Electronic Colloquium on Computational Complexity, Report No. 131 (2006)

ECCC

On Heuristic Time Hierarchies

Preliminary Version

Konstantin Pervyshev *
University of California, San Diego
pervyshev@cs.ucsd.edu

Abstract

We study the existence of time hierarchies for heuristic algorithms. We prove that a time hierarchy exists for heuristics algorithms in such syntactic classes as **NP** and **coNP**, and also in semantic classes **AM** and **MA**. Earlier, Fortnow and Santhanam (FOCS'04) proved the existence of a time hierarchy for heuristics algorithms in **BPP**. We present an alternative approach and give a simpler proof.

1 Introduction

There is an interesting phenomenon related to the creation of algorithms. Once having developed a solution for some problem, one may find more and more efficient solutions for the same problem later. Everybody can witness that such things happen from time to time. But is it always the case that one may find better and better solutions for the same problem?

Complexity theory answers this question in the following rigorous way. For any constants c and d such that $1 \leq c < d$, there exists a language that can be recognized by some deterministic algorithm in time $O(n^d)$ but can be recognized by no deterministic algorithm in time $O(n^c)$. This means that there are problems having certain solutions, such that nobody can come up with much more efficient solutions for these problems.

The above result, called a deterministic time hierarchy, was proven by Hennie and Stearns in 1960s [HS66]. One decade later, it was shown that a time hierarchy also exists for non-deterministic algorithms [Coo73, SFM78, Žák83]. More of that, the developed techniques allow to prove the existence of a time hierarchy for virtually any syntactic model of computations. But it still remains unknown whether any purely semantic model of computations has a time hierarchy.[1]

Semantic models are opposed to syntactic ones in that the former require that every machine that belongs to a model satisfies some promise (like a promise of bounded error). Moreover, it is not possible to computationally recognize whether a given machine satisfies such a promise. While deterministic and non-deterministic algorithms are the examples of syntactic models, various kinds of probabilistic algorithms constitute semantic models. In view of the importance of probabilistic algorithms, it is highly astonishing that they are not known to have a time hierarchy.

*The most part of this work was done while the author was at St. Petersburg State University, Russia.

[1] A time hierarchy is known to exist for **IP**, a semantic model. However, the computational power of **IP** is equal to that of **PSPACE**, which is a syntactic model. In this sense, **IP** is not a purely semantic model.

ISSN 1433-8092

Notwithstanding, Fortnow and Santhanam [FS04] recently came up with a result on a time hierarchy for heuristic probabilistic algorithms. We say that some machine (or algorithm) is a $\delta(n)$-heuristic for some language L, if this machine recognizes L on a fraction $\delta(n)$ of inputs of every length. On other inputs, it may perform arbitrary, in particular, violate a promise of some semantic model. Fortnow and Santhanam proved that a time hierarchy exists for $(1-1/n^a)$-heuristic algorithms in **BPP**, i.e. $(1-1/n^a)$-heuristic probabilistic algorithms with the promise of bounded two-sided error. This result raises a number of open questions in regard to time hierarchies in both semantic and syntactic models.

Our paper makes progress in studying heuristic time hierarchies and answers the two following questions. First, as virtually any syntactic model has a time hierarchy for "traditional" algorithms, does every syntactic model have a time hierarchy for heuristic algorithms? Second, as there is a time hierarchy for heuristic algorithms in **BPP**, are there time hierarchies for heuristic algorithms in other semantic models?

1.1 Our Results

On the side of syntactic models, we prove that a time hierarchy exists for heuristic non-deterministic algorithms. Formally, we show that there exists a language that cannot be recognized by $(1/2+1/n^a)$-heuristic non-deterministic algorithms in time $O(n^c)$ but is recognizable by "traditional" non-deterministic algorithms in some polynomial time[2] :

Theorem. *For any positive constants a and c,*

$$\mathbf{NP} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{NTIME}[n^c].$$

We would like to stress that the above result doesn't follow immediately neither from a time hierarchy for "traditional" non-deterministic algorithms, nor from the techniques used for proving it.

Our approach works not only for heuristic algorithms in **NP**, but also for heuristic algorithms in $\mathbf{\Sigma_k}$ and $\mathbf{\Pi_k}$. Generally, we can show a time hierarchy for heuristic algorithms in any syntactic model that is efficiently closed under taking majority answer of polynomially many machines.

On the side of semantic models, we prove that a time hierarchy exists for heuristic Arthur-Merlin and Merlin-Arthur games:

Theorem. *For any positive constants a and c,*

$$\mathbf{heur}_{1-1/n^a}\mathbf{AM} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{AMTime}[n^c]$$
$$\mathbf{heur}_{1-1/n^a}\mathbf{MA} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{MATime}[n^c].$$

Further, we give an alternative proof of a time hierarchy for heuristic probabilistic algorithms with the promise of bounded two-sided error:

Theorem. *For any positive constants a and c,*

$$\mathbf{heur}_{1-1/n^a}\mathbf{BPP} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{BPTime}[n^c]. \quad (1)$$

Previously, Fortnow and Santhanam [FS04] proved that for some constant a and for any $b \geq a$,

$$\mathbf{heur}_{1-1/n^b}\mathbf{BPP} \not\subseteq \mathbf{heur}_{1-1/n^a}\mathbf{BPTime}[n^c]. \quad (2)$$

Rahul Santhanam pointed out to us that using "slightly" non-uniform amplification by Impagliazzo, Jaiswal and Kabanets [IJK06] and proving (2) in "slightly" non-uniform setting, one can

[2] One may try to replace "some" polynomial time with $O(n^d)$-time, where $d > c$.

obtain $(1/2 + 1/n^a)$-heuristic algorithms on the right side of (2), therefore matching the result with (1). Thus, a time hierarchy with $(1/2 + 1/n^a)$-heuristic algorithms on the side of the less time seems to be the "right" time hierarchy for heuristic **BPP**.

In regard to heuristic **BPP**, the strength of our approach is that it gives the "right" time hierarchy in a direct way. Also, we think that our approach is simpler than that of [FS04]. In particular, we don't make any use of optimal algorithms and complete self-reducible languages for polynomial space or exponential time. Our belief is that every time hierarchy should have a simple proof as what we all expect from computational models is that the time resource is easy to spend.

1.2 Related Work

The study of time hierarchies for heuristic algorithms is not the only line of research on time hierarchies for semantic models. Recently, there was a number of result on time hierarchies for "slightly" non-uniform algorithms. In [Bar02], Barak proved a time hierarchy for algorithms in **BPP** that are provided with non-uniform advice of length roughly $\log \log n$. The amount of advice that suffices for the existence of the time hierarchy was reduced by Fortnow and Santhanam [FS04], who proved a time hierarchy for algorithms in **BPP** that receive only one bit of advice (see also [GST04]). Further, Fortnow, Santhanam and Trevisan gave a proof of a time hierarchy for **RP** with one bit of advice [FST05]. Finally, van Melkebeek and Pervyshev proved the existence of a time hierarchy for any reasonable semantic model of computations with one bit of advice [vMP06].

For more details on time hierarchies in semantic models, reader is referred to a recent survey by Fortnow and Santhanam [FS06].

Related to time hierarchies for heuristic algorithms is the notion of random languages. A language L is $\epsilon(n)$-random for some family of algorithms if every algorithm from the family recognizes L on a fraction at least $1/2 - \epsilon(n)$ and at most $1/2 + \epsilon(n)$ of inputs of every length n. In [Wil83], Wilber proved that for any constants c, d and a, such that $1 \leq c < d$, there is a language in $\mathbf{DTIME}[n^d]$ that is $1/n^a$-random for $\mathbf{DTIME}[n^c]$ (see also [GW00]). This implies a time hierarchy for deterministic heuristic algorithms:

$$\mathbf{DTIME}[n^d] \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{DTIME}[n^c].$$

The latter, being weaker than the result of Wilber on random languages, can be proved by a straightforward diagonalization. Still, we are not aware of any results on random languages for any computational model other than deterministic algorithms.

2 Our Technique

Somewhat surprisingly, delayed diagonalization appears to be extremely useful for proving time hierarchies for heuristic algorithms. It helps not only with heuristic algorithms in **NP**, which is not known to be closed under complementation, but also in **BPP**. However, what we use is not a traditional delayed diagonalization. Our diagonalization has a "statistical" flavor.

Non-deterministic case. Let us explain our method on the example of heuristic **NP**. To prove a time hierarchy, we construct a polynomial-time non-deterministic machine N that disagrees with any $O(n^c)$-time non-deterministic machine M_i on a fraction at least $1/2 - 1/n^a$ of inputs of some length.

Let n_i be some input length and let $n_i^* = 2^{n_i^{c+1}}$. We have $c+1$ on the top of $2^{n_i^{c+1}}$ in order to diagonalize against machines that work in time $O(n^c)$, where constant hidden in O-notation may be arbitrary.

On an input x of length n, machine N performs as follows:

$$N(x) = \begin{cases} \text{majority}_{y\in\{0,1\}^{n+1}}\{M_i(y)\} & \text{if } n_i \le n < n_i^* \\ \neg\,\text{majority}_{y\in\{0,1\}^{n_i}}\{M_i(y)\} & \text{if } n = n_i^* \end{cases} \tag{3}$$

Let v be the most popular answer of machine M_i on inputs of length n_i :

$$v = \underset{y\in\{0,1\}^{n_i}}{\text{majority}}\{M_i(y)\}. \tag{4}$$

Let L_N be the language recognized by N :

$$L_N = \{x : N(x) = 1\}. \tag{5}$$

Assume that M_i recognizes L_N on a fraction $1/2 + 1/n^a$ of inputs of every length, i.e. M_i is a $(1/2 + 1/n^a)$-heuristic for L_N. According to (3), N returns $\neg v$ on every input of length n_i^*. This implies, by our assumption, that M_i returns $\neg v$ on more than a half of inputs of length n_i^*. But then, N outputs $\neg v$ on every input of length $n_i^* - 1$. We can continue reasoning in this way and obtain that M_i returns $\neg v$ on more than a half of inputs of length n_i. This contradicts to (4), hence M_i is not a $(1/2 + \epsilon)$-heuristic for L_N.

We are almost done with our time hierarchy. It remains to make sure that N can perform as (3) in polynomial time. Unfortunately, it is not feasible to compute the majority answer of M_i over inputs of length $n + 1$ in time that is just polynomial in n.

We overcome this difficulty by making an observation that it is still possible to obtain a contradiction even if we allow machine N to fail on some small fraction of inputs of length n to compute the most popular answer of M_i at length $n + 1$. This "heuristic" computation of majority, which is good on all but a small fraction of inputs, can be done in polynomial time using expander graphs.

Here, we would like to suggest a way of looking at our delayed diagonalization. Informally, what actually is constructed in the delayed diagonalization is a "channel" that allows N to transfer some value (think of $\neg v$) from input length n_i^* to input length n_i. In our heuristic setting, this channel is prone to errors. The assumption that a $(1/2 + 1/n^a)$-heuristic M_i recognizes L_N implies that M_i may disagree with N on a fraction $1/2 - 1/n^a$ of inputs of every length n. Therefore every segment of the "channel" that connects input lengths n and $n + 1$ may introduce an error with probability up to $1/2 - 1/n^a$. Then the computation of majority answer of M_i at length $n + 1$ allows N to reconstruct the one-bit message $\neg v$ being transferred over the "channel".

Probabilistic case. The reason for delayed diagonalization works for **BPP** is more subtle. As **BPP** is closed under complementation, it seems we may use a straightforward diagonalization: define N so that on an input x it simulates M_i and returns $\neg M_i(x)$. However, this fails since M_i may violate the promise of bounded two-sided error on a fraction more than $1/2-1/n^a$ of inputs, thus forcing N to violate the promise on the same set of inputs. As the straightforward solution doesn't work, let us try our "statistical" diagonalization.

First of all, note that, in the semantic setting, it is not clear what the most "popular" answer of M_i at some length is. A reason for that is that the answer of M_i may be undefined on a fraction up to $1/2-1/n^a$ of inputs of length n. Nonetheless, the assumption that M_i significantly agrees with machine N, which we will construct, will imply that M_i either answers 1 on a clear majority of inputs, or answers 0, also on a clear majority (recall the non-deterministic setting).

Then, a good news is that, in probabilistic setting, machine N can compute the most "popular" answer of M_i at length $n+1$ pretty simple: N may just simulate M_i, amplified for some polynomial number of times, on several randomly selected inputs of length $n+1$. This will work fine under assumption that M_i significantly agrees with N. Still, one more step is to be taken in order to ensure that N satisfies the promise of bounded two-sided error on most of inputs in case M_i doesn't agree with N.

In order to do that, we replace the simulation of amplified machine M_i on a randomly chosen input x of length $n+1$ with a more complicated process. A somewhat similar trick may be traced in [FS04]. Let π be the probability over randomly chosen input y of length $n+1$ and over randomness of amplified machine M_i that the latter accepts y, i.e.

$$\pi = \mathop{\mathbf{Pr}}_{y\in\{0,1\}^{n+1}}[\text{amplified } M_i(y) = 1], \tag{6}$$

where probability is taken over inputs y of length $n+1$ and over internal randomness of machine M_i. Machine N, on an input x of length n, can obtain an estimation $\hat{\pi}$ of this probability and compare it to some threshold θ_x, which depends on the input given to N. If the obtained estimation is greater than the threshold, then machine N returns 1; otherwise, N outputs 0.

Our thresholds θ_x, $x \in \{0,1\}^n$, are chosen so that they are uniformly distributed on some interval $[1/2 - 1/p_1(n), 1/2 + 1/p_1(n)]$ for some polynomial $p_1(n)$. One may see that, since the thresholds are close to $1/2$, machine N indeed computes the most "popular" answer of machine M_i. What is more, machine N satisfies the promise of bounded two-sided error on all but a small fraction of inputs whatever machine M_i. To see this, we note that for most of inputs x, it unavoidably holds that $|\theta_x - \pi| > 1/p_2(n)$ for some polynomial $p_2(n)$. On such inputs, by Chernoff bound, machine N either obtains $\hat{\pi} > \theta_x$ (and returns 1) with probability $1 - e^{-\Theta(n)}$, or obtains $\hat{\pi} < \theta_x$ (and returns 0), also with probability $1 - e^{-\Theta(n)}$. Therefore N satisfies the promise of bounded two-sided error on these inputs.

We would like to note that, in some sense, this idea is a "randomization" of that for non-deterministic case.

A detailed implementation of our ideas is given in the rest of this paper. We start with a time hierarchy for heuristic algorithms in **NP** (Section 3). Then, "randomize" this proof and obtain a time hierarchy for heuristic algorithms in **BPP** (Section 4). After that, we prove a time hierarchy for heuristic algorithms in **MA** (Section 5). In some sense, this proof combines the proofs for heuristic **NP** and **BPP**. Finally, we sketch a proof of a time hierarchy for heuristic **AM** (Section 6).

3 Time Hierarchy for Heuristic Algorithms in NP

For a non-deterministic machine M, let $V_M(x)$ denote its answer on an input x. In contrast to semantic models, this value is defined on every input x. Also, we use notation $M(x,w)$ for the result of execution of M on an input x provided a witness w.

Definition 1. *A language L belongs to a class $\mathbf{heur}_{\delta(n)}\mathbf{NTIME}[n^c]$ if there exists a non-deterministic machine M such that for any input length n, it runs in time $O(n^c)$, and*

$$|\{x : V_M(x) = L(x)\}| \geq \delta(n) \cdot 2^n. \tag{7}$$

We say that M solves L on a fraction at least $\delta(n)$ of inputs of every length n. One may also define a class $\mathbf{heur}_{\delta(n)}\mathbf{NP}$.

In our proofs of time hierarchies for heuristic algorithms, it is crucial that machine N is able to compute majority answer of a machine M_i over all inputs of some length. In case of **BPP**

and **MA**, this can be done in polynomial time using the power of randomization. But in case of **NP** and **AM**, we have to derandomize the computation of majority using expanders (more specifically, mixers):

Definition 2. *We say that a d-regular on the left bipartite graph $G = ([N_1], [N_2], E)$ is an ϵ-mixer if for every subset B of vertices on the right such that $|B| \leq (1/2 - \epsilon)N_2$, there are at most ϵN_1 vertices v on the left such that $|\Gamma(v) \cap B| \geq d/2$.*

Definition 3. *We say that a family of $d(n)$-regular on the left $\epsilon(n)$-mixers G_n is explicit if for any vertex v on the left of G_n, $\Gamma(v)$ is computable in time polynomial in n. In particular, $d(n)$ is no more than polynomial in n.*

These mixers can be obtained from expander graphs (for example, from Margulis construction [Mar73, GG81]):

Lemma 4. *For any positive constant a, there exists an explicit family of regular on the left $1/(2(n+1)^a)$-mixers $G_n = ([2^n], [2^{n+1}], E_n)$.*

Now, we can proceed to our time hierarchy theorem for heuristic algorithms in **NP**:

Theorem 5. *For any positive constants a and c,*

$$\mathbf{NP} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{NTIME}[n^c].$$

Assume $\{M_i\}$ is an efficient enumeration of nondeterministic machines. Every machine in this enumeration is restricted to run for no more than m^{c+1} steps on inputs of length m and can be simulated with at most polynomial overhead. Furthermore, every machine appears infinitely often in this enumeration. The latter and $c+1$ on the top of m^{c+1} allow us to diagonalize against $O(n^c)$-time machines, where constant hidden in O-notation may vary for different machines M_i.

To prove our theorem, we construct a non-deterministic machine N that disagrees with any machine M_i on a fraction more than $1/2 - 1/n^a$ of inputs of some length. Essentially, our proof is a delayed diagonalization. For each machine M_i, we reserve a set of input lengths $S_i = [n_i : n_i^*]$, where $n_i^* = 2^{(n_i)^{c+1}}$ and $n_{i+1} = n_i^* + 1$. We choose n_1 big enough so that all the inequalities in this proof hold for every $n \geq n_1$.

Given an input x of length n, non-deterministic machine N performs as follows:

find i such that $n \in S_i$
if $n = n_i^*$ **then**
  for every $y \in \{0,1\}^{n_i}$
    deterministically compute $V_{M_i}(y)$
  return $\neg\,\text{majority}_{y\in\{0,1\}^{n_i}}\{V_{M_i}(y)\}$
else
  interpret x as a vertex on the left of G_n
  for every $y \in \Gamma(x) \subseteq \{0,1\}^{n+1}$
    guess a witness w_y and simulate $M_i(y, w_y)$
  return $\text{majority}_{y\in\Gamma(x)}\{M_i(y, w_y)\}$

Here and later, majority operator in case of a tie between values 0 and 1 gives preference to 1. Note that for $n \neq n_i^*$, we have $V_N(x) = \text{majority}_{y\in\Gamma(x)}\{V_{M_i}(y)\}$.

Let a language L_N consist of those strings x that are accepted by N, i.e.

$$L_N = \{x : V_N(x) = 1\}. \tag{8}$$

The following two lemmas establish our theorem.

Lemma 6.

$$L_N \in \mathbf{NP}.$$

PROOF: Recall that the size of any set of neighbors $\Gamma(x)$ in graph G_n is polynomial in n. Also, note that N can find index i such that $n \in S_i$ in polynomial time. This can be done simply by computing numbers n_1, n_2, n_3, ... one by one. Then N is apparently a polynomial-time non-deterministic machine. □

Lemma 7.

$$L_N \notin \mathbf{heur}_{1/2+1/n^a}\mathbf{NTIME}[n^c]$$

PROOF: This proof goes by contradiction. Assume that the statement of our lemma is false, and, consequently, there exists a non-deterministic machine M_i that for any input length n, agrees with machine N on a fraction at least a $1/2 + 1/n^a$ of inputs of that length.

Consider a set S_i that corresponds to machine M_i. Let v be the most "popular" answer of machine M_i at the first input length in this set:

$$v = \operatorname*{majority}_{x\in\{0,1\}^{n_i}}\{V_{M_i}(x)\} \tag{9}$$

We want to show that our assumption that M_i agrees with N on a fraction at least $1/2 + 1/n^a$ of inputs of every input length leads to a "statistical" contradiction. Let

$$A_{M_i}^{(n)} = \{x \in \{0,1\}^n : V_{M_i}(x) = v\} \tag{10}$$

and

$$A_N^{(n)} = \{x \in \{0,1\}^n : V_N(x) = v\} \tag{11}$$

By (9), it holds that $|A_{M_i}^{(n_i)}| \geq \frac{1}{2} \cdot 2^{n_i}$. However, under our assumption, the following claim is true, therefore we obtain a desirable contradiction.

Claim 8. *Under our assumption, for every $n \in S_i$,*

$$|A_N^{(n)}| < \frac{1}{2n^a}2^n \quad \textit{and} \quad |A_{M_i}^{(n)}| < \Big(\frac{1}{2} - \frac{1}{2n^a}\Big)2^n. \tag{12}$$

Note that the bound on the size of $A_{M_i}^{(n)}$ follows from our assumption (which implies that M_i disagrees with N on a fraction less than $1/2 - 1/n^a$ of inputs of length n) and from the bound on the size of $A_N^{(n)}$. The latter bound can be proved by induction on n. For input length $n = n_i^*$, due to the deterministic simulation, we have $A_N^{(n)} = \emptyset$. Now, let n be any input length less than n_i^* in our set S_i. One may see that

$$A_N^{(n)} \subseteq \{x \in \{0,1\}^n : |\Gamma(x) \cap A_{M_i}^{(n+1)}| \geq d/2\}, \tag{13}$$

where d is the regularity of G_n on the left (for $v = 1$, these two sets coincide as majority operator returns 1 in case of a tie). As G_n is a $1/(2(n+1)^a)$-mixer, the upper bound on the size of $A_N^{(n)}$ follows:

$$\frac{|A_N^{(n)}|}{2^n} \leq \frac{1}{2(n+1)^a} < \frac{1}{2n^a}. \tag{14}$$

Hence our claim follows. □

4 Time Hierarchy for Heuristic Algorithms in BPP

Definition 9. *Assume a machine M. Then*

$$V_M(x) = \begin{cases} 1 & \text{if } \mathbf{Pr}[M(x)=1] > 2/3 \\ 0 & \text{if } \mathbf{Pr}[M(x)=0] > 2/3 \\ \perp & \text{otherwise} \end{cases} \tag{15}$$

We say that M has a bounded two-sided error on input x if $V_M(x) \in \{0,1\}$.

Definition 10. *A language L belongs to a class $\mathbf{heur}_{\delta(n)}\mathbf{BPTime}[n^c]$ if there exists a machine $M(x,r)$ such that for any input length n, it runs in time $O(n^c)$, and*

$$|\{x : V_M(x) = L(x)\}| \geq \delta(n) \cdot 2^n. \tag{16}$$

We say that M solves L on a fraction at least $\delta(n)$ of inputs of every length n. One may also define a class $\mathbf{heur}_{\delta(n)}\mathbf{BPP}$ in a straightforward way.

Theorem 11. *For any positive constants a and c,*

$$\mathbf{heur}_{1-1/n^a}\mathbf{BPP} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{BPTime}[n^c].$$

The proof of this theorem may be seen as a "randomization" of the proof of a time hierarchy for heuristic **MA**.

Assume $\{M_i\}$ is an efficient enumeration of probabilistic machines. Every machine in this enumeration is restricted to run for no more than m^{c+1} steps on inputs of length m and can be efficiently simulated. Furthermore, every machine appears infinitely often in the enumeration.

The two procedures below help to estimate the majority answer of machine M_i over inputs of length $n+1$:

```
Estimate-Over-Inputs (M_i, 1^{n+1})
  for (n+1)^{4a+1} times
    choose x ∈ {0,1}^{n+1} uniformly at random
    run Amplify-Over-Randomness (M_i, x)
  return π̂ = the fraction of positive answers

Amplify-Over-Randomness (M_i, x) /* |x| = n+1 */
  for n+1 times
    simulate M_i(x) with new randomness
  return majority answer
```

Let

$$\pi = \mathop{\mathbf{Pr}}_{x\in\{0,1\}^{n+1}} [\text{ Amplify-Over-Randomness } (M_i, x) = 1], \tag{17}$$

Very roughly speaking, we have $\pi \approx \mathbf{Pr}_{x\in\{0,1\}^{n+1}}[V_{M_i}(x) = 1]$. However, one should remember the following: 1) for some inputs x, it may hold that $V_{M_i}(x)$ is undefined, i.e. $V_{M_i}(x) = \perp$, and 2) even if $V_{M_i}(x) = v$, machine M_i may output $\neg v$ with some probability (which is less than $1/3$).

Proposition 12 (Chernoff-Hoefding)**.** *Given $X_1, X_2, \ldots, X_n$ identically and independently distributed, such that $X_i \in [0,1]$ and $\mathbb{E}[X_i] = \mu$, then*

$$\mathbf{Pr}[|\frac{\sum_{i=1}^n X_i}{n} - \mu| \geq \epsilon] \leq e^{-\frac{\epsilon^2 n}{2}}.$$

Chernoff bound guarantees that $\hat{\pi}$ estimates π good enough:

$$\mathbf{Pr}\left[|\hat{\pi}-\pi| \geq \frac{1}{2(n+1)^{2a}}\right] < e^{-\frac{(n+1)}{8}}. \tag{18}$$

To decide whether the most "popular" answer of machine M_i at length $n+1$ is 1 or 0, our machine N, which we are going to construct, obtains estimation $\hat{\pi}$ and compares it to some threshold θ_x. This threshold depends on input provided to machine N. We define threshold θ_x so that it is uniformly spread over some interval centered at 1/2:

$$\theta_x = 1/2 + \frac{x-1/2}{(n+1)^a}, \tag{19}$$

where x in the enumerator is interpreted as a binary number. Clearly,

$$1/2 - \frac{1}{2(n+1)^a} \leq \theta_x \leq 1/2 + \frac{1}{2(n+1)^a}. \tag{20}$$

Given an input x of length n, machine N performs as follows:

find i such that $n \in S_i$
if $n = n_i^*$ **then**
  for every input $y \in \{0,1\}^{n_i}$
    deterministically compute $V_{M_i}(y)$
  return $\neg$ majority$_{y\in\{0,1\}^{n_i}}\{V_{M_i}(y) : V_{M_i}(y) \neq \perp\}$
else
  let $\hat{\pi}$ = Estimate-Over-Inputs $(M_i, 1^{n+1})$
  if $\hat{\pi} > \theta_x$ **return** 1 **else return** 0

Let a language L_N consists of those strings x that N accepts, i.e.

$$L_N = \{x : V_N(x) = 1\}. \tag{21}$$

Complement of L_N consists of those strings x such that $V_N(x)$ is either 0 or $\perp$. The two following lemmas establish our theorem.

Lemma 13.

$$L_N \in \mathbf{heur}_{1-1/n^a}\mathbf{BPP}.$$

Proof: Obviously, N is a polynomial-time probabilistic machine. Also, it is clear that machine N satisfies the promise of bounded two-sided error on all inputs of length n_i^*. Now assume that $n < n_i^*$ and let us prove that N satisfies the promise on a fraction at least $1 - 1/n^a$ of inputs of length n. Let

$$X^1 = \left\{x \in \{0,1\}^n : \theta_x \leq \theta - \frac{1}{2(n+1)^{2a}}\right\} \quad X^0 = \left\{x \in \{0,1\}^n : \theta + \frac{1}{2(n+1)^{2a}} \leq \theta_x\right\} \tag{22}$$

$$X^\perp = \left\{x \in \{0,1\}^n : \theta - \frac{1}{2(n+1)^{2a}} < \theta_x < \theta + \frac{1}{2(n+1)^{2a}}\right\} \tag{23}$$

By (18), for any input $x \in X^1$, N obtains $\hat{\pi} > \theta_x$ (and outputs 1) with probability at least $1 - e^{-\Theta(n)}$. Also, for any input $x \in X^0$, N obtains $\hat{\pi} < \theta_x$ (and outputs 0) with probability at least $1 - e^{-\Theta(n)}$.

These are only inputs in $X^\perp$ on which N may violate the promise. Fortunately, such inputs constitute a fraction less than $1/n^a$ of inputs of length n:

$$\frac{|X^\perp|}{2^n} < \frac{1}{(n+1)^{2a}} : \frac{1}{(n+1)^a} + \frac{1}{2^n} < \frac{1}{n^a}. \tag{24}$$

The first inequality follows from (23), and also (19) and (20). □

Lemma 14.

$$L_N \notin \mathbf{heur}_{1/2+1/n^a}\mathbf{BPTime}[n^c]$$

PROOF: This proof goes by contradiction. Assume for some machine M_i, for any input length n, on a fraction at least $1/2 + 1/n^a$ of inputs x of this length, $V_{M_i}(x) = L_N(x)$.

Consider a set S_i of input lengths that corresponds to machine M_i. Let v be the most "popular" answer other than $\perp$ of machine M_i at the first input length in the set:

$$v = \underset{x\in\{0,1\}^{n_i}}{\text{majority}}\{V_{M_i}(x) : V_{M_i}(x) \neq \perp\}. \tag{25}$$

We want to show that our assumption that M_i computes L_N on a large fraction of inputs of every input length leads to a "statistical" contradiction. For this purpose, let

$$A^{(n)}_{M_i} = \{x \in \{0,1\}^n : V_{M_i}(x) \in \{v, \perp\}\} \tag{26}$$

$$A^{(n)}_{N} = \{x \in \{0,1\}^n : V_{N}(x) \in \{v, \perp\}\}. \tag{27}$$

By (25), $|A^{(n_i)}_{M_i}| \geq \frac{1}{2} \cdot 2^{n_i}$. However, under our assumption, the following claim is true, therefore we obtain a desirable contradiction.

Claim 15. *Under our assumption, for every $n \in S_i$,*

$$A^{(n)}_N = \emptyset \quad \textit{and} \quad |A^{(n)}_{M_i}| \leq (1/2 - 1/n^a) \cdot 2^n.$$

First, note that the bound on the size of $A^{(n)}_{M_i}$ follows from our assumption that M_i recognizes L_N on a fraction at least $1/2+1/n^a$ of inputs of every length n and from the emptiness of $A^{(n)}_N$. To prove that $A^{(n)}_N = \emptyset$, we proceed by induction on n. For input length $n = n^*_i$, due to the deterministic simulation, we have $A^{(n)}_N = \emptyset$.

Now, let n be any input length less than n^*_i in our set S_i. By the induction hypothesis, $|A^{(n+1)}_{M_i}| \leq (1/2 - 1/(n+1)^a) \cdot 2^{n+1}$. In case $v = 1$, we have $\pi < 1/2 - 1/(n+1)^a + e^{-\Theta(n)}$ (see (17)), and, since $\theta_x \geq 1/2 - 1/(2(n+a)^a)$, it holds that $V_N(x) = 0 = \neg v$ for every x of length n. The term $e^{-\Theta(n)}$ originates from Amplification-Over-Randomness. In case $v = 0$, we have $\pi > 1/2 + 1/(n+1)^a - e^{-\Theta(n)}$ and $V_N(x) = 1 = \neg v$ for every x of length n. □

5 Time Hierarchy for Heuristic Algorithms in MA

Definition 16 (Merlin-Arthur Games)**.** *Assume a machine M. Let*

$$V_M(x) = \begin{cases} 1 & \textit{if } \exists w\, \mathbf{Pr}_r[M(x,w,r) = 1] > 2/3 \\ 0 & \textit{if } \forall w\, \mathbf{Pr}_r[M(x,w,r) = 1] < 1/3 \\ \perp & \textit{otherwise} \end{cases} \tag{28}$$

We say that M is a correct Merlin-Arthur game on an input x if its result $V_M(x)$ is not equal to $\perp$.

Definition 17. *A language L belongs to a class $\mathbf{heur}_{\delta(n)}\mathbf{MATime}[n^c]$ if there exists a machine $M(x,w,r)$ such that for any input length n, it runs in time $O(n^c)$, and*

$$|\{x : V_M(x) = L(x)\}| \geq \delta(n) \cdot 2^n. \tag{29}$$

We say that M solves L on a fraction at least $\delta(n)$ of inputs of every length n. One may also define a class $\mathbf{heur}_{\delta(n)}\mathbf{MA}$ in a straightforward way.

Theorem 18. *For any positive constants a and c,*

$$\mathbf{heur}_{1-1/n^a}\mathbf{MA} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{MATime}[n^c].$$

For the purpose of proving this theorem, we need a little bit more elaborated family of mixers than the one from Lemma 4. In what follows, we need a family of mixers that "saves" $a\lceil \log n \rceil$ bits on inputs of length n that constitute the left party of a mixer. These bits, denoted with k, are used by N to select a threshold θ_k which is used for deciding whether answer of M_i on input $x' \circ k$ is 0 or 1.

The mixers that we need can be obtained from expander graphs (for example, from Margulis construction [Mar73, GG81]):

Lemma 19. *There exists an explicit family of $1/(2(n+1)^a)$-mixers $G_n = ([2^{n-a\lceil \log n \rceil}], [2^{n+1}], E_n)$. The regularity of G_n is at most polynomial in n.*

Let $\pi_y(w)$ be the probability of y being accepted by machine M_i provided a witness w, i.e.

$$\pi_y(w) = \mathop{\mathbf{Pr}}_r[M_i(y, w, r) = 1]. \tag{30}$$

This probability can be estimated in a straightforward way:

```
Estimate-Over-Randomness (M_i, y, w_y)    /* |y| = n + 1 */
  for n^{2a+1} times
    simulate M_i on input y with witness w_y
  return π̂_y(w) = the fraction of positive answers
```

Let us choose an integer K so that $1/K$ is roughly equal to $1/n^a$:

$$K = 2^{a\lceil \log n \rceil}. \tag{31}$$

Chernoff bound (Proposition 12) guarantees that $\hat{\pi}$ estimates π good enough:

$$\mathbf{Pr}\left[|\hat{\pi}_y(w) - \pi_y(w)| < \frac{1}{10K}\right] > 1 - e^{-n/(200 \cdot 2^{2a})}. \tag{32}$$

To decide whether M_i accepts y provided a witness w, machine N computes $\hat{\pi}_y(w)$ and compares it to one of K different thresholds θ_k, $0 \le k \le K-1$. Which one of the thresholds machine N uses depends on its input. We define these thresholds so that they are uniformly spread over interval ranging from 2/5 to 3/5:

$$\theta_k = 2/5 + \frac{k}{5K}. \tag{33}$$

Given an input x of length n, machine N provided a witness W performs as follows:

```
find i such that n ∈ S_i
if n = n_i^* then
  for every y ∈ {0,1}^{n_i}
    deterministically compute V_{M_i}(y)
  return ¬ majority_{y∈{0,1}^{n_i}} {V_{M_i}(y) : V_{M_i}(y) ≠ ⊥}
else
  let n' = n − a⌈log n⌉
  let x = x' ∘ k so that |x'| = n'    /* then 0 ≤ k < K */
  interpret x' as a vertex on the left of G_n
  interpret W as a set of witnesses {w_y}_{y∈Γ(x')}
  for every y ∈ Γ(x') ⊆ {0,1}^{n+1}
    let π̂_y(w_y) = Estimate-Over-Randomness(M_i, y, w_y)
    if π̂_y(w_y) > θ_k then let r_y = 1 else let r_y = 0
  return majority_{y∈Γ(x')} {r_y}
```

Let a language L_N consists of those strings x that N accepts, i.e.

$$L_N = \{x : V_N(x) = 1\}. \tag{34}$$

Note that the complement of L_N consists of x such that $V_N(x)$ is either 0 or $\perp$. Our theorem follows from the two lemmas given below.

Lemma 20.

$$L_N \in \mathbf{heur}_{1-1/n^a}\mathbf{MA}.$$

PROOF: Apparently, witness $W = \{w_y\}_{y\in\Gamma(x')}$ is provided to N without seeing its coins. Therefore N is a one-round Merlin-Arthur game. Furthermore, one can see that N runs in polynomial-time. It remains to verify that N is a correct Merlin-Arthur game on a fraction at least $1-1/K$ of inputs x of every length n. Then our lemma follows as $1 - 1/K \geq 1 - 1/n^a$.

Since behavior of N at input length $n = n_i^*$ is deterministic, we have to consider only the case $n \neq n_i^*$ when proving the correctness of N. Fix any $x' \in \{0,1\}^{n-a\lceil \log n \rceil}$. Let $k_{x'}$ be the greatest k, $0 \leq k < K$, with the following property: exists advice $W = \{w_y\}_{y\in\Gamma(x')}$ such that for at least a half of y in $\Gamma(x')$, it holds that $\pi_y(w_y) > \theta_k - \frac{1}{10K}$. If no such k exists, let $k_{x'} = 0$.

We claim that any input $x' \circ k$ such that $0 \leq k < k_{x'}$ is accepted by N provided some witness W with probability at least $1 - e^{-\Theta(n)}$. Further, any input $x' \circ k$ such that $k_{x'} < k < K$ is rejected by N whatever witness W with probability at least $1 - e^{-\Theta(n)}$. We do not make any claim about the case $k = k_{x'}$.

Let us justify the above statements. In case $k < k_{x'}$, there exists W such that for at least a half of y in $\Gamma(x')$, we have

$$\pi_y(w_y) > \theta_{k_{x'}} - \frac{1}{10K} \geq \theta_k + \frac{1}{10K}. \tag{35}$$

Therefore, for at least a half of y in $\Gamma(x')$, machine N provided advice W obtains $\hat{\pi}_y(w_y) > \theta_k$ (and $r_y = 1$) with probability at least $1 - e^{-\Theta(n)}$ (recall bound (32)). Consequently, by union bound, $\mathrm{majority}_{y\in\Gamma(x')}\{r_y\} = 1$ with probability at least $1 - \frac{|\Gamma(x')|}{2} \cdot e^{-\Theta(n)} = 1 - e^{-\Theta(n)}$ over randomness of N. The latter equality holds since the regularity on the left of G_n is at most polynomial in n.

In another case, when $k_{x'} < k$, whatever witness $W = \{w_y\}_{y\in\Gamma(x')}$ is provided, for more than a half of y in $\Gamma(x')$, we have

$$\pi_y(w_y) \leq \theta_{k_{x'}} - \frac{1}{10K} < \theta_k - \frac{1}{10K}. \tag{36}$$

Therefore, whatever W, for more than a half of y in $\Gamma(x')$, machine N obtains $\hat{\pi}_y(w_y) < \theta_k$ (and $r_y = 0$) with probability at least $1 - e^{-\Theta(n)}$. Again, by union bound and the regularity of G_n, $\mathrm{majority}_{y\in\Gamma(x')}\{r_y\} = 0$ with probability at least $1 - e^{-\Theta(n)}$ over randomness of N.

Let us summarize the both cases. For any input $x = x' \circ k$ such that $k \neq k_{x'}$ ($k_{x'}$ depends on x'),

$$\text{either} \quad \exists W \ \Pr_r[N(x,W,r) = 1] > 1 - e^{-\Theta(n)} \quad \text{or} \quad \forall W \ \Pr_r[N(x,W,r) = 1] < e^{-\Theta(n)}. \tag{37}$$

Consequently, N is correct on a fraction at least $1 - 1/K \geq 1 - 1/n^a$ of inputs x of length n. Hence our claim and lemma follow. □

Lemma 21.

$$L_N \notin \mathbf{heur}_{1/2+1/n^a}\mathbf{MATime}[n^c].$$

PROOF: The proof goes by contradiction. Assume that the statement of our lemma is false, and, consequently, there exists a machine M_i such that for any input length n, for a fraction at least $1/2 + 1/n^a$ of inputs x of that length, we have $V_{M_i}(x) = L_N(x)$.

Consider a set S_i that corresponds to machine M_i. Let v be the most "popular" answer of machine M_i at the first input length in the set:

$$v = \underset{x \in \{0,1\}^{n_i}}{\text{majority}}\{V_{M_i}(x) : V_{M_i}(x) \neq \perp\} \tag{38}$$

We want to show that our assumption that M_i computes L_N on a large fraction of inputs of every input length leads to what we call a "statistical" contradiction. Let

$$A_{M_i}^{(n)} = \{x \in \{0,1\}^n : V_{M_i}(x) \in \{v, \perp\}\} \tag{39}$$

and

$$A_N^{(n)} = \{x \in \{0,1\}^n : V_N(x) \in \{v, \perp\}\} \tag{40}$$

By (38), it holds that $|A_{M_i}^{(n_i)}| \geq \frac{1}{2} \cdot 2^{n_i}$. However, under our assumption, the following claim is true, therefore we obtain a desirable contradiction.

Claim 22. *Under our assumption, for every $n \in S_i$,*

$$|A_N^{(n)}| < \frac{1}{2n^a} 2^n \quad \textit{and} \quad |A_{M_i}^{(n)}| < \Big(\frac{1}{2} - \frac{1}{2n^a}\Big) 2^n. \tag{41}$$

Note that the bound on the size of $A_{M_i}^{(n)}$ follows from our assumption and the bound on the size of $A_N^{(n)}$. The latter bound can be proved by induction on n. For input length $n = n_i^*$, due to the deterministic simulation, we have $A_N^{(n)} = \emptyset$. Next, let n be any input length less than n_i^* in our set S_i. Let

$$C = \{x' \in \{0,1\}^{n'} : |\Gamma(x') \cap A_{M_i}^{(n+1)}| < d/2\}, \tag{42}$$

where d is the regularity of G_n on the left. As G_n is a $1/(2(n+1)^a)$-mixer, we get a lower bound on the size of C:

$$\frac{|C|}{2^{n'}} \geq 1 - \frac{1}{2(n+1)^a} > 1 - \frac{1}{2n^a}. \tag{43}$$

We want to prove that for any $x = x' \circ k$ such that $x' \in C$, it holds that $V_N(x' \circ k) = \neg v$.

Assume that $v = 0$. Let $W = \{w_y\}_{y \in \Gamma(x')}$ be chosen as follows. If $V_{M_i}(y) = 1$, then let w_y be such that $\pi_y(w_y) > 2/3$. Otherwise let w_y be chosen arbitrary. By (42), for more than a half of $y \in \Gamma(x')$, it holds that $V_{M_i}(y) = \neg v = 1$. Therefore, by our choice of W and bound (32), for more than a half of y in $\Gamma(x')$, N obtains $\hat{\pi}_y(w_y) > 2/3 - \frac{1}{10K} > \theta_k$ with probability at least $1 - e^{-\Theta(n)}$. By union bound, N accepts $x = x' \circ k$ with probability at least $1 - e^{-\Theta(n)}$. This means that $V_N(x) = 1 = \neg v$.

Now we assume that $v = 1$. For more than a half of $y \in \Gamma(x')$, it holds that $V_{M_i}(y) = 0$. Thus, whatever W, for more than a half of $y \in \Gamma(x')$, N obtains $\hat{\pi}_y(w_y) < 1/3 + \frac{1}{10K} < \theta_k$ with probability at least $1 - e^{-\Theta(n)}$. By union bound, N rejects $x = x' \circ k$ with probability at least $1 - e^{-\Theta(n)}$. Thus $V_N(x) = 0 = \neg v$.

Summing up, for any $x = x' \circ k$ such that $x' \in C$, we have $V_N(x' \circ k) = \neg v$. This implies that $\frac{|A_N^{(n)}|}{2^n} \leq (1 - \frac{|C|}{2^{n'}}) < \frac{1}{2n^a}$. Hence our lemma follows. □

6 More Time Hierarchies for Heuristic Algorithms

Theorem 23. *For any positive integer k, for any positive constants a and c,*

$$\mathbf{\Sigma_k} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{\Sigma_k Time}[n^c]$$
$$\mathbf{\Pi_k} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{\Pi_k Time}[n^c].$$

Proof: This theorem can be proven similarly to Theorem 5, a time hierarchy for heuristic algorithms in **NP**. Just note that $\mathbf{\Sigma_k}$ and $\mathbf{\Pi_k}$ are closed under taking majority answer of polynomially many machines. □

Definition 24 (Arthur-Merlin Games)**.** *Assume a machine M. Let*

$$V_M(x) = \begin{cases} 1 & \text{if } \mathbf{Pr}_r[\exists w : M(x,w,r) = 1] > 2/3 \\ 0 & \text{if } \mathbf{Pr}_r[\exists w : M(x,w,r) = 1] < 1/3 \\ \perp & \text{otherwise} \end{cases} \tag{44}$$

We say that M is a correct Arthur-Merlin game on x if $V_M(x) \in \{0,1\}$.

Theorem 25. *For any positive constants a and c,*

$$\mathbf{heur}_{1-1/n^a}\mathbf{AM} \not\subseteq \mathbf{heur}_{1/2+1/n^a}\mathbf{AMTime}[n^c].$$

Proof: This proof is a simplification of the proof of Theorem 18, a time hierarchy for heuristic algorithms in **MA**. Though, one change in the algorithm of N has to be done. Machine N now receives a witness for every run of machine M_i in Estimate-Over-Randomness. This is needed because, in case of **AM**, there is no single good witness that make M_i accept x with high probability; good witness depends on the random coins of M_i.

```
Estimate-Over-Randomness (M_i, y)    /* |y| = n + 1 */
  for n^{2a+1} times
    generate a new randomness r for M_i
    receive a witness w_r for a run of M_i with randomness r
    simulate M_i on input y with witness w and randomness r
  return π̂_y = the fraction of positive answers
```

One may see that Merlin is able to provide the set of witnesses $W = \{w_r\}$ once it learns all the random coins of Arthur.

Let π_y be the probability of y having a witness that makes M_i accept y (cf. (30)), i.e.

$$\pi_y = \mathop{\mathbf{Pr}}_r[\exists w M_i(y,w,r) = 1] \tag{45}$$

Then, by Chernoff bound,

$$\mathbf{Pr}\left[|\hat{\pi}_y - \pi_y| < \frac{1}{10K}\right] > 1 - e^{-n/(200\cdot 2^{2a})}. \tag{46}$$

We make two claims. First, we claim that $L_N = \{x : V_N(x) = 1\}$ belongs to $\mathbf{heur}_{1-1/n^a}\mathbf{AM}$. Second, we claim that L_N doesn't belong to $\mathbf{heur}_{1/2+1/n^a}\mathbf{AMTime}[n^c]$.

We can justify the first claim following the lines of Lemma 20. Indeed, N is a polynomial-time Arthur-Merlin game. To prove that N is correct on a fraction at least $1 - 1/n^a$ of inputs of every length, let $k_{x'}$ be the greatest k, $0 \le k \le K$, such that for at least a half of y in $\Gamma(x')$, it holds that $\pi_y > \theta_k - \frac{1}{10K}$. If no such k exists, let $k_{x'} = 0$. In case $k < k_{x'}$, for at least a half

of y in $\Gamma(x')$, we have $\pi_y > \theta_k + \frac{1}{10K}$. In case $k > k_{x'}$, for more than a half of y in $\Gamma(x')$, it holds that $\pi_y \leq \theta_k - \frac{1}{10K}$. Therefore,

$$\text{either} \quad \mathop{\mathbf{Pr}}_r[\exists W N(x,W,r)=1] > 1 - e^{-\Theta(n)} \quad \text{or} \quad \mathop{\mathbf{Pr}}_r[\exists W N(x,W,r)=1] < e^{-\Theta(n)} \tag{47}$$

for a fraction at least $1 - 1/K \geq 1 - 1/n^a$ of inputs $x \in \{0,1\}^n$. Hence our first claim follows.

We leave the second claim solely to a reader. Its proof follows closely the lines of Lemma 21. □

Acknowledgements

The author would like to thank Edward A. Hirsch and Rahul Santhanam for helpful discussions. Besides, the author is grateful to Edward for his help with reviewing early drafts of this paper, and to Rahul for suggesting to apply the techniques to heuristic **NP**.

References

[Bar02] B. Barak. A probabilistic-time hierarchy theorem for "slightly non-uniform" algorithms. In *International Workshop on Randomization and Approximation Techniques in Computer Science*. LNCS, 2002.

[Coo73] S. Cook. A hierarchy for nondeterministic time complexity. *Journal of Computer and System Sciences*, 1973.

[FS04] L. Fortnow and R. Santhanam. Hierarchy theorems for probabilistic polynomial time. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[FS06] L. Fortnow and R. Santhanam. Recent work on hierarchies for semantic classes. *SIGACT News*, 37(3), 2006.

[FST05] L. Fortnow, R. Santhanam, and L. Trevisan. Hierarchies for semantic classes. In *ACM Symposium on Theory of Computing (STOC)*, 2005.

[GG81] O. Gaber and Z. Galil. Explicit construction of linear size superconcentrators. *Journal of Computer and System Sciences (JCSS)*, 22, 1981.

[GST04] O. Goldreich, M. Sudan, and L. Trevisan. From logarithmic advice to single-bit advice. In *Electronic Colloquium on Computational Complexity, technical reports*, 2004.

[GW00] O. Goldreich and A. Wigderson. On pseudorandomness with respect to deterministic observers. In *Proceedings of the satellite workshops of the 27th ICALP*, 2000.

[HS66] F. Hennie and R. Stearns. Two-tape simulation of multitape Turing machines. *Journal of the ACM*, 13, 1966.

[IJK06] R. Impagliazzo, R. Jaiswal, and V. Kabanets. Approximately list-decoding direct product codes and uniform hardness amplification. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

[Mar73] G. Margulis. Explicit constructions of expanders. *Probl. Pered. Inform.; English translation, Probl. Inform. Transm.*, 9(4), 1973.

[SFM78] J. Seiferas, M. Fischer, and A. Meyer. Separating nondeterministic time complexity classes. *Journal of the ACM*, 25, 1978.

[vMP06] D. van Melkebeek and K. Pervyshev. A generic time hierarchy for semantic models with one bit of advice. In *IEEE Conference on Computational Complexity*, 2006.

[Wil83] R. Wilber. Randomness and the density of hard problems. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1983.

[Žák83] S. Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 1983.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/