Limited Nondeterminism

Judy Goldsmith * Matthew A. Levy * Martin Mundhenk †
Computer Science Department
University of Kentucky

1 Introduction

Amy is a systems analyst working for a university electrical engineering department. The EE people want to be able to give her a circuit and find out if a polylogarithmic number of ones in the input can force the circuit to output a one. Amy has a BS in computer science, so she knows that CSAT (circuit satisfiability) is NP-complete. But the conditions seem restrictive enough that she decides to give it a try. First, she writes a program to see if a logarithmic number of ones will force a circuit to one. This program works well, and it doesn't take too long, even on reasonably big circuits. Her superiors are impressed, and so she decides to tackle $\log^2 n$ ones.

Amy tries the same approach, and her program works for small circuits. But, when she tries larger circuits, the program runs for ages! Her bosses are getting impatient, but no algorithm she can think of can deal with the large circuits effectively. She remembers her CS training, and decides that this must be NP-complete. Armed with a compendium of reductions, Amy sits down to prove that her problem is NP-complete....

After running through several reams of paper, Amy's enthusiasm wanes. None of her techniques help. The problem is in NP, although it only takes $\log^2 n$ guesses, but she can't reduce any NP-complete problem to it. What's she going to do?

This story is essentially the same one we tell about NP-completeness. The difference is, in this case, the problem actually has a subexponential algorithm, and doesn't seem to be NP-complete. Can we convince Amy's bosses that this problem is inherently very hard?

The key is in the nondeterministic solution. The algorithm never needs more than $\log^2 n$ nondeterministic bits. If Amy were more familiar with limited nondeterminism, she would know the answer.

As it turns out, this story has a happy ending. Amy has lunch with her theorist friend Alex, and discusses the problem. Alex does a little research, and finds that Amy's problem is complete for a class called β_2P. Every problem in β_kP needs at most $\log^k n$ nondeterministic bits and polynomial time. Since $\beta_1\mathrm{P} = \mathrm{P}$, Amy was able to solve the problem for the logarithmic case. Alex writes up an explanation for Amy's bosses, who chalk it up to experience and start looking for approximations.

Amy's situation goes right to the heart of some of the central questions in complexity theory: What is the power of nondeterminism in bounded-resource computation? Can nondeterminism decrease the time complexity of some problems? If so, how much nondeterminism is needed to make the difference?

---

*Supported in part by NSF grant CCR-9315354

†Permanent address: Universität Trier, FB IV – Informatik, D-54286 Trier. Supported in part by the Office of the Vice Chancellor for Research and Graduate Studies at the University of Kentucky, and by Deutsche Forschungsgemeinschaft (DFG), grant Mu 1226/2-1

There are other problems than Amy's in NP that do not use the full nondeterministic power of NP. For instance, Pratt's algorithm [31] solves primality with only $O(n^2)$ nondeterministic bits; computing the Vapnis-Chervonenkis (V-C) dimension requires only $O(\log^2 n)$ nondeterministic bits [30]. These problems come from a wide variety of fields within computer science: cryptography and computational number theory (Pratt's algorithm, for instance); learning theory (the V-C dimension); formal language theory (Unary Generators [9]) and computational geometry [6], to name a few.

The latter results are concerned with limited nondeterminism in P and in NP. Other nondeterministic classes can also be restricted in natural ways. In this survey, we will also examine the power of limited nondeterminism in real-time, within nondeterministic NC circuits, and even in NE. In the work surveyed, the bounds on nondeterminism are given *a priori*. There are also classes which implicitly use limited nondeterminism and alternation. For example, the classes LOGNP and LOGSNP, defined by Papadimitriou and Yannakakis [30], are both contained in β_2P.

There are also hierarchies defined by both limited nondeterminism and some form of alternation, such as the W-hierarchy [15], a hierarchy within NP of parameterized problems [13], and the Sharply Bounded Hierarchy, an alternation-based hierarchy built on quasilinear time and $O(\log n)$-bit existential and universal quantifiers [5, 9].

Extensive studies are done on limited nondeterminism in the context of finite automata and pushdown automata. (See [19], [20], [26], [29], [28], and [35] for finite automata, and [32], [33], and [37] for pushdown automata.) In this context, the limits on nondeterminism are more likely to be a function of the size of the automaton than of the length of the input for a computation. Thus, this work is outside the scope of this survey.

Certainly, nondeterminism is a powerful tool, and has been applied in many different settings. This survey does not address the relationship of restricting nondeterminism to restricting other resources which are strongly connected to nondeterminism (e.g., randomness), nor approximability issues [11]. Our goal is to provide an introduction to some of the work in a potentially rich, but still largely unmined, area of research.

2 Limiting Nondeterminism

Turing machines are considered here as multitape Turing machines; nondeterministic steps of Turing machines are binary choices. Kintala and Fischer [25] considered time bounded Turing machines which make a bounded number of nondeterministic steps. In a sense, nondeterminism is seen as an additional resource for deterministic computations, and the deterministic and nondeterministic complexity of a computation is measured separately.

Definition 2.1 *(cf. [25]) Let $\mathcal{F}$ be a class of functions and $\mathcal{C} = \bigcup_{f \in \mathcal{F}} \mathrm{DTIME}(f)$ be the class of sets accepted by deterministic $\mathcal{F}$-time bounded Turing machines. For a total function $g : \mathcal{N}_0 \to \mathcal{N}_0$,*[1] *$g(n)$-$\mathcal{C}$ denotes the class of sets accepted by* nondeterministic *$\mathcal{F}$-time bounded Turing machines making at most $g(n)$ nondeterministic steps on every input of length n.*

This model of limited nondeterminism *restricts* unlimited nondeterministic time-bounded computations. In order to *add* nondeterminism to complexity classes independent of the computational model for the classes, Cai and Chen [12] defined the so-called Guess-and-Check model. The nondeterministically chosen bits are appended to the input. If one of the choices can be verified as a

[1] $\mathcal{N}$ denotes the set of positive natural numbers $\{1, 2, \ldots\}$, $\mathcal{N}_0$ denotes $\{0\} \cup \mathcal{N}$.

witness, the original input is accepted. This model can also be used to describe e.g. nondeterministic circuit classes.

Definition 2.2 *[12] Let $s(n)$ be a function and let $\mathcal{C}$ be a complexity class. Then* $\mathrm{sGC}(b(n), \mathcal{C})$ *is defined to be the class of all sets A, for which there is a set $B \in \mathcal{C}$ such that for every x, $x \in A$ iff $\exists z, |z| \leq b(|x|)$ such that $\langle x, z \rangle \in B$.*

Some classes have similar characterizations in both models. For instance, adding polynomial nondeterminism to P yields NP, i.e. $\bigcup_c n^c\text{-P} = \bigcup_c \mathrm{sGC}(n^c, \mathrm{P}) = \mathrm{NP}$. On the other hand it is not clear if both these models can capture the same classes, since in the GC model the witness can be used to increase the computation time. For example, for $\mathrm{E} = \bigcup_c \mathrm{DTIME}(2^{cn})$, the class $\bigcup_c \mathrm{sGC}(n^c, \mathrm{E})$ contains $\mathrm{EXP} = \bigcup_c \mathrm{DTIME}(2^{n^c})$, whereas it is unknown whether EXP is contained in $\bigcup_c n^c\text{-E}$, a subclass of NE. Similar observations can be made for the space bounded case.[2] There, the witness can be used as additional computation space, yielding $\bigcup_c n^c\text{-L} = \mathrm{NL} \subseteq \mathrm{NP} = \bigcup_c \mathrm{sGC}(n^c, \mathrm{L})$.

3 The β-Hierarchies

Kintala and Fischer [24, 25] defined classes of sets accepted by polynomial-time bounded Turing machines that make a polylogarithmic number of nondeterministic steps. These seminal papers motivated consideration of variants of these classes. We present results on real-time, polynomial-time, and exponential-time computations with limited nondeterminism.

3.1 Polynomial Time

Kintala and Fischer [25] defined the classes $g(n)$-P as in Definition 2.1.

Using exhaustive search, every polynomial-time computation with a logarithmic number of nondeterministic steps can be simulated by a deterministic polynomial-time computation. I.e. in this notation $\mathrm{P} = (c \log n)$-P for every $c \in \mathcal{N}_0$. Furthermore, since no polynomial-time Turing machine can make more than a polynomial number of nondeterministic steps, NP is equal to the union over all polynomials f of the classes $f(n)$-P.

Using standard padding arguments, one can show that there exist NP-complete sets in n^ε-P for every $\varepsilon > 0$. The original goal in [25] was to define candidates for sets which are neither in P nor NP-complete, and thus the focus was on polynomial-time machines that use a subpolynomial and superlogarithmic amount of nondeterminism. This led to the definition of the *β-hierarchy over* P.

Definition 3.1 *(cf. [25]) For every $k \in \mathcal{N}$, $\beta_k\mathrm{P}$ is defined as the class $\bigcup_c (c \log^k n)$-P. The $\beta\mathrm{P}$-hierarchy is $\beta\mathrm{P} = \bigcup_{k \in \mathcal{N}} \beta_k\mathrm{P}$.*

As mentioned above, $\mathrm{P} = \beta_1\mathrm{P}$. The βP-hierarchy consists of $\beta_1\mathrm{P} \subseteq \beta_2\mathrm{P} \subseteq \cdots \subseteq \bigcup_{k \in \mathcal{N}} \beta_k\mathrm{P}$ and lies between P and NP (cf. [25]). Note that $\beta_k\mathrm{P} = \bigcup_c \mathrm{sGC}(c \log^k n, \mathrm{P})$ (see [12]).

3.1.1 Basic properties

If $\mathrm{P} = \mathrm{NP}$, then $\beta\mathrm{P} = \beta_1\mathrm{P} = \beta_i\mathrm{P} = \mathrm{co}\beta_j\mathrm{P}$ for all $i, j \in \mathcal{N}$. If the βP-hierarchy consists of at least two levels (i.e. if $\beta_1\mathrm{P} \subset \beta\mathrm{P}$) or if at least one of its levels is not closed under complementation (i.e. $\beta_i\mathrm{P} \neq \mathrm{co}\beta_i\mathrm{P}$ for some i), then $\mathrm{P} \neq \mathrm{NP}$. It is not known whether any "minor" collapse, such

[2] Definition 2.1 can be stated for space bounded computations in a natural way.

as $\beta_k\mathrm{P} = \mathrm{co}\beta_k\mathrm{P}$ (for $k > 1$) or $\beta\mathrm{P} = \beta_k\mathrm{P}$, would have any consequences on the collapse of other classes in the hierarchy, or even on other complexity classes.

However, such consequences would not be provable with known relativizing techniques. Beigel and Goldsmith [4] showed that "everything goes:" the classes of the βP-hierarchy can be collapsed, separated, or closed under complementation in each way which is *consistent* with its general inclusion structure given above — relative to oracles. "Consistent" means that the unrelativized inclusion structure of the βP-hierarchy and its complement must not be injured.

Theorem 3.2 *[4] Imagine any consistent inclusion structure of the βP-hierarchy. There exists an oracle relative to which the βP-hierarchy has this structure.*

3.1.2 Complete problems and other structural properties

Properties which distinguish the βP-hierarchy and NP are known only in relativized worlds. The next two theorems show similarities between NP and classes in the βP-hierarchy.

Theorem 3.3 *[2] $\beta_k\mathrm{P}$ is closed under polynomial-time disjunctive reducibility, for every $k \in \mathcal{N}$.*

This implies that every $\beta_k\mathrm{P}$ is also closed under polynomial-time and logspace many-one reductions (denoted $\leq_m^P$ and $\leq_m^L$). It is open whether $\beta_k\mathrm{P}$ is closed under polynomial-time conjunctive reducibility. In [2] this question is answered negatively in a relativized world.

Theorem 3.4 *(cf. [2, 12, 17, 25, 36]) For every $k \in \mathcal{N}$, $\beta_k\mathrm{P}$ has $\leq_m^L$-complete sets.*

In [25] "generic" complete sets were shown to be complete for $\beta_k\mathrm{P}$. Those sets are

$$\beta_k\text{-C} = \{\langle M, x, 1^t\rangle \mid M(x) \text{ accepts in } \leq t \text{ steps using } \leq \log^k |x| \text{ nondeterministic steps}\}.$$

In [2] it is proved that the following variants of the circuit value problem are complete for the $\beta_k\mathrm{P}$ classes:

$$\beta_k\text{-}\textsc{Cvp} = \{\langle x, C\rangle \mid C \text{ is a circuit with } |x| + \lceil\log^k |x|\rceil \text{ inputs which outputs 1 on some input } xz\}.$$

These are generalizations of P-complete sets, leading to the conjecture that no restriction of an NP-complete set would be $\beta_k\mathrm{P}$-complete [2]. Farr [17] and Szelepcsényi [36] disproved that conjecture. As an example we give the definition of a variant of the NP-complete set SAT of satisfiable Boolean formulae ϕ in conjunctive normal form, i.e. ϕ is a set of clauses of literals.

For a formula ϕ and a partial assignment $\mathcal{A}$ of literals from ϕ, let $\phi^{\mathcal{A}}$ denote the formula obtained from ϕ by deleting all clauses which contain a literal in $\mathcal{A}$ and all literals which complements are in $\mathcal{A}$. If one of the clauses becomes empty, then $\mathcal{A}$ does not satisfy ϕ. Every unit clause $\{l\}$ in ϕ *forces* any satisfying assignment including $\mathcal{A}$ to contain l. If iterating this process leads to an empty formula, we say that $\mathcal{A}$ *forces* a satisfying assignment. Define β_k-SAT to be the set of all pairs $\langle\phi, V\rangle$ of formula ϕ and set V of at most $\log^k |\phi|$ variables in ϕ, for which there exists an assignment $\mathcal{A}$ to all variables in V that forces a satisfying assignment for ϕ. Note that β_1-SAT is p-isomorphic to the P-complete complement of the unit resolution problem.

Theorem 3.5 *[17, 36] For every $k \in \mathcal{N}$, β_k-SAT is $\leq_m^L$-complete for $\beta_k\mathrm{P}$.*

The above algorithm can be described as "greedy" [36], or by saying that an assignment of Boolean values to the variables is "forced" [17]. The proofs of Theorem 3.5 can be extended to other P-complete problems solvable having "greedy" or "forcing" algorithms. The β_kP-completeness of variants of 3SAT, vertex cover, clique, Hamiltonian circuit, and 3-dimensional matching are proved in [36], and of restrictions of 3-colourability and Hamiltonian circuit in [17].

Cai and Chen [12] considered restrictions of the weighted circuit satisfiability problem. They proved, for every $k \in \mathcal{N}$, the β_kP-completeness of

$$\textsc{Bwcs}_k = \{C \mid C \text{ is a circuit with } m \text{ inputs which outputs 1 on some input containing at most } \log^{k-1} m \text{ many 1s}\}.$$

There are more structural similarities between classes in βP and NP. Several equivalent characterizations of the P = NP question can be translated to the P = β_kP question. In [3] it is shown that P = NP if and only if an NP-complete set reduces to some sparse set via a certain type of reducibility. A set is sparse if it has at most polynomially many elements of each length. Informally, this result gives a lower bound on the information content of NP-complete sets. The same lower bound can be proven for sets in βP.

Theorem 3.6 *For $k \in \mathcal{N}$, $\mathrm{P} = \beta_k\mathrm{P}$ iff a $\leq_m^L$-complete set for β_k polynomial-time reduces via a bounded truth-table reducibility to a set which polynomial-time conjunctively reduces to a sparse set.*

For every $\varepsilon > 0$, NP is the closure of NTIME(n^ε) under logspace reductions, but every β_kP is closed under these reductions. This yields

Theorem 3.7 *[34] For any $k \in \mathcal{N}$, $\beta_k\mathrm{P} = \mathrm{NP}$ iff $\mathrm{NTIME}(n^\varepsilon) \subseteq \beta_k\mathrm{P}$ for some $\varepsilon > 0$.*

There are some differences between the β_kP and the $\mathrm{N}^k\mathrm{P}_m$ classes defined in Section 4. For instance, the $\mathrm{N}^k\mathrm{P}_m$ classes are closed under quasilinear-time reductions, but not under polynomial-time reductions. Furthermore, Theorems 3.7 and 4.5 show that these hierarchies relativize in dramatically different ways.

One can also characterize βP in terms of oracle computations. The following characterization uses nondeterministic polylogarithmic time bounded oracle machines as defined in [23]. Since polylogarithmic time does not suffice to copy the input on the oracle tape, it is assumed that the input is written on the oracle tape for free.

Theorem 3.8 *[2]* $\beta\mathrm{P} = \mathrm{NPL}^{\mathrm{P}}$.

Another characterization of βP is obtained using robust oracle machines. In [27] it is shown that NP is the class of sets accepted by one-sided robust polynomial-time bounded oracle machines using an NP oracle, formally $\mathrm{P}^{\mathrm{NP}}_{1\text{-help}} = \mathrm{NP}$. Bounding the number of queries to the oracle yields a characterization of βP.

Theorem 3.9 *[2]* $\mathrm{P}^{\mathrm{NP}}_{1\text{-help}}[\log^k n] = \mathrm{P}^{\beta_k\mathrm{P}}_{1\text{-help}}[\log^k n] = \beta_k\mathrm{P}$.

3.2 Below and Above Polynomial Time

Hemaspaandra and Jha [22] considered limited nondeterminism in linear-exponential time computations and defined the *β-hierarchy over* E.

Definition 3.10 *[22] For every $k \in \mathcal{N}$, β_kE is defined as $\bigcup_c (cn^k)$-E. The βE-hierarchy is $\beta\mathrm{E} = \bigcup_{k\in\mathcal{N}} \beta_k\mathrm{E}$.*

Observe that $\beta_1\mathrm{E} = \mathrm{E}$. Like βP, it is unknown whether βE is a proper hierarchy. In [22], it is shown that the βE-hierarchy, like the βP-hierarchy, doesn't have the downward separation property — relative to some oracles.

Theorem 3.11 *[22] For every $k \in \mathcal{N}$, there exists an oracle relative to which*

$$\mathrm{E} = \beta_1\mathrm{E} = \beta_k\mathrm{E} \subset \beta_{k+1}\mathrm{E} \subset \beta_{k+2}\mathrm{E} \subset \cdots$$

Wolf [38] defined nondeterministic circuits, which are allowed to have nondeterministic input gates. Let NC^i denote the class of sets decided by logspace uniform circuits of polynomial size and depth $\log^i n$, and $\mathrm{NC} = \bigcup_i \mathrm{NC}^i$.

Definition 3.12 *[38] For every $k \in \mathcal{N}$, $\beta_k\mathrm{NC} = \bigcup_c \mathrm{sGC}(c\log^k n, \mathrm{NC})$. The βNC-hierarchy is $\beta\mathrm{NC} = \bigcup_{k\in\mathcal{N}} \beta_k\mathrm{NC}$.*

In [38] it is shown that quasigroup isomorphism, Latin square isomorphism and Latin square graph isomorphism problems are contained in $\beta_2\mathrm{NC}^2$.

Similarly to the nondeterministic generalizations of P, all nondeterministic NC classes are between NC and NP, and have complete problems.

Theorem 3.13
1. *[38] $\beta_1\mathrm{NC} = \mathrm{NC}$, and $\mathrm{NP} = \bigcup_{k\in\mathcal{N}} \mathrm{sGC}(n^k, \mathrm{NC})$.*
2. *[12] $\beta_k\mathrm{NC}^i$ contains $\leq_m^L$-complete sets for for all $i \in \mathcal{N}, k \in \mathcal{N}_0$.*

A similar result as in Theorem 3.7 for time-bounded classes can be shown for space-bounded classes.

Theorem 3.14 *[38] For any $k \in \mathcal{N}$, if $\mathrm{SPACE}(n^\varepsilon) \subseteq \mathrm{sGC}(n^k, \mathrm{NC})$ for some $\varepsilon > 0$, then $\mathrm{PSPACE} = \mathrm{NP}$.*

In [5] a similar assumption for time-bounded computations yields a stronger consequence: if $\bigcup_r \mathrm{DTIME}(n^k \log^r n) \subseteq \mathrm{SBH}$ for some $k > 1$, then $\mathrm{P} = \mathrm{PSPACE}$, where SBH is the Sharply Bounded Hierarchy defined in [5].

3.3 Real-Time Computations

A *real-time machine* (cf. [8]) is a (multi-tape) Turing machine which reads a new input symbol at every step. Let R (resp. NR) denote the class of all sets accepted by deterministic (resp. nondeterministic) real-time machines. Book and Greibach [8] showed that NR is the class of sets accepted by nondeterministic linear-time Turing machines. $g(n)$-R is defined as in Definition 2.1.

Observe that $\mathrm{R} = c$-R for every constant $c \in \mathcal{N}_0$ and $\mathrm{NR} = n$-R. Unlike the nondeterministic generalizations of P and E, exhaustive search cannot be used to show whether R equals $(\log n)$-R. Hartmanis and Stearns [21] showed that $\mathrm{R} \subset \mathrm{NR}$. Fischer and Kintala [18] refined the result by showing that there is an infinite hierarchy of classes between R and $(\log n)$-R.

Theorem 3.15 *[18] Let g, h be real-time constructible functions, $g(n) \leq \log n$, and $h(n) = o(g(n))$. Then $h(n)$-R $\subset g(n)$-R.*

This yields the following proper hierarchy of classes between R and $(\log n)$-R, where $\log^{(k)} n = \underbrace{\log \cdots \log}_{k \text{ times}} n$: $\mathrm{R} \subset \cdots \subset (\log^{(k+1)} n)\text{-R} \subset (\log^{(k)} n)\text{-R} \subset \cdots \subset (\log n)\text{-R}$.

Remember that this distinguishes real-time computations from the polynomial-time or linear exponential-time computations considered above, since $\mathrm{P} = (\log n)\text{-P}$ and $\mathrm{E} = n\text{-E}$. But it is unknown whether the hierarchy of classes obtained by real-time computations with polylogarithmically many nondeterministic steps is also proper.

Definition 3.16 *$\beta_k\mathrm{R}$ denotes the class $\bigcup_c (c \log^k n)\text{-R}$. The $\beta\mathrm{R}$-hierarchy is $\beta\mathrm{R} = \bigcup_{k \in \mathcal{N}} \beta_k\mathrm{R}$.*

The βR-hierarchy has the inclusion structure $\mathrm{R} \subset \beta_1\mathrm{R} \subseteq \beta_2\mathrm{R} \subseteq \cdots \subseteq \beta\mathrm{R} \subseteq \mathrm{NR}$.

3.4 Relations between the β-Hierarchies

The structure of the βR-hierarchy is closely related to the structure of the β-hierarchies over P and E. Using padding arguments, one can relate collapses of βR to collapses of βE.

Theorem 3.17 *Let $k \in \mathcal{N}$.*

1. *$\beta_k\mathrm{R} = \beta_{k+1}\mathrm{R}$ implies $\beta_k\mathrm{P} = \beta_{k+1}\mathrm{P}$, and $\beta_k\mathrm{R} = \mathrm{co}\beta_k\mathrm{R}$ implies $\beta_k\mathrm{P} = \mathrm{co}\beta_k\mathrm{P}$.*
2. *[22] $\beta_k\mathrm{P} = \beta_{k+1}\mathrm{P}$ implies $\beta_k\mathrm{E} = \beta_{k+1}\mathrm{E}$, and $\beta_k\mathrm{P} = \mathrm{co}\beta_k\mathrm{P}$ implies $\beta_k\mathrm{E} = \mathrm{co}\beta_k\mathrm{E}$.*

Book [7] showed that the collapse of E and NE is strongly related to the structure of NP, namely $\mathrm{E} = \mathrm{NE}$ iff $\mathrm{NP} - \mathrm{P}$ contains a tally set $T \subseteq 0^*$. Hemaspaandra and Jha proved this result also for classes of limited nondeterminism between E and NE. The proof takes advantage of the fact that for every $A \in \mathrm{E}$ (resp. $A \in \beta_k\mathrm{E}$, $A \in \mathrm{NE}$), the set $A\#^{2^n} = \{x\#^{2^{|x|}} \mid x \in A\}$ is in P (resp. in $\beta_k\mathrm{P}$, NP), and $A\#^{2^n}$ is p-isomorphic to a tally set.

Theorem 3.18 *Let $k \in \mathcal{N}$.*

1. *$\beta_{k+1}\mathrm{P} = \beta_k\mathrm{P}$ iff for every $A \in \beta_{k+1}\mathrm{R}$ exists a constant $c \in \mathcal{N}$ such that $A\#^{n^c} \in \beta_k\mathrm{R}$.*
2. *[22] $\beta_{k+1}\mathrm{E} = \beta_k\mathrm{E}$ iff there is no tally set in $\beta_{k+1}\mathrm{P} - \beta_k\mathrm{P}$.*

4 Quasilinear Time

Many basic algorithmic problems, e.g. multiplication and sorting, can be performed in *quasilinear time*, i.e. in time $O(n \log^k n)$ for some constant k. The $\log^k n$ factor in the time bound makes the class of quasilinear time decidable sets robust. For example, it is independent of the number of tapes of a multitape Turing machine, and is closed under quasilinear time reductions, denoted $\leq_m^{QL}$. There are interesting open problems concerning quasilinear time bounds. It is unsolved whether a circuit can be topologically ordered in quasilinear time on a Turing machine, whereas it is known that quasilinear time suffices for this task on a random access machine.

Definition 4.1 *[9] Let $l \in \mathcal{N}$ and $m \in \mathcal{N}_0$. Then P_l denotes the class $\bigcup_r \mathrm{DTIME}(n^l \log^r n)$, NP_l is $\bigcup_r \mathrm{NTIME}(n^l \log^r n)$, and $\mathrm{N}^m\mathrm{P}_l$ is $(m \log n)\text{-P}_l$.*

Using exhaustive search to simulate nondeterministic computations deterministically yields:

Observation 4.2 *[9] For any m and l, $\mathrm{N}^m\mathrm{P}_l$ is contained in P_{m+l}, and $\bigcup_{m,l \in \mathcal{N}} \mathrm{N}^m\mathrm{P}_l = \mathrm{P}$.*

4.1 Basic Properties

These classes fall into the Guess-and-Check model: $\mathrm{N}^m\mathrm{P}_l = \mathrm{sGC}(m \log n, \mathrm{P}_h)$ (cf. [12]). This hierarchy exhibits the downward separation property w.r.t. both time and nondeterminism.

Theorem 4.3 *[9] For all $m, l \in \mathcal{N}_0$,*

1. *if $\mathrm{N}^1\mathrm{P}_l \subseteq \mathrm{P}_l$, then $\mathrm{N}^1\mathrm{P}_k \subseteq \mathrm{P}_k$ for all $k \geq l$, and*
2. *if $\mathrm{N}^{m+1}\mathrm{P}_l \subseteq \mathrm{N}^m\mathrm{P}_l$, then $\mathrm{N}^k\mathrm{P}_l \subseteq \mathrm{N}^m\mathrm{P}_l$ for all $k \geq m$.*

In fact, this theorem relativizes. The hierarchy also exhibits a kind of upward separation:

Theorem 4.4 *[9] Suppose that for all l, there is an m such that $\mathrm{N}^m\mathrm{P}_1$ is not contained in P_l. Then $\mathrm{P} \neq \mathrm{NP}$.*

Unlike the βP-hierarchy, there are only countably many relativized configurations of this hierarchy possible.

Theorem 4.5 *[9] There are oracles relative to which:*

1. *$\mathrm{N}^m\mathrm{P}_k = \mathrm{P}_k$ for all m and k, and yet $\mathrm{NP} \not\subseteq \mathrm{P}$.*
2. *$\mathrm{P}_m \neq \mathrm{N}^1\mathrm{P}_m$ and $\mathrm{N}^1\mathrm{P}_m = \mathrm{NP}_m$ for all m.*
3. *$\mathrm{N}^i\mathrm{P}_j \neq \mathrm{N}^k\mathrm{P}_l$ for all distinct pairs (i, j) and (k, l) and also $\mathrm{P} \neq \mathrm{NP}$.*

4.2 Complete Sets

Theorem 4.6 *[9] $\mathrm{N}^m\mathrm{P}_l$ contains $\leq_m^{QL}$-complete sets for all $m, l \in \mathcal{N}_0$.*

There are "generic" complete sets for these classes, similar to those for the β_kP. Other complete problems are obtained from NP-complete problems by bounding the size of allowable witnesses for membership. Let $\mathrm{CS{\small AT}}(k)$ be the set of satisfiable, topologically ordered circuits that have fewer than $k \log n$ inputs, where n is the number of gates in the circuit. (Constants 0 and 1 are not counted as inputs.)

Theorem 4.7 *[9] For all $k \in \mathcal{N}_0$, $\mathrm{CS{\small AT}}(k)$ is $\leq_m^{QL}$-complete for $\mathrm{N}^k\mathrm{P}_1$.*

Cai and Chen [12] give variants of the bounded weighted circuit satisfiability problem complete for every $\mathrm{N}^m\mathrm{P}_1$. Additional complete problems for $\mathrm{N}^k\mathrm{P}_1$ can be obtained by the "forcing" or "greedy" technique of Farr [17] and Szelepcsényi [36] (see Section 3) by restricting the amount of nondeterminism to $k \log n$ bits instead of $O(\log^k n)$. For example, the problem $\mathrm{S{\small HORT}S{\small AT}_{ord}}(k)$ of Buss and Goldsmith [9] could have been obtained from β_k-SAT in this way. (Historically, however, it derived from Abrahamson *et al.* [1], who were the first to apply the technique to Boolean formulae.)

Farr's [17] reductions from β_k-SAT to restricted versions of the 3-colourability problem or Hamiltonian cycle problem use quasilinear time. Consequently, one can define ordered versions of these problems that are quasilinear-time equivalent to $\mathrm{S{\small HORT}S{\small AT}_{ord}}(k)$ and thus are $\mathrm{N}^k\mathrm{P}_1$-complete. The other reductions considered by Farr and Szelepcsényi use higher-degree polynomials; however, we conjecture that restrictions similar to the ordering property of $\mathrm{S{\small HORT}S{\small AT}_{ord}}(k)$ will yield other

N^kP_1-complete problems. All of these problems can be considered either as restricted versions of NP-complete problems or as generalized versions of P-complete problems.

Not all fixed-parameter versions of NP-complete problems yield complete problems for limited nondeterminism classes. Let VERTEX-COVER(k) be the set of undirected graphs that have a vertex cover of size k. For any fixed k, the set VERTEX-COVER(k) can be accepted in linear time [10, 16].

Acknowledgements. We thank Jonathan Buss for his help on this paper.

References

[1] K.R. Abrahamson, J.A. Ellis, M.R. Fellows, and M.E. Mata. Completeness for families of fixed parameter problems. Technical Report DCS-141-IR, U. Victoria (Canada), 1990.

[2] C. Àlvarez, J. Díaz, and J. Torán. Complexity classes with complete problems between P and NP-C. In *Proceedings Foundation of Computation Theory*, pages 3–24. Lecture Notes in Computer Science #380, Springer, 1989. Journal version see [14].

[3] V. Arvind, J. Köbler, and M. Mundhenk. On bounded truth-table, conjunctive, and randomized reductions to sparse sets. In *Proc. 12th Conference on the Foundations of Software Technology & Theoretical Computer Science*, pages 140–151. Lecture Notes in Computer Science #652, Springer, 1992.

[4] R. Beigel and J. Goldsmith. Downward separation fails catastrophically for limited nondeterminism classes. Submitted, ftp://ftp.cs.yale.edu/pub/beigel/bg-beta-draft.PS.gz.

[5] S. A. Bloch, J.F. Buss, and J. Goldsmith. Sharply bounded alternation within P. Submitted, http://www.cs.engr.uky.edu/~goldsmit/main.ps.

[6] S.A. Bloch, J.F. Buss, and J. Goldsmith. How hard are n^2-hard problems? *SIGACT News*, 91:83–85, 1994.

[7] R. Book. Tally languages and complexity classes. *Information and Control*, 26:186–193, 1974.

[8] R. Book and S. Greibach. Quasi-realtime languages. *Mathematical Systems Theory*, 4:97–111, 1970.

[9] J.F. Buss and J. Goldsmith. Nondeterminism within P. *SIAM Journal on Computing*, 22:560–572, 1993.

[10] S.R. Buss. Personal communication. See [9].

[11] L. Cai and J. Chen. On fixed-parameter tractability and approximability of NP optimization problems. *Journal of Computer and System Sciences*, 1996.

[12] L. Cai and J. Chen. On the amount of nondeterminism and the power of verifying. *SIAM Journal on Computing*, 1996.

[13] L. Cai, J. Chen, R.G. Downey, and M.R. Fellows. On the structure of parameterized problems in NP. submitted.

[14] J. Díaz and J. Torán. Classes of bounded nondeterminism. *Mathematical Systems Theory*, 23:21–32, 1990. Journal version of parts of [2].

[15] R.G. Downey and M.R. Fellows. Fixed-parameter intractability. In *Proceedings 7th Structure in Complexity Theory Conference*, pages 36–49. IEEE, 1992.

[16] R.G. Downey and M.R. Fellows. *Parameterized Computational Feasibility*, pages 219–244. Birkhäuser, 1995.

[17] G. Farr. On problems with short certificates. *Acta Informatica*, 31:479–502, 1994.

[18] P. Fischer and C.M.R. Kintala. Real-time computations with restricted nondeterminism. *Mathematical Systems Theory*, 12:219–231, 1979.

[19] J. Goldstine, C.M.R. Kintala, and D. Wotschke. On measuring nondeterminism in regular languages. *Information and Computation*, 82(2):179–194, June 1990.

[20] J. Goldstine, H. Leung, and D. Wotschke. On the relation between ambiguity and nondeterminism in finite automata. *Information and Computation*, 100:261–270, 1992.

[21] J. Hartmanis and R.E. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[22] L.A. Hemaspaandra and S.K. Jha. Defying upward and downward separation. *Information and Computation*, 121, 1995.

[23] R. Impagliazzo and M. Naor. Decision trees and downward closures. In *Proceedings 3rd Structure in Complexity Theory Conference*, pages 29–38. IEEE, 1988.

[24] C.M.R. Kintala. *Computations with a restricted number of nondeterministic steps.* PhD thesis, Pennsylvania State University, 1977.

[25] C.M.R. Kintala and P. Fischer. Refining nondeterminism in relativized complexity classes. *SIAM Journal on Computing*, 13:329–337, 1984.

[26] C.M.R. Kintala and D. Wotschke. Amounts of nondeterminism in finite automata. *Acta Informatica*, 13:199–204, 1980.

[27] K. Ko. On helping by robust oracle machines. *Theoretical Computer Science*, 52:15–36, 1987.

[28] H. Leung. On finite automata with limited nondeterminism. In *Proceedings Mathematical Foundations of Computer Science*, pages 355–363. Lecture Notes in Computer Science #629, Springer, 1992.

[29] A. Meyer and M. Fischer. Economics of description by automata, grammars and formal systems. In *Proceedings of 12th SWAT Symposium*, pages 188–191, 1988.

[30] C.H. Papadimitriou and M. Yannakakis. On limited nondeterminism and the complexity of the V-C dimension. In *Proceedings 8th Structure in Complexity Theory Conference*, pages 12–18. IEEE, 1993.

[31] V. R. Pratt. Every prime has a succinct certificate. *SIAM J. Comp.*, 4:214–220, 1975.

[32] K. Salomaa and S. Yu. Degrees of nondeterminism for pushdown automata. In *Proceedings 8th FCT*, volume 529, pages 380–392. Lecture Notes in Computer Science, Springer, 1991.

[33] K. Salomaa and S. Yu. Limited nondeterminism for pushdown automata. *Bulletin of the EATCS*, 50, 1993.

[34] L. Sanchis. Constructing language instances based on partial information. *Int. Journal of Foundations of Computer Science*, 5, 1994.

[35] I. Simon. The nondeterministic complexity of a finite automaton. In M. Lothaire, editor, *Mots - mélanges offerts à M.P. Schützenberger*, pages 384–400. Hermes, Paris, 1990.

[36] R. Szelepcsényi. β_k-complete problems and greediness. Technical Report TR 455, University of Rochester, Computer Science Dept., May 1993.

[37] D. Vermeir and W. Savitch. On the amount of nondeterminism in pushdown automata. *Fundamenta Informaticae*, 4:401–418, 1981.

[38] M.J. Wolf. Nondeterministic circuits, space complexity and quasigroups. *Theoretical Computer Science*, 125:295–313, 1994.