BRICS

Basic Research in Computer Science

BRICS RS-99-46 Miltersen et al.: Half-Exponential Circuit Size in the Exponential Hierarchy

Super-Polynomial Versus Half-Exponential Circuit Size in the Exponential Hierarchy

Peter Bro Miltersen
Vinodchandran N. Variyam
Osamu Watanabe

BRICS Report Series **RS-99-46**

ISSN 0909-0878 **December 1999**

See back inner page for a list of recent BRICS Report Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark

Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/99/46/`

Super-polynomial versus half-exponential circuit size in the exponential hierarchy

Peter Bro Miltersen* N. V. Vinodchandran*
Osamu Watanabe †

December, 1999

Abstract

Lower bounds on circuit size were previously established for functions in Σ_2^p, $\mathrm{ZPP}^{\mathrm{NP}}$, Σ_2^{exp}, $\mathrm{ZPEXP}^{\mathrm{NP}}$ and $\mathrm{MA}_{\mathrm{exp}}$. We investigate the general question: Given a time bound $f(n)$. What is the best circuit size lower bound that can be shown for the classes MA-TIME$[f]$, ZP-TIME$^{\mathrm{NP}}[f], \ldots$ using the techniques currently known? For the classes $\mathrm{MA}_{\mathrm{exp}}$, $\mathrm{ZPEXP}^{\mathrm{NP}}$ and Σ_2^{exp}, the answer we get is "half-exponential". Informally, a function f is said to be half-exponential if f composed with itself is exponential.

1 Introduction

One of the main issues of complexity theory is to investigate how powerful non-uniform (e.g. circuit based) computation is, compared to uniform (machine based) computation. In particular, a 64K dollar question is whether exponential time has polynomial size circuits. This being a challenging open question, a series of papers have looked at circuit size of functions further up

*BRICS, Department of Computer Science, University of Aarhus. bromille,vinod@brics.dk. Supported by the ESPRIT Long Term Research Programme of the EU under project number 20244 (ALCOM-IT). Part of N. V. Vinodchandran's work was done while he was a Junior Research Fellow at the Institute of Mathematical Sciences, Chennai 600 113, India.

†Department of Mathematical and Computing Sciences, Tokyo Institute of Technology. Meguro-ku Ookayama, Tokyo 152-8552. watanabe@is.titech.ac.jp. Supported in part by JSPS/NSF International Collaboration Grant.

the exponential hierarchy. In the early eighties, Kannan [10] established that there are languages in $\Sigma_2^{\text{exp}} \cap \Pi_2^{\text{exp}}$ which do not have circuits of polynomial size. Later, using methods from learning theory, in particular Bshouty et al [6], Köbler and Watanabe [12] improved Kannan's result and showed that in fact ZPEXP^{NP} contains languages which do not have polynomial size circuits. Recently, Buhrman, Fortnow and Thierauf [7] improved it further to show that even the exponential version of the interactive class MA does not have polynomial size circuits. More precisely, they showed that $\text{MA}_{\text{exp}} \cap \text{coMA}_{\text{exp}} \not\subseteq \text{P/poly}$. This is an improvement since it follows from the result $\text{MA} \subseteq \text{ZPP}^{\text{NP}}$ (due to Arvind and Köbler [1], and independently, to Goldreich and Zuckerman [8]) using padding that $\text{MA}_{\text{exp}} \subseteq \text{ZPEXP}^{\text{NP}}$. In contrast to the previous results, their proof uses non-relativizable techniques.

Some of the lower bounds mentioned above have analogous versions in the polynomial hierarchy. For instance Kannan showed that for any k, $\Sigma_2^p \cap \Pi_2^p$ contains a language that does not have circuits of size n^k. Köbler and Watanabe showed the same statement with $\Sigma_2^p \cap \Pi_2^p$ replaced with ZPP^{NP}. However, interestingly, it is not known if MA has linear sized circuits. So there the analogy seemingly stops. It seems interesting to understand the scenario better, so we consider the following question. *What is the smallest time bound for which* $\text{MA-TIME}[f]$ *is known not to have linear sized circuits?* Also, it is natural to ask how far the techniques can be extended if one is not happy with just super-polynomial bounds. For example, do the techniques of Kannan in fact show that there are languages in Σ_2^{exp} that require exponential size circuits? In general, we can ask the following question: *Given a time bound f. What is the best circuit size lower bound we can show for functions in* $\text{MA-TIME}[f], \text{ZP-TIME}^{\text{NP}}[f]$*,* Σ_2^f*?* One of the main objectives of this paper is to investigate these questions.

The complexity (in the uniform setting) of functions with *exponential* circuit complexity is of particular interest (note that a random function will have exponential circuit complexity). In this direction, the only previous work seems to be done by Kannan [10]. Indeed, Kannan shows that the third level of the exponential hierarchy, and in fact, $\Sigma_3^e \cap \Pi_3^e$, contains a function with *maximum* circuit size. As was shown by Shannon and Lupanov, this happens to be $\Theta(2^n/n)$.

In [10], the author also makes claim to the following statement (*) [10, Theorem 4, page 48], apparently answering one of the questions we asked in the beginning of this paper:
(*) *There is a universal constant l such that for any time-constructible function $f(\cdot)$ satisfying $n^l \leq f(n) \leq 2^{n/20} \forall n$, there is a language in $\Sigma_2^{f(n)} \cap \Pi_2^{f(n)}$ that does not have $O((f(n))^{1/l})$-size circuits.*

In particular, statement (*) implies that the second level of the exponential hierarchy does not have $2^{o(n)}$ sized circuits.

Though statement (*) is very likely to be a true, it is not clear to us that it was, in fact, given a proof in [10]. Nor do we know how to prove it. We suggest that the statement is reopened and considered an open problem. We analyse this issue in Section 3. In this context, we note that the bound of $\Sigma_3^e \cap \Pi_3^e$ for functions requiring maximum size, can be improved to Δ_3^e by using a binary search approach. This may be a folklore, but does not seem to be explicitly mentioned anywhere in the literature. Also, we need this improvement for some of the statements we prove later.

2 Notations, definitions and results

We assume the definitons of standard compelxity classes. Please refer to [5, 14] for these and other standard complexity-theoretic definitions.

Let $f(n) \geq n$ be a time constructible function. Then the complexity classes of interest to us (with time bound f) are TIME$[f]$, $\Sigma_k^f, \Pi_k^f, \Delta_k^f$, ZP-TIME$^{\mathrm{NP}}[f]$ (class of languages recognized by a randomized Turing machine with an NP-oracle, running in *expected* time $f(n)$ and always returning the right answer) and MA-TIME$[f]$ (class of languages recognized by a Merlin-Arthur game, where the length of Merlin's proof and the time of Arthur's computation is bounded by $f(n)$). For any class of time bounds $\mathcal{F}$ and any of the the above-mentioned complexity class $\mathcal{C}$, we denote the class $\cup_{f \in \mathcal{F}} \mathcal{C}$ by $\mathcal{C}[\mathcal{F}]$. The polynomial and exponential versions of these classes are of special interest. For polynomial versions the notations are standard. The notations for the exponential versions that we use are EXP, $\Sigma_k^{\exp}, \Pi_k^{\exp}, \Delta_k^{\exp}$, ZPEXP$^{\mathrm{NP}}$ and MA$_{\exp}$, respectively . SIZE$[f]$ denotes the class of languages accepted by circuit families of size bounded by $f(n)$ for sufficiently large n.

Furthermore, we let Σ_k^e (Π_k^e, Δ_k^e) denote the class $\cup_f \Sigma_k^f$ $(\cup_f \Pi_k^f, \cup_f \Delta_k^f$ respectively) where the union is over all $f \in 2^{O(n)}$.

2.1 Fractionally exponentially growing functions

First we motivate the definition of such functions. One of the main difficulties in answering the questions posed in the introduction is that the "best" lower bounds to be obtained are not easily expressible in terms of conventional mathematical notation. Usually, lower bounds in complexity theory can be described with expressions involving the operations $\{+, -, *, \exp, \log\}$ only. Growth rates so expressible are called *L-functions* by Hardy [9]. Unfortunately, the answers we get to the questions posed in the introduction involves functions that are not approximated well by any L-function. For instance, the best lower bound that can be shown using current techniques for the classes $\mathrm{MA}_{\mathrm{exp}}$, $\mathrm{ZPEXP}^{\mathrm{NP}}$, or Σ_2^{exp} seems to be "half-exponential", i.e. it is a bound that, composed with itself, becomes exactly exponential. Any L-function with a smaller growth rate (making it a valid substitute in a lower bound statement) will have, in fact, much, much smaller growth rate, and thus make the statement much weaker.

Intuitively, the notions of $\frac{1}{2}$-exponentially, or even $\frac{1}{k}$-exponentially growing functions are clear. One naive approach of defining them is as follows: We say that a "nice" function f has at most, say, half-exponential growth if $f(f(n)) \leq 2^{p(n)}$ for some polynomial p and all n. With an appropriate interpretation of "nice", such a definition would be adequate for some of the statements we prove, such as "For any half-exponential function f, there is a problem g in $\mathrm{MA}_{\mathrm{exp}}$, so that g does not have circuits of size f". However, it is not obvious how to generalize this approach to meaningfully express and prove statements such as "For any (7/8)-exponential function f, there is problem g in MA-TIME$[\exp_{14/8}]$, so that g does not have circuits of size f". Also, suppose we want to express our lower bounds in terms of a statement such as "There are function in complexity class C requiring circuit size at least $f(n)$" for some *specific* function f. Then it is not obvious how to pick such function f in a way close to optimal, i.e., as fast growing as possible. This seems unsatisfactory: To get any feeling for what the lower bound really means, it seems desirable to have concrete examples of functions of *exactly*, say, half-exponential growth in mind.

With these issues in mind, we take the following approach (see [16] for a discussion). Let $e(x) = e^x - 1$, where e denotes the base of the natural logarithm. Consider the following functional equation (called Abel's equation in the mathematics literature).

$$A(e(x)) = A(x) + 1. \qquad (1)$$

Let $A : R_+ \rightarrow R_+$ be a fixed solution to this (R_+ denote the set of positive real numbers). Then, with respect to this solution we can define the *α-iterate* of $e(x)$ as; $e_\alpha(x) = A^{-1}(A(x) + \alpha)$. Now, define the class of time bounds

$$\exp_\alpha = \{f : \mathbf{N} \rightarrow \mathbf{N} \mid f \text{ is time constructible },\exists k \forall n, f(n) \leq e_\alpha(n^k)\}.$$

In order to use these class of functions as complexity bounds, we would like the following robustness property to hold among these classes of functions. Let α, β rationals. Then for $f \in \exp_\alpha$ and $g \in \exp_\beta$, we would like $f(g(.))$ to be in $\exp_{\alpha+\beta}$. There exist solutions for Equation 1 which give rise to functions with this property. The one due to Szekeres [16] is an example (please refer to [15] for a proof this). We use this property in many of our proofs, often with out making any explicite reference to it.

Time constructibility of these functions is a more subtle issue. In [13], the authors give a numerical procedure for approximating $e_\alpha(x)$. We strongly believe that a rigorous analysis of the procedure given in [13] will give us the time constructibility of these functions also. This analysis may require finding out the rate of convergence of the procedure given.

2.2 New results in this paper

We first show the extension of the lower bounds shown in [12] and [7] to get a general lower bound result. More precisely, we show the following theorems.

Theorem 1 *For any rational value c, $0 \leq c < 1$, and any $k \geq 1$, there is a language $L_{c,k}$ in $\text{ZP-TIME}^{\text{NP}}[\exp_{2c}]$, so that, for infinitely many n, $L_{c,k} \cap \{0,1\}^n$ is not computed by a circuit of size $e_c(n^k)$. This holds relative to any oracle.*

Theorem 2 *For any rational value c, $0 \leq c \leq \frac{1}{2}$, and any k, there is a language $L_{c,k}$ in $\text{MA-TIME}[\exp_{c+\frac{1}{2}}] \cap \text{coMA-TIME}[\exp_{c+\frac{1}{2}}]$, and, for any $\frac{1}{2} \leq c < 1$, and any k, there is a language $L_{c,k}$ in $\text{MA-TIME}[\exp_{2c}] \cap \text{coMA-TIME}[\exp_{2c}]$, so that, for infinitely many n, $L_{c,k} \cap \{0,1\}^n$ cannot be computed by a circuit of size $e_c(n^k)$.*

As Buhrman et al already established in [7] that there are oracles relative to which MA_{exp} has polynomial circuits, it is clear that Theorem 2 does not relativize.

It is interesting to compare the bounds for MA-TIME and ZP-TIME$^{\mathrm{NP}}$. The lower bounds we can prove for $\mathrm{MA}_{\mathrm{exp}}$ and $\mathrm{ZPEXP}^{\mathrm{NP}}$ are essentially the same; namely, half-exponential. They are also the same for time bounds bigger than exponential. But as soon as we consider time bounds smaller than exponential, the lower bound for MA-TIME becomes weaker than the one for ZP-TIME$^{\mathrm{NP}}$. In particular, addressing a question from the introduction, for any $k > 1$, we can prove that MA-TIME$[\exp_{\frac{1}{2}}]$ does not have circuits of size n^k, but we are unable to prove the same lower bound for MA-TIME$[\exp_\sigma]$ for any value of $\sigma < \frac{1}{2}$. In contrast, we know that $\mathrm{ZPP}^{\mathrm{NP}}$ does not have circuits of size n^k. On the hand, we see that for any $\epsilon > 0$, MA-TIME$[\exp_{\frac{1}{2}+\epsilon}]$ does not have polynomial sized circuits, thus improving the result of [7] stating that this is the case for $\mathrm{MA}_{\mathrm{exp}}$.

We are unable to improve these lower bounds without going to the complexity class $\Delta_3^{f(n)}$. In particular, we don't know if a super-half-exponential lower bounds can be proven for Σ_2^{exp}.

Next, we consider how well circuits can *approximate* members of the uniform classes. We say that a function f on input domain $\{0,1\}^n$ is approximated by a circuit C, if $C(x) = f(x)$ for at least a $\frac{1}{2} + \frac{1}{p(n)}$ fraction of the input domain[1], for some polynomial p. In particular we show the following.

Theorem 3 *For any rational value c, $0 < c \leq \frac{1}{2}$, and any k, there is a language $L_{c,k}$ in* MA-TIME$[\exp_{c+\frac{1}{2}}] \cap$ coMA-TIME$[\exp_{c+\frac{1}{2}}]$*, and, for any rational $\frac{1}{2} \leq c < 1$, and any k, there is a language $L_{c,k}$ in* MA-TIME$[\exp_{2c}] \cap$ coMA-TIME$[\exp_{2c}]$*, so that for infinitely many n, $L_{c,k} \cap \{0,1\}^n$ cannot be approximated by circuits of size $e_c(n^k)$.*

For proving this, we use the random-self reducibility properties of some classes of high complexity along with known (by now standard) techniques for increasing the hardness of functions. These techniques are heavily employed in the context of pseudorandom generator constructions.

The oracle constructed in [7] witnesses the fact that this theorem does not hold in all relativized world. However, by replacing MA-TIME$\cap$coMA-TIME with ZP-TIME$^{\mathrm{NP}}$, it is possible to prove a theorem that does hold relative to any oracle.

[1] The setting of the desired level of approximation to inverse polynomial is somewhat arbitrary; it avoids a third parameter besides time and circuit size in the theorem, thus improving readability.

Note that, except for the fact that the case $c = 0$ is not covered, Theorem 3 improves Theorem 2 by replacing "computed" with "approximated". In contrast, the ZP-TIME$^{\text{NP}}$-version of the theorem does not strictly improve Theorem 1, as the lower bounds for subexponential time bounds become worse. It would be interesting to remedy this, and, in particular, to show that ZPP$^{\text{NP}}$ cannot be approximated by linear sized circuits.

3 Kannan revisited

In [10], Kannan proves that $\Sigma_3^e \cap \Pi_3^e$ contains a function with maximum circuit complexity. The argument used in the proof actually gives that $\Sigma_3^f \cap \Pi_3^f$ contains functions of superpolynomial circuit complexity for any superpolynomial f.

In the paper, statement (*) of the Introduction, i.e., Theorem 4, page 48, is also claimed.

In order to claim (*), the following lemma is proved (Lemma 4, page 47).

Lemma *If $f(n)$ is any increasing time-constructible super-polynomial function, then there is a language L in $\Sigma_2^{f(n)} \cap \Pi_2^{f(n)}$ that does not have small circuits.*

By *small circuits* are meant circuits of size $O(n^k)$ for some fixed k (page 41). The proof of the lemma proceeds as follows: If SAT (the problem of deciding the satisfiability of boolean formulae) does not have polynomial circuits, then the lemma is true. In the case of SAT having polynomial circuits, by Karp and Lipton's theorem [11], the polynomial hierarchy collapses to $\Sigma_2^p \cap \Pi_2^p$. This implies that Σ_3^f can be simulated in $\Sigma_2^{f^{O(1)}}$, and since it was already established that $\Sigma_3^f \cap \Sigma_3^f$ contains functions of superpolynomial circuit complexity for any superpolynomial f, the lemma follows.

After proving the above lemma, statement (*) is claimed. But if our description above is accurate, it seems that what has actually been proven is: Either SAT does not have polynomial sized circuits or statement (*) is true. But this does not seem to imply (*), as the fact that SAT does not have polynomial sized circuits does not imply that it has exponential sized circuits.

In order to get a provable statement (†) of the same syntactic form as (*), it seems necessary to make up a more "balanced" "Either A or B" statement so that A as well as B implies (†).

To make up such as statement, we note that Karp and Lipton's technique actually gives the following lemma:

Lemma 4 *Let $f(n) \geq n$ be any time constructible functions. There is a constant c, so that if SAT has circuits of size $f(n)$ then Σ_3^n is included in $\Sigma_2^{f(n^c)^c}$.*

With this in mind, we now construct the statement (†):

(†) *For any time-constructible function $f(n) \leq 2^n$, there is a language in $\Sigma_2^{f(f(n)^c)^c} \cap \Pi_2^{f(f(n)^c)^c}$ that on infinitely many n does not have $f(n)^{\frac{1}{2}}$-size circuits.*

We prove the statement (†). Let $f(n)$ be given. Suppose SAT does not have circuits of size $f(n)$, then (†) follows. Otherwise SAT has circuits of size $f(n)$. Then, from Lemma 4, it follows that Σ_3^n is included in $\Sigma_2^{f(n^c)^c}$. By padding, we conclude that $\Sigma_3^{f(n)}$ is included in $\Sigma_2^{f(f(n)^c)^c}$. Then $\Sigma_3^{f(n)} \cap \Pi_3^{f(n)}$ is included in $\Sigma_2^{f(f(n)^c)^c} \cap \Pi_2^{f(f(n)^c)^c}$. But since $\Sigma_3^{f(n)} \cap \Pi_3^{f(n)}$ contains a function that does not have circuits of size $f(n)^{\frac{1}{2}}$, we are done.

In particular, the second level of the exponential hierarchy does not have circuit size $f(n)$ for any fixed half-exponential function f. The statement (†) can certainly be improved a bit by polynomial fiddling, but we don't see how to get any *essential* improvement. In particular, we don't see how to establish that the second level of the exponential hierarchy does not have σ-exponential circuits for any $\sigma > \frac{1}{2}$.

In the next two sections, the simple idea above is extended to ZP-TIME$^{\text{NP}}$ and MA-TIME, proving the first two theorems of the introduction.

Before we move on to proving the theorems stated in the introduction, we note that in fact Δ_3^e contains functions with maximum circuit complexity. We give a proof of this fact here since we shall need this result to prove Theorem 1. Let $M(n)$ denote the maximum possible circuit complexity of a function on n variables.

Lemma 5 *There is a language $L \in \Delta_3^e$ so that, for all n, the circuit complexity of $L \cap \{0,1\}^n$ is $M(n)$.*

Proof Let a *truth table* be a string over $\{0, 1\}$ of length 2^n for some n - such a string can be interpreted as the truth table of a Boolean function on n variables.

Let L_1 be the language consisting of tuples $\langle x, 1^{2^n}, 1^s \rangle$ so that x is a Boolean string of length less than 2^n that is the prefix of some truth table of length 2^n, so that the corresponding Boolean function cannot be computed by a circuit of size of s. Clearly, $L_1 \in \mathrm{NP}^{\mathrm{NP}}$; we guess the truth table and verify that no small circuit computes the same function. The procedure in figure 1 now generates on input n the lexicographically first truth table of a Boolean function on n variables with maximum circuit complexity. The Δ_3^e language L with maximum circuit complexity is $L = \{x | \mathrm{hard}(|x|)_{\mathrm{index}(x)} = 1\}$, where $\mathrm{index}(x)$ is the lexicographic index of x in $\{0,1\}^{2^n}$.

```
Procedure hard(int: n)
  $s := 2^n$;
  repeat until $\langle \lambda, 1^{2^n}, 1^s \rangle \in L_1$
    $s := s - 1$;
  $t := \lambda$;
  while $|t| < 2^n$ do
    if $\langle t0, 1^{2^n}, 1^s \rangle \in L_1$ then $t := t0$ else $t := t1$ endif
  return $t$
```

Figure 1: Generating the truth table of a hard function

■

Lemma 6 *For any time constructible function $n \le f(n) \le M(n)$, there is a language L_f in $\Delta_3^{f(n)^2}$ so that for all n, the circuit complexity of $L_f \cap \{0,1\}^n$ is at least $f(n)$.*

Proof By Shannon's theorem, the Boolean function on $g(n) = \min\{n, \lceil 2 \log f(n) \rceil\}$ variables with maximum circuit complexity has circuit complexity at least $f(n)$. Let L be the language of Lemma 5 and let $x \in L_f$ if and only if $x_{1\ldots g(n)} \in L$. ■

4 Zero error algorithms with an oracle for NP

In this section, we prove Theorem 1. We shall follow the line of proof of Köbler and Watanabe, showing that $\mathrm{ZPP}^{\mathrm{NP}}$ does not have linear circuits.

The difference of their proof to Kannan's sketched in the previous section, is in using an improved collapse of the polynomial hierarchy on the assumption of SAT having small circuits. We state a result from [6, 12] as a lemma, suitable for our application.

Lemma 7 ([6, 12]) *Let $f(n) \geq n$ be any time constructible function. Assume that SAT has circuits of size $f(n)$. Then, there is a randomized Turing machine with access to an* NP*-oracle that on input 1^n runs in expected time polynomial in $f(n)$, halts with probability 1, and outputs a circuit for SAT of size $f(n)$ when it does.*

Proof of Theorem 1. According to Lemma 6, there is a language L in $\Delta_3^{\exp_c}$ which does not have circuits of size $e_c(n^k)$. Let M be the machine accepting this language, running in time f for some $f \in \exp_c$, using an $\mathrm{NP}^{\mathrm{NP}}$-oracle. The $\mathrm{NP}^{\mathrm{NP}}$-oracle can be simulated by a polynomial time non-deterministic machine M_1 with an oracle for SAT. On an input x, the machine M queries M_1 which queries its SAT oracle. The longest query to the SAT oracle, when M is given a input of length n, has length at most some polynomial in f, say f^d. Now, $f^d \in \exp_c$. We can assume, without loss of generality, that all queries have length f^d.

Suppose SAT does not have circuits of size $e_c(n^k)$. Then we are done. So let us assume that SAT has circuits of size $e_c(n^k)$. Then we will show how to simulate the machine M (accepting L) by a $\text{ZP-TIME}^{\mathrm{NP}}[g]$ algorithm for some $g \in \exp_{2c}$. This simulation works in two steps.

The first step is to find a SAT circuit for instances of length f^d. By assumption, there is such a circuit of size at most h, for some $h \in \exp_{2c}$. Hence, according to Lemma 7, we can find the circuit by a $\text{ZP-TIME}^{\mathrm{NP}}[\exp_{2c}]$ computation.

Now, having found the circuit, the second step is to simulate the machine M using M_1 as oracle, but with M_1 using the circuit in place of its SAT-oracle (We can easily modify M and M_1 so that the circuit is given as input). Since a circuit of size h has already been found, this simulation can be done in $\text{TIME}^{\mathrm{NP}}[\exp_{2c}]$. The two steps together form a $\text{ZP-TIME}^{\mathrm{NP}}[\exp_{2c}]$ computation.

Furthermore, it is easy to verify that the above proof and also Lemma 7 relativises. Hence we have the Theorem.

5 Merlin-Arthur games

In this section, we prove Theorem 2. In [7], for showing that $\mathrm{MA}_{\mathrm{exp}}$ contains languages without polynomial size circuits, the authors make use of a theorem due to Babai et al [4] which states; "EXP $\subseteq$ P/poly $\Rightarrow$ EXP $=$ MA". We follow the same line of argument as in [7], but we need the following refinement of the result of Babai et al [4], stated with a more general range of time and size bounds. The result essentially follows from a theorem in [2] on *transparent* proofs.

Lemma 8 *Let $g(n) \geq n$ and $s(n) \geq n$ are increasing time constructible functions. Then there is a constant $c > 1$ so that the following holds.*
If $g(n) \leq 2^n$ then

$$\mathrm{TIME}[g(n)^c] \subseteq \mathrm{SIZE}[s(n)] \Rightarrow \mathrm{TIME}[g(n)] \subseteq \text{MA-TIME}[s(3n)^c].$$

If $g(n) > 2^n$ then

$$\mathrm{TIME}[2^{cn}] \subseteq \mathrm{SIZE}[s(n)] \Rightarrow \mathrm{TIME}[g(n)] \subseteq \text{MA-TIME}[s(3\log g(n))^c].$$

Proof Given a Turing machine T operating in $g(n)$ steps, we can construct a Turing transducer T', operating in $g(n)^c$ steps which on input x outputs a *transparent* proof [2] of the fact that T accepts (or rejects) on input x. Also, we can make a Turing machine T'' operating in $g(|x|)^c$ steps which on input $\langle x, i\rangle$, outputs the i'th bit of the output string of T' on input x. We now construct an Arthur-Merlin game accepting the same language as T, as follows: Merlin sends Arthur the description of a circuit of size $s(n)$ computing the same function of as T''. By the assumption, such a circuit exists. Note that this circuit is a succinct representation of the transparent proof for the computation. Arthur, following the protocol in [2], now verifies that the circuit indeed is a succinct encoding of a string close to a transparent proof of the correct computation.

The statement of the theorem now follows; note that the two cases of the theorem corresponds to which part of the input $\langle x, i\rangle$ is the larger. The factor 3, occurring twice in the statement of the theorem, is an upper bound on the ratio of the length of $\langle x, y\rangle$ and the length of x or y for a reasonable pairing function $\langle x, y\rangle$. ■

Proof of Theorem 2. We divide the proof into two cases, according to whether $c \leq \frac{1}{2}$ or $c > \frac{1}{2}$.
Case 1 $c \leq \frac{1}{2}$: We need to show that $\text{MA-TIME}[\exp_{c+\frac{1}{2}}]$ does not have circuits of size $e_c(n^k)$. Clearly, we can assume that $\mathrm{TIME}[\exp_{c+\frac{1}{2}}]$ *does* have

circuits of size $e_c(n^k)$, otherwise we are done. But then according to Lemma 8, TIME[$\exp_{c+\frac{1}{2}}$] is included in MA-TIME[$\exp_c$]. By padding, TIME[$\exp_{c+1}$] is included in MA-TIME[$\exp_{c+\frac{1}{2}}$]. But TIME[$\exp_{c+1}$] contains a language that cannot be computed by circuits of size $e_c(n^k)$, and we are done.
Case 2 $c > \frac{1}{2}$: We need to show that MA-TIME[$\exp_{2c}$] does not have circuits of size $e_c(n^k)$. We can again assume that TIME[$\exp_{2c}$] *does* have circuits of size $e_c(n^k)$. In particular, this is the case for EXP, and by Lemma 8, we conclude that TIME[$\exp_{2c}$] is included in MA-TIME[$\exp_{c-1+2c}$]. By padding, TIME[$\exp_{2c+(1-c)}$] is included in MA-TIME[$\exp_{c-1+2c+(1-c)}$] which is the class MA-TIME[$\exp_{2c}$]. But TIME[$\exp_{2c+(1-c)}$] $=$ TIME[$\exp_{c+1}$] contains a language that cannot be computed by circuits of size $e_c(n^k)$ and we are done.

6 Non-Approximability

In this section, we show Theorem 3 of the introduction. We need a result from [4]. We state it as a lemma in a form convenient for our application.

Lemma 9 [4] *There is a constant $\epsilon > 0$ so that the following holds. There is a deterministic quasi-polynomial time procedure* harden*, taking as input the truth table of a Boolean function f on n variables, and outputting the truth table of a Boolean function* harden(f) *on n^2 variables, with the following property. Let $s > n^{1/\epsilon}$. If f cannot be computed by circuits of size s, then any circuit of size s^ϵ taking n^2 inputs, will agree with* harden(f) *on at most a $\frac{1}{2} + s^{-\epsilon}$ fraction of the input domain $\{0,1\}^{n^2}$.*

Proof of Theorem 3. We divide the proof into two cases.
Case 1. $c \geq \frac{1}{2}$: According to Theorem 2, there is a language L in MA-TIME[$\exp_{2c}$] $\cap$ coMA-TIME[$\exp_{2c}$] that cannot be computed by circuits of size $(e_c(n^{2k}))^{1/\epsilon}$. We now define a new language fulfilling the requirements of the theorem. As the language only needs to satisfy the desired property on infinitely many input lengths, we shall only consider input lengths of the form n^2. The following computation defines the language. Since $2c \geq 1$, we can, on input x of length n^2, compute the truth table t of the characteristic function of $L \cap \{0,1\}^n$, without leaving the complexity class MA-TIME[$\exp_{2c}$] $\cap$ coMA-TIME[$\exp_{2c}$], as this class is closed under exponential time Turing reductions. Now, compute the truth table t' =harden(t),

and look up x in t'. This is the result of the computation. The language so defined full-fills the requirements of the theorem and we are done.
Case 2. $c < \frac{1}{2}$: Let $c = \frac{1}{2} - \delta$. We already know that $\mathrm{MA}_{\mathrm{exp}} \cap \mathrm{coMA}_{\mathrm{exp}}$ contains a language that cannot be approximated by circuits of size $e_{\frac{1}{2}}(n^k)$. More than that, from the proof we see that we can ensure that any circuit of this size will actually err on at least a $\frac{1}{2} - e_{1/2}(n)^{-1}$ fraction of the input domain for infinitely many n. By padding, MA-TIME$[\exp_{1-\delta}] \cap$ coMA-TIME$[\exp_{1-\delta}]$ contains a language, so that any circuit of size $e_{\frac{1}{2}-\delta}(n^k)$ has to err on at least a $\frac{1}{2} - e_{1/2}(e_{-\delta}(n))^{-1} = \frac{1}{2} - e_c(n)^{-1}$ fraction of the input domain. As $c > 0$, this means that no circuit of size $e_{\frac{1}{2}-\delta}(n^k)$ approximates the language, and we are done.

References

[1] V. Arvind and J. Köbler. On resource-bounded measure and pseudorandomness. Proceedings of the *17th conference on the Foundations of Software Technology and Theoretical Computer Science*, LNCS Vol. 1346, (1997), pp. 235–249.

[2] L. Babai, L. Fortnow, L. Levin and M. Szegedy. Checking computations in polylogarithmic time. Procedings of the *23rd ACM Symposium on the Theory of Computation*, (1991), pp. 21–31.

[3] L. Babai, L. Fortnow, and C. Lund. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity*, 1, (1990), pp. 3–40.

[4] L. Babai, L. Fortnow, N. Nisan and A. Wigdersen. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3, (1993), pp. 307–318.

[5] J. L. Balcázar, J. Díaz and J. Gabarró. *Structural Complexity – I & II.* Springer Verlag, Berlin Heidelberg, 1988.

[6] N.H. Bshouty, R. Cleve, R. Gavalda, S. Kannan, and C. Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, 52, (1996), pp. 421-433.

[7] H. Buhrman, L. Fortnow and T. Thierauf. Nonrelativizing separations. Proceedings of the *13th IEEE conference on Computational Complexity*, (1998), pp. 8–12.

[8] O. Goldreich and D. Zuckerman. Another proof that BPP $\subseteq$ PH (and more). *ECCC TR97-045*, (1997). Available at `http://www.eccc.uni-trier.de/eccc/`.

[9] G.H. Hardy. *Orders of Infinity.* Cambridge Tracts in Mathematics and Mathematical Physics, No. 12. Second edition. Cambridge University Press, Cambridge, 1924.

[10] R. Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55, (1982), pp. 40–56.

[11] R. M. Karp and R. J. Lipton. Some connections between nonuniform and uniform complexity classes. Proceedings of the *12th ACM Symposium on Theory of Computing*, (1980), pp. 302–309.

[12] J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. Proceedings of the *International Colloquium on Automata, Languages and Programming*, LNCS Vol. 944, (1995), pp. 196–207.

[13] K.W. Morris and G. Szekeres. Tables of the logarithm of iteration of $e^x - 1$. *J. Australian Math. Soc.* 2, (1962), pp. 321-327.

[14] C. Papadimitriou. *Computational Complexity.* Addison-Wesley Publishing Company, 1994.

[15] P. B. Miltersen, N. V. Vinodchandran and O. Watanabe. Super-polynomial versus half-exponential circuit size in the exponential hierarchy. *Research Report c-130*, Dept. of Math. and Comput. Sc., Tokyo Inst. of Tech. Available at `http://www.is.titech.ac.jp/research/research-report/C/`, 1999.

[16] G. Szekeres. Fractional iteration of exponentially growing functions. *J. Australian Math. Soc.* 2, (1962), 301-320.

Recent BRICS Report Series Publications

RS-99-46 Peter Bro Miltersen, Vinodchandran N. Variyam, and Osamu Watanabe. *Super-Polynomial Versus Half-Exponential Circuit Size in the Exponential Hierarchy*. December 1999. 14 pp. Appears in Asano, Imai, Lee, Nakano and Tokuyama, editors, *Computing and Combinatorics: 5th Annual International Conference*, COCOON '99 Proceedings, LNCS 1627, 1999, pages 210–220.

RS-99-45 Torben Amtoft. *Partial Evaluation for Designing Efficient Algorithms—A Case Study*. December 1999.

RS-99-44 Uwe Nestmann, Hans Hüttel, Josva Kleist, and Massimo Merro. *Aliasing Models for Mobile Objects*. December 1999. ii+46 pp. To appear in a special FOOL6 issue of *Information and Computation*. An extended abstract of this revision, entitled *Aliasing Models for Object Migration*, appeared as Distinguished Paper in Amestoy, Berger, Daydé, Duff, Frayssé, Giraud and Daniel, editors, *5th International Euro-Par Conference*, EURO-PAR '99 Proceedings, LNCS 1685, 1999, pages 1353–1368, which in turn is a revised part of another paper called *Migration = Cloning ; Aliasing* that appeared in Cardelli, editor, *Foundations of Object-Oriented: 6th International Conference*, FOOL6 Informal Proceedings, 1999 and as such supersedes the corresponding part of the earlier BRICS report RS-98-33.

RS-99-43 Uwe Nestmann. *What is a 'Good' Encoding of Guarded Choice?* December 1999. ii+34 pp. To appear in a special EXPRESS '97 issue of *Information and Computation*. This revised report supersedes the earlier BRICS report RS-97-45.

RS-99-42 Uwe Nestmann and Benjamin C. Pierce. *Decoding Choice Encodings*. December 1999. ii+62 pp. To appear in *Journal of Information and Computation*. An extended abstract appeared in Montanari and Sassone, editors, *Concurrency Theory: 7th International Conference*, CONCUR '96 Proceedings, LNCS 1119, 1996, pages 179–194.