computation. In *Proceedings 11th Computational Complexity*, pages 12–21. IEEE Computer Society Press, 1996.

[FSS84] M. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.

[Hås88] J. Håstad. *Computational Limitations of Small Depth Circuits.* MIT Press, Cambridge, 1988.

[HLS+93] U. Hertrampf, C. Lautemann, T. Schwentick, H. Vollmer, and K. W. Wagner. On the power of polynomial time bit-reductions. In *Proceedings 8th Structure in Complexity Theory*, pages 200–207, 1993.

[HVW96] U. Hertrampf, H. Vollmer, and K. W. Wagner. On balanced vs. unbalanced computation trees. *Mathematical Systems Theory*, 29:411–421, 1996.

[JMT94] B. Jenner, P. McKenzie, and D. Thérien. Logspace and logtime leaf languages. In *9th Annual Conference Structure in Complexity Theory*, pages 242–254, 1994.

[Pap94] C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[Ruz81] W. L. Ruzzo. On uniform circuit complexity. *Journal of Computer and Systems Sciences*, 21:365–383, 1981.

[RV95] K. Regan and H. Vollmer. Gap-languages and log-time complexity classes. Technical report, Department of Computer Science, SUNY Buffalo, 1995. Submitted for publication.

[Tor91] J. Toran. Complexity classes defined by counting quantifiers. *Journal of the ACM*, 38:753–774, 1991.

[Ver93] N. Vereshchagin. Relativizable and non-relativizable theorems in the polynomial theory of algorithms. *Izvestija Rossijskoj Akademii Nauk*, 57:51–90, 1993.

[Wag86] K. W. Wagner. Some observations on the connection between counting and recursion. *Theoretical Computer Science*, 47:131–147, 1986.

[Wra77] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3:23–33, 1977.

[Yao85] A. C. C. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings 26th Foundations of Computer Science*, pages 1–10. IEEE Computer Society Press, 1985.