On the Reducibility of Sets Inside NP to Sets with Low Information Content

University of Rochester, Computer Science Department
Technical Report TR-2002-782

Mitsunori Ogihara*
Department of Computer Science
University of Rochester
Rochester, NY 14627, USA
ogihara@cs.rochester.edu

Till Tantau†
Fakultät für Elektrotechnik und Informatik
Technische Universität Berlin
10623 Berlin, Germany
tantau@cs.tu-berlin.de

May 22, 2002

Abstract

We study whether sets inside NP can be reduced to sets with low information content but possibly still high computational complexity. Examples of sets with low information content are tally sets, sparse sets, P-selective sets and membership comparable sets. For the graph automorphism and isomorphism problems GA and GI, for the directed graph reachability problem GAP, for the determinant function det, and for logspace self-reducible languages we establish the following results:

1. If GA is $\leq^{\mathrm{p}}_{\mathrm{tt}}$-reducible to a P-selective set, then GA $\in$ P.
2. If GI is $\mathcal{O}(\log n)$-membership comparable, then GI $\in$ RP.
3. If GAP is logspace $\mathcal{O}(1)$-membership comparable, then GAP $\in$ L.
4. If det is $\leq^{\log}_{\mathrm{T}}$-reducible to an L-selective set, then det $\in$ FL.
5. If A is logspace self-reducible and $\leq^{\log}_{\mathrm{T}}$-reducible to an L-selective set, then $A \in$ L.

The last result is a strong logspace version of the characterisation of P as the class of self-reducible P-selective languages. As P and NL have logspace self-reducible complete sets, it also establishes a logspace analogue of the conjecture that if SAT is $\leq^{\mathrm{p}}_{\mathrm{T}}$-reducible to a P-selective set, then SAT $\in$ P.

Keywords: computational complexity, selective, membership comparable, self-reduction, low information content, sparse, graph isomorphism, graph automorphism, circuit value problem, reachability problem.

*Supported in part by NSF grants CCR-9701911, CCR-9725021, DUE-9980943, INT-9726724, NIH grants RO1-AG18231 (5-25589) and P30-AG18254, Alzheimer's Association Grant PIO-1999-1519, and DARPA grant F30602-98-2-0133.

†Work done in part while visiting the University of Rochester, New York. Supported in part by the TU Berlin Erwin-Stephan-Prize grant.

1 Introduction

Sets with low information content typically have high computational complexity, but they become computationally tractable when a small amount of extra advice (in the sense of Karp and Lipton [33]) is available for each word length. For example, only one advice bit per word length is needed to decide a tally set. A less trivial example are P-selective sets [53], which can be decided by an NP-machine [30], if $n+1$ advice bits are available for the words of length n, and by a P-machine [34], if n^2 bits are available. Much research has focused on the question of whether sets with low information content can be helpful in deciding natural computationally difficult problems. Although it is easily seen that *every* set is $\leq_{\mathrm{m}}^{\mathrm{exp}}$-reducible to some tally set, for more realistic types of reductions like logspace or polynomial-time reductions, sets with low information content have turned out to be remarkably useless. For P-selective sets the following result is known:

Fact 1 ([2, 13, 46]). *For all $\epsilon > 0$ the following holds:*

1. *If* SAT *is $\leq_{n^{1-\epsilon}\text{-tt}}^{\mathrm{p}}$-reducible to a* P*-selective set, then* SAT $\in$ P.
2. *If* GI *is $\leq_{n^{1/2-\epsilon}\text{-tt}}^{\mathrm{p}}$-reducible to a* P*-selective set, then* GI $\in$ P.
3. *If* GA *is $\leq_{n^{1-\epsilon}\text{-tt}}^{\mathrm{p}}$-reducible to a* P*-selective set, then* GA $\in$ P.

Assuming that SAT, GI and GA are not decidable in polynomial time, each of the above results is a negative statement about reducibility to P-selective sets. We improve on the result for the graph automorphism problem by showing that *if* GA *is $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reducible to a* P*-selective set, then* GA $\in$ P. This result is a corollary of the following theorem:

Theorem 2. *If any solution of the promise problem* (1GA, GA) *is $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reducible to an $\mathcal{O}(1)$-membership comparable set, then* GA $\in$ P.

Membership comparable sets are a generalization of selective sets due to Ogihara [46]. A set A is polynomial-time f-membership comparable, f-mc in short, if for any $f(n)$ many words of length at most n we can exclude, in polynomial time, one possibility for their characteristic string with respect to A. As shown by Ogihara [46] membership comparable sets have low information content as only polynomially many advice bits per word length are needed in order to decide them in polynomial time. For the satisfiability problem Sivakumar showed the following result:

Fact 3 ([54]). *If* SAT *is $\mathcal{O}(\log n)$-mc, then* SAT $\in$ RP.

This improved earlier results of Toda [58] and Beigel [12] stating that if SAT is $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reducible to a P-selective set, then SAT $\in$ RP. The "improvement" is due to the fact that the $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reduction closure of the P-selective sets is properly contained in the class of all $\mathcal{O}(\log n)$-mc sets [56]. Our Theorem 4 also improves Toda and Beigel's result, but in a different manner, similar to Theorem 2. For the graph isomorphism problem we show that a direct analogue of Fact 3 holds.

Theorem 4. *If any solution of* $(1\text{SAT}, \text{SAT})$ *is* $\leq^{\text{p}}_{\text{tt}}$*-reducible to an* $\mathcal{O}(1)$*-mc set, then* $\text{SAT} \in \text{RP}$.

Theorem 5. *If* GI *is* $\mathcal{O}(\log n)$*-mc, then* $\text{GI} \in \text{RP}$.

One of the ultimate goals of studying reduction closures of selective sets is to prove the following important conjecture: if $\text{NP} \subseteq \text{P}^{\text{P-sel}}$ then $\text{P} = \text{NP}$. However, it is known [5] that even watered-down versions of this conjecture can be either true or false relative to some oracle. We show that many plausible logspace versions of this conjecture *do hold*. The class L-sel of L-selective sets is defined in the obvious way by requiring the machines to run in logarithmic space rather than polynomial time. The class BH(L-sel) is the boolean hierarchy [15] over L-sel.

Theorem 6. *Let* $C \in \{\text{DCFL}, \text{CFL}, \text{SAC}^1, \text{SL}, \text{UL}, \text{NL}, \text{FewL}, \text{L}^{\#\text{L}}, \text{P}, \text{Mod}_2\text{L}, \text{Mod}_3\text{L}, \dots\}$. *If* $C \subseteq \text{L}^{\text{BH(L-sel)}}$ *then* $C \subseteq \text{L}$.

As an application of Theorem 6 we prove the following result on the determinant function det, which takes an integer matrix to its determinant.

Theorem 7. *If* $\det \in \text{FL}^{\text{BH(L-sel)}}$*, then* $\det \in \text{FL}$.

For all of the classes mentioned in Theorem 6 (except for SL, UL, and FewL, where we use slightly a different argument) the proof of Theorem 6 is based on the following characterization of L in terms of logspace self-reducibility [8]. It is a strong counterpart to the characterization of P as the class of polynomial-time self-reducible set in P-sel [17] and to similar characterizations of NP [28] and of $\text{NP} \cap \text{coNP}$ [27].

Theorem 8. L *is the class of logspace self-reducible sets in* $\text{L}^{\text{BH(L-sel)}}$.

Concerning reducibility to logspace $\mathcal{O}(1)$-mc sets instead of L-selective sets, we prove the following theorem.

Theorem 9. *If* GAP *is logspace* $\mathcal{O}(1)$*-mc, then* $\text{GAP} \in \text{L}$.

An important class of languages with low information content is the class of sparse sets. Reducibility to sparse sets has been studied for a long time and the following results are known (see [5, 49] for related and even stronger results).

Fact 10 ([47]). *If* SAT *is* $\leq^{\text{p}}_{\text{btt}}$*-reducible to a sparse set, then* $\text{SAT} \in \text{P}$.

Fact 11 ([61]). *If* CVP *is* $\leq^{\log}_{\text{btt}}$*-reducible to a sparse set, then* $\text{CVP} \in \text{L}$.

We make progress on improving the last result by showing the following theorem.

Theorem 12. *Let* C *be any of the classes from Theorem 6. If all sets in* C *are* $\leq^{\log}_{\text{T}}$*-reducible to sparse sets that have only a constant number of words per length and are closed under prefix, then* $C \subseteq \text{L}$.

In particular, if CVP is $\leq_{\mathrm{T}}^{\log}$-reducible to a sparse set with the properties mentioned in Theorem 12, then CVP $\in$ L.

We use different proof techniques for our different theorems. For the result on the graph automorphism problem we use the observation of Lozano and Torán [39] and Agrawal and Arvind [1] that search non-adaptively reduces to decision for the graph automorphism problem, as well as the parallel census technique [25, 26]. For the result on the graph isomorphism problem we use the interactive proof protocol proposed by Goldreich, Micali, and Wigderson [24]. For the characterization of L we show that search reduces to decision for logspace self-reducible sets, and use the observation of Tantau [55], see also [44], that the tournament reachability problem is first-order definable and hence decidable in logarithmic space. For the conditional collapse results in Theorem 6 we build on Balcázar's work [8] on logspace self-reducible context-free languages, on Venkateswaran's characterization [62] of SAC^1, and on Allender, Barrington, and Hesse's [3] recent algorithm for converting Chinese remainder representations into binary representations in logarithmic space. For the result on GAP and logspace $\mathcal{O}(1)$-mc sets, we adapt the proof techniques used by Agrawal, Arvind, Beigel, Kummer, Ogihara, and Stephan [2, 13, 46] to logarithmic space.

This paper is organized as follows. In Section 2 we review the basic notions. In Section 3 we introduce the new notion of membership enumerability and study its basic properties. This notion allows us to obtain uniform proofs of our main theorems later on. In Section 4 we prove Theorems 2, 4 and 5 on GA, SAT, and GI. In Section 5 we treat logspace computations and prove Theorems 6, 7, 8, 9 and 12.

2 Preliminaries

For a language $A \subseteq \Sigma^*$ we define χ_A^* to be the characteristic function of A extended to tuples. It takes tuples of words as input and returns their characteristic string as output. The pairing function $\langle ., \ldots, . \rangle$ takes arbitrary tuples of words and encodes them into one word, such that coding and decoding can be done easily. The join $A \oplus B$ of two languages is defined by $A \oplus B := \{\, 0x \mid x \in A \,\} \cup \{\, 1x \mid x \in B \,\}$. For definitions of standard complexity classes see [48]. The class CFL contains the context-free languages, DCFL contains the deterministic-context-free languages. The class SL of symmetric logspace is defined in [38]. The class UL was introduced by Àlvarez and Jenner [4], and FewL and $\mathrm{Mod}_k\mathrm{L}$ are due to Buntrock et al. [18]. The class SAC^1 consists of all languages decided by logarithmically depth-bounded polynomially-sized semi-unbounded-fan-in circuits. When discussing circuit classes, selection of uniformity is an issue (see the work of Ruzzo [52]). Our choice is logspace uniformity. The nonuniform version of SAC^1 is closed under complementation [16] and this closure property holds for the logspace-uniform SAC^1 as well. For definitions of polynomial-time truth-table reductions and Turing reduction see [37], for the logspace counterparts see [36].

The notion of P-selectivity is due to Selman [53], while the notion of membership comparability is due to Ogihara [46]. Both notions can readily be generalized to logarithmic space.

Definition 13 ([53]). A *selector* for a language A is a binary function g such that for all $x, y \in \Sigma^*$

1. $g(x, y) \in \{x, y\}$,
2. if $x \in A$ or $y \in A$, then $g(x, y) \in A$.

A language is in the class P-sel if it has a selector in FP, it is in L-sel if it has a selector in FL.

Definition 14 ([46]). Let $f\colon \mathbb{N} \to \mathbb{N}$. An *$f$-membership comparing function* for a language A is a function g such that for all words $x_1, \ldots, x_{f(n)}$ of length at most n we have $b := g\big(\langle x_1, \ldots, x_{f(n)}\rangle\big) \in \{0,1\}^{f(n)}$ and $b \neq \chi_A^*\big(x_1, \ldots, x_{f(n)}\big)$. A language is in the class P-mc(f) if it has an f-membership comparing function in FP. It is in L-mc(f) if it has an f-membership comparing function in FL.

As shown in [43, 46] we have P-sel $\subsetneq$ P-mc(2) and the same proof technique can be used to show L-sel $\subsetneq$ L-mc(2). The classes L-sel and L-mc(k) share many properties with the polynomial-time counterparts P-sel and P-mc(k). For example, $\mathrm{P}^{\text{L-sel}} = \mathrm{P/poly}$ and L-sel cannot be decided with sublinear advice for any recursive time bound [30]. Likewise the exact advice bounds for P-mc(k) shown by Ronneburger [50] also hold for L-mc(k).

It is well-known that all sets in NP have *prover-verifier protocols* of the following kind:

Definition 15. A *prover-verifier protocol* for a language $A \in \mathrm{NP}$ is a pair (f, V) such that f is a polynomially length-bounded function, $V \in \mathrm{L}$, for all $x \in A$ we have $\big(x, f(x)\big) \in V$, and for all $x \notin A$ and all y we have $(x, y) \notin V$. We say that a language A *has a prover in FC*, if there exists a prover-verifier protocol for A such that f is an element of the function class *FC*.

Every set in NP has a prover in $\mathrm{FP}^{\mathrm{NP}}$. In the literature it is often said that *search reduces to decision* for a language A, if A has a prover in FP^A and the verifier works in polynomial time. We will however always use verifiers that work in logarithmic space.

Definition 16 ([23, 22]). A *promise problem* is a pair (A, B) of languages consisting of a *promise* A and a *problem* B. A *solution* of a promise problem is a set L such that $L \cap A = B \cap A$.

Examples are the promise problems $(1\mathrm{GA}, \mathrm{GA})$ and $(1\mathrm{SAT}, \mathrm{SAT})$, where 1GA is the set of graphs having at most one nontrivial automorphism, and 1SAT is the set of boolean formulas having at most one satisfying assignment. See [35] for a discussion of the properties of these problems.

Definition 17 ([8]). A language A is *logspace self-reducible*, if there exists a deterministic logspace oracle machine such that on every input x it

1. accepts x iff $x \in A$, when given A as oracle,

2. only asks queries that are lexicographically smaller than x and have the same length as x,
3. only asks queries that are identical to x except for the last $\log |x|$ bits.

Definition 18. Let C be a class of languages. The *boolean hierarchy* $\mathrm{BH}(C)$ *over* C is the smallest superset $D \supseteq C$ such that $A, B \in D$ implies $\overline{A} \in D$ and $A \cap B \in D$.

3 Membership Enumerable Languages

In this section we study the new notions of polynomial-time and logspace *membership enumerability*. The idea is that a language A is membership enumerable, if χ_A^* is enumerable in the sense of Cai and Hemachandra [19].

Definition 19. An *enumerator* for a function $f\colon \Sigma^* \to \Sigma^*$ is a function $g\colon \Sigma^* \to \Sigma^*$, such that for all x we have $g(x) = \langle x_1, \ldots, x_\ell \rangle$ for some ℓ and some words $x_i \in \Sigma^*$, and $f(x) = x_i$ for some i. A *membership enumerator* for a set A is an enumerator for χ_A^*. A language is in the class P-men if it has a membership enumerator in FP, it is in L-men if it has a membership enumerator in FL.

Theorem 20. *The class* P-men *is closed under join and* $\leq_{\mathrm{tt}}^{\mathrm{p}}$*-reductions. The class* L-men *is closed under join and* $\leq_{\mathrm{T}}^{\log}$*-reductions.*

Proof. To show that P-men is closed under $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reductions, let $A \leq_{\mathrm{tt}}^{\mathrm{p}} B \in$ P-men. To show that $A \in$ P-men, let $x_1, \ldots, x_m$ be any input words. For each x_i use the $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reduction to compute polynomially many queries $q_i^1, \ldots, q_i^\ell$ to B. Here, "polynomially many" means that ℓ is bounded by a polynomial in the total length of the input $\langle x_1, \ldots, x_m \rangle$. After stringing all queries together, using $B \in$ P-men we can come up with polynomially many possibilities for the characteristic string of the queries. Each of them induces a characteristic string for the original input words. We can then simply output the set of all of these possibilities.

Next let $A, B \in$ P-men. We show $A \oplus B \in$ P-men. Let any words $x_1, \ldots, x_m$ be given as input. By possibly rearranging their ordering, w.l.o.g. we may assume that the first ℓ many words concern membership in A, whereas $x_{\ell+1}, \ldots, x_m$ concern membership in B. We enumerate a set P of possibilities for $\chi_A^*(x_1, \ldots, x_\ell)$ and a set Q of possibilities for $\chi_B^*(x_{\ell+1}, \ldots, x_m)$ and output the set $\{\, bc \mid b \in P, c \in Q \,\}$.

For logarithmic space, note that $A \leq_{\mathrm{T}}^{\log} B$ iff $A \leq_{\mathrm{tt}}^{\log} B$ as shown by Ladner and Lynch [36]. Keeping this in mind, we can simply repeat the above proofs, replacing polynomial time by logarithmic space everywhere. We only have to be careful that we do not write down any intermediate values like the queries q_i^j, but instead recompute them whenever necessary. □

Corollary 21. *The boolean hierarchies over* P-men *and* L-men *collapse completely.*

Proof. Any boolean connective of sets $A_1, \ldots, A_k$ is $\leq_{\mathrm{btt}}^{\log}$-reducible to the join of the same sets. □

Theorem 22. *If A has a prover in $\mathrm{FP}_{\mathrm{tt}}^{\text{P-men}}$ then $A \in \mathrm{P}$. If A has a prover in $\mathrm{FL}^{\text{L-men}}$ then $A \in \mathrm{L}$.*

Proof. Assume A has a prover $f \in \mathrm{FP}_{\mathrm{tt}}^{B}$ and $B \in$ P-men. On input x we run the prover, who produces queries $q_1, \ldots, q_\ell$ such that it can deduce its certificate $f(x)$ from $\chi_B^*(q_1, \ldots, q_\ell)$. As $B \in$ P-men we can generate a list of possibilities for $\chi_B^*(q_1, \ldots, q_\ell)$ in polynomial time. For each of these possibilities we reconstruct the prover's output and check it with the verifier. If the verifier accepts one of the outputs, we accept, otherwise we reject.

Once more this proof also works for logarithmic space, as $\mathrm{FL}^B = \mathrm{FL}_{\mathrm{tt}}^B$ and as we can reproduce intermediate values as needed. □

Theorem 23. *Let $A \in$ P-mc(f) for some monotone nondecreasing, polynomial-time computable function f. Then χ_A^* has an enumerator that can be computed in time $\mathcal{O}\big(n^{\mathcal{O}(f(n))}\big)$.*

Proof. Let $A \in$ P-mc(f) via a membership comparing function g. Let an input $\langle x_1, \ldots, x_m\rangle$ be given and let $n := |\langle x_1, \ldots, x_m\rangle|$ be its length. We wish to enumerate possibilities for $\chi_A^*(x_1, \ldots, x_m)$. We will say that a bit string $b = b_1 \ldots b_j$ with $j \le m$ is *consistent*, if for every $f(n)$ many indices $i_1, \ldots, i_{f(n)} \in \{1, \ldots, j\}$ we have $g\big(\langle x_{i_1}, \ldots, x_{i_{f(n)}}\rangle\big) \neq b_{i_1} \ldots b_{i_{f(n)}}$. This means that b is a possible characteristic string of the first j many input words that does not contradict any output of g.

In a first stage we run g on every possible selection of $f(n)$ many of the input words. As there are $m^{f(n)} \le n^{f(n)}$ many such selections, this takes time $n^{f(n)}p(n)$, where p is a polynomial bounding the runtime of g.

In a second stage we inductively construct a list of all consistent bit strings of length j. For $j = 1$ we start with the list $L_1 = \{0, 1\}$. Then we cross out all inconsistent bit strings, arriving at a list L_1'. Having constructed L_j', we define $L_{j+1} := \big\{\, b0 \mid b \in L_j' \,\big\} \cup \big\{\, b1 \mid b \in L_j' \,\big\}$ and let L_{j+1}' contain all consistent bit strings in L_{j+1}.

Let ℓ_j be a bound on the maximum number of consistent bit strings of length j. As has been shown by different authors [10, 11, 20, 14] this number is bounded by $S\big(j, f(n)\big) = \sum_{i=0}^{f(n)-1} \binom{j}{i} \le j^{f(n)-1} + 1 \le n^{f(n)}$. As there are m lengths and as checking the consistency of a single bit string takes time at most $\mathcal{O}\big(n^{f(n)}\big)$, the runtime of the second stage is at most $\mathcal{O}\big(\ell_m m n^{f(n)}\big) \subseteq \mathcal{O}\big(n^{2f(n)+1}\big)$. □

In particular, for constant $f \equiv k$ we get P-mc$(k) \subseteq$ P-men, a result first shown by Nickelsen [43]. We also get $\mathrm{P}_{\mathrm{tt}}^{\text{P-sel}} \subseteq$ P-men $\subseteq$ P-mc(log).

Theorem 24. L-sel $\subseteq$ L-men.

Proof. Let $A \in$ L-sel via a selector $g \in$ FL. Let any input words $x_1, \ldots, x_m$ be given and let $V := \{x_1, \ldots, x_m\}$ be the set of these words. The selector g, which we may assume to be commutative (see [31]), induces a tournament (see [41]) whose vertex set is V and where there is an edge from x_i to x_j iff $g(x_i, x_j) = x_j$. Let G be this tournament.

It is known [29, 30] that if $V \cap A \neq \emptyset$ there must exist an index i such that the set of all vertices reachable from x_i in G is exactly $V \cap A$. As shown by Tantau [55], the tournament

reachability problem is first-order definable and hence, by an observation of Immerman [32], decidable in logarithmic space. We can thus iterate over all vertices x_i and for each of them output a bit string whose j-th bit is 1 iff x_j is reachable from x_i in the tournament G. Additionally, we output the bit string 0^m. One of the bit strings output is now guaranteed to be the characteristic string of the input words. □

4 Application to GA, GI, and SAT

In this section we prove the theorems proclaimed in the introduction on GA, GI, and SAT. For the proof of Theorem 2 on the graph automorphism problem we build on Lemma 25 below. The proof of Lemma 25 uses the parallel census technique (see [25, 26]) together with the ideas of the proofs of Lemmas 5.2 and 5.3 of [39]. For the proof Theorem 4 on the satisfiability problem we construct prover-verifier protocols for solutions of $(1\mathrm{SAT}, \mathrm{SAT})$. For the proof of Theorem 5 on the graph isomorphism problem we use the interactive proof system proposed in [24], which is also used in [39] and [9] in a similar fashion.

Lemma 25. *If A is a solution of $(1\mathrm{GA}, \mathrm{GA})$, then* GA *has a prover in* $\mathrm{FP}^A_{\mathrm{tt}}$.

Proof. We use the notations of [39]. Our prover will, on input of a graph $G = (V, E)$ with $|V| = n$, output a non-trivial automorphism, if one exists. Our verifier must check whether its input is a non-trivial automorphism, which can be done in logarithmic space.

To compute the non-trivial automorphism, using Mathon's method [40] we first construct graphs $G^{(0)}, \ldots, G^{(n-1)}$ that form a tower of pointwise stabilizers in G's automorphism group. If G has a non-trivial automorphism, there is an index i_0 such that the following conditions hold:

1. None of $G^{(i_0)}, \ldots, G^{(n-1)}$ has a non-trivial automorphism.
2. The graph $G^{(i_0-1)}$ has an automorphism mapping i_0 to some $j > i_0$, and this automorphism trivially induces an automorphism of G.
3. For every $j > i_0$ there is at most one automorphism in $G^{(i_0-1)}$ mapping i_0 to j.
4. In $G^{(i_0-1)}$ there is no non-trivial automorphism mapping i_0 to i_0.

For each i and j consider the graph $H^{ij} := G^{(i-1)}_{[i]} \cup G^{(i-1)}_{[j]}$ obtained by making a copy of $G^{(i-1)}$ and labeling i in a special way in the first copy and j in an identical way in the second copy. For all $i > i_0$ these graphs will not have any nontrivial automorphism. For $i = i_0$ for all $j > i$ the graphs H^{ij} will have at most one non-trivial isomorphism. Thus all of these graphs fulfill our promise.

As our queries to A we ask all of the H^{ij} and also $H^{ij}_{k\ell} := H^{ij}_{[k]} \cup H^{ij}_{[\ell]}$ for every pair k, ℓ of vertices in H^{ij}. Here we also label the k-th vertex and the ℓ-th vertex in the copies in the same special way. From the answers to the queries H^{ij}, $1 \le i < j \le n$, we can reconstruct the value of i_0. (This is where we use the parallel census technique.) We can also find a $j > i_0$ such that $H^{i_0 j}$ has a unique automorphism, from which an automorphism of G can be reconstructed as follows: the automorphism maps the k-th vertex to the ℓ-th vertex iff

$H_{k\ell}^{i\circ j}$ has a non-trivial automorphism. Since for all k and ℓ the graph $H_{k\ell}^{i\circ j}$ has at most one non-trivial automorphism, we can reconstruct the non-trivial automorphism of $H^{i\circ j}$ from the values of $\chi_A\big(H_{k\ell}^{i\circ j}\big)$. □

Proof of Theorem 2. By Lemma 25 the language GA has a prover in $\mathrm{FP}_{\mathrm{tt}}^{A}$. By assumption A is $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reducible to a set in P-mc(k) for some k. Hence by Theorem 23 the set A is $\leq_{\mathrm{tt}}^{\mathrm{p}}$-reducible to a set in P-men, and hence $A \in$ P-men. By Theorem 22 this yields GA $\in$ P. □

Using the same argument, but allowing for super-polynomially large sets of possible certificates, it is easy to show the following more general result: *if any solution A of the promise problem* $(1\mathrm{GA}, \mathrm{GA})$ *is* $\leq_{\mathrm{tt}}^{\mathrm{p}}$*-reducible to an f-mc set, then* GA $\in \mathrm{DTIME}\big[n^{\mathcal{O}(f(n))}\big]$.

Proof of Theorem 4. Let A be a solution of $(1\mathrm{SAT}, \mathrm{SAT})$ that is in P-men via a machine M. For an n-variable formula ϕ let Q_ϕ denote the output of M on input $\langle \phi_1, \ldots, \phi_n \rangle$, where ϕ_i is ϕ with the i-th variable substituted by 1. We define a set B as the set of all formulas ϕ for which Q_ϕ contains a satisfying assignment of ϕ.

Clearly, $B \in$ P. Just as clearly, if ϕ has no satisfying assignment then $\phi \notin B$. Now suppose ϕ has exactly one satisfying assignment. Then $\phi_i \in 1\mathrm{SAT}$ for all i, and $\chi_A^*(\phi_1, \ldots, \phi_n)$ will be the satisfying assignment of ϕ, and hence $\phi \in B$. This shows that $B \in$ P is a solution of $(1\mathrm{SAT}, \mathrm{SAT})$. By Valiant and Vazirani's result [60] we get SAT $\in$ RP. □

Proof of Theorem 5. Let GI $\in$ P-mc($c \log n$) for some c via f. We show GI $\in$ RP. Let (G_0, G_1) be a pair of graphs given as input. We run Algorithm 1. We claim that the algorithm will never accept non-isomorphic graphs and will accept isomorphic graphs with probability at least $1/n^c$.

First, consider the case $(G_0, G_1) \notin$ GI. Then the graphs in H_i will be isomorphic iff $r_i = 1$. Hence $\chi_{\mathrm{GI}}^*(H_1, \ldots, H_\ell) = r$ and $f(H_1, \ldots, H_\ell)$ will *never* be equal to r.

Next, consider the case $(G_0, G_1) \in$ GI. Then all the membership comparing function f "sees" is a bunch of isomorphic graphs. No information concerning the bit string r is contained in the graph pairs H_i. More precisely, the number of times it gets any fixed list of graph pairs as input is independent of r. Hence the probability that $f(H_1, \ldots, H_\ell)$ equals the randomly chosen r is at least $1/(2^{c \log n}) = 1/n^c$. □

5 Characterization of L and Application to Classes inside P

In this section we prove Theorems 6, 7, 8, 9, and 12 from the introduction. We begin by showing that L is the class of logspace self-reducible, logspace membership enumerable sets. We easily deduce Theorem 8. For the proof of Theorem 6 we mainly show that numerous complexity classes inside P have logspace self-reducible complete problems. To prove Theorem 12 on reducibility to sparse sets of a special type, we show that these sparse sets are in the class BH(L-sel). The proof technique of Theorem 9, which claims that GAP cannot be in L-mc(k) unless GAP $\in$ L, is a transferal of the technique used in [2, 13, 46] for showing that SAT $\in$ P-mc(k) implies SAT $\in$ P.

Algorithm 1

input (G_0, G_1) with k vertices
let $\ell := \lceil c \log n \rceil$ where n is the size of (G_0, G_1)
guess $r = r_1 \dots r_\ell \in \{0,1\}^\ell$
forall $i \in \{1, \dots, \ell\}$ **do**
  guess permutation $\sigma_i \colon \{1, \dots, k\} \to \{1, \dots, k\}$
  let $H_i := \big(G_1, \sigma_i(G_{r_i})\big)$.
if $f(H_1, \dots, H_\ell) = r$ **then**
  output "isomorphic" and **accept**
else
  output "perhaps non-isomorphic" and **reject**

Lemma 26. *If A is logspace self-reducible, then A has a prover in* FL^A.

Proof. Let A be logspace self-reducible via M. On input x the prover's queries will be all queries $q_1, \dots, q_m$ asked by M in the self-reduction tree of M on input x. Note that all of these queries are identical to x except for the last $\log |x|$ bits and can hence easily be produced in logarithmic space. The prover outputs the certificate $\chi^*_A(q_1, \dots, q_m)$. On input (x, b) the verifier checks whether $b = \chi^*_A(q_1, \dots, q_m)$. This can be done by checking whether b is internally consistent with the behavior of M. If it is the correct characteristic string, the verifier accepts if M accepts x with b as answers to its oracle queries. □

Theorem 27. L *is the class of logspace self-reducible, logspace membership enumerable sets.*

Proof. Let A be a logspace self-reducible set in L-men. Then A has a prover in FL^A by Lemma 26. By Theorem 22 we get $A \in \mathrm{L}$. □

We get Theorem 8 as a corollary, as $\mathrm{L}^{\mathrm{BH(L\text{-}sel)}} \subseteq \mathrm{L}^{\mathrm{BH(L\text{-}men)}} \subseteq \mathrm{L}^{\text{L-men}} \subseteq$ L-men. The inclusions follow, in order, from Theorem 24, Corollary 21, and Theorem 20.

Proof of Theorem 6. For each of the C we show that for all $A \in C$ there exists a $B \in \mathrm{L}^C$ such that

1. either B is logspace self-reducible and A is $\leq^{\log}_{\mathrm{T}}$-reducible to B,
2. or A has a prover in FL^B.

As $B \in \mathrm{L}^C \subseteq \mathrm{L}^{\mathrm{BH(L\text{-}sel)}} \subseteq$ L-men, from the first statement we deduce $B \in \mathrm{L}$ by Theorem 8, and hence $A \in \mathrm{L}$. From the second statement we deduce $A \in \mathrm{L}$ by Theorem 22. The first statement will be true for Cases 1 to 5 below, the second statement for Cases 6 and 7.

Case 1: $C \in \{\mathrm{NL}, \mathrm{P}\}$. As Balcázar [8] has shown that NL and P both have a logspace self-reducible $\leq^{\log}_{\mathrm{m}}$-complete sets, the first statement is true for $C \in \{\mathrm{NL}, \mathrm{P}\}$.

Case 2: $C \in \{\mathrm{DCFL}, \mathrm{CFL}\}$. Balcázar [8] has constructed a logspace self-reducible set B_1 that is $\leq^{\log}_{\mathrm{m}}$-complete for LOGDCFL, the $\leq^{\log}_{\mathrm{m}}$-reduction closure of DCFL. As $B_1 \in \mathrm{L}^{\mathrm{DCFL}}$,

the first statement is true for $C = \mathrm{DCFL}$. Balcázar also constructed a $\leq_{\mathrm{m}}^{\log}$-complete set B_2 for LOGCFL.

Case 3: $C = \mathrm{SAC}^1$. Venkateswaran [62] has shown $\mathrm{SAC}^1 = \mathrm{LOGCFL}$. Hence every set in SAC^1 reduces to the logspace self-reducible set $B_2 \in \mathrm{L}^{\mathrm{SAC}^1}$.

Case 4: $C \in \{\mathrm{Mod}_k\mathrm{L} \mid k \geq 2\}$. For each k, the problem $B := \{\langle G, s, t, i\rangle \mid G$ is a topologically-sorted dag in which the number of path from s to t is congruent i modulo $k\}$ is $\leq_{\mathrm{m}}^{\log}$-complete for $\mathrm{Mod}_k\mathrm{L}$. It is also logspace self-reducible, as on input $\langle G, s, t, i\rangle$ we can ask the queries $\langle G, s', t, i'\rangle$ for all successors s' of s and all $i' \in \{0, \ldots, k-1\}$. From the answers to these questions we can easily deduce whether $\langle G, s, t, i\rangle \in B$. Hence the first statement is true for $\mathrm{Mod}_k\mathrm{L}$.

Case 5: $C = \mathrm{L}^{\#\mathrm{L}}$. The problem of counting the paths from vertex 1 to vertex n in a topologically-sorted n-vertex dag is a well-known canonically $\leq_{\mathrm{m}}^{\log}$-complete problem for #L. Define B to be the set of all strings of the form $\langle G, 1^k, s, t, \ell\rangle$ such that G is a topologically-sorted dag, $k \geq 2$ is an integer, s and t are vertices of G, ℓ is a binary integer such that $0 \leq \ell \leq k-1$, and the number of paths in G from s to t is congruent to ℓ modulo k. Since the number k is given in unary and G is topologically sorted, B is logspace self-reducible. Recent work of Allender, Barrington, and Hesse [3] shows that converting the Chinese remainder representation of a number into its binary representation can be done in logspace-uniform NC^1, and thus in logspace. Suppose we wish to compute the number X of paths in an n-vertex topologically-sorted graph G from vertex 1 to vertex n. Since $X \leq n^n \leq 2^{n^2}$, X can be recovered in logspace from $X \bmod 2$, $X \bmod 3$, $X \bmod 5$, $\ldots$, $X \bmod p$, where p is the n^2-th prime number. As it is known [51] that the k-th prime number is $\mathcal{O}(k^2)$, the primes needed in the Chinese remainder representation of X are at most n^4. They can thus be calculated in logspace. So B is $\leq_{\mathrm{T}}^{\log}$-hard for #L and thus $\leq_{\mathrm{T}}^{\log}$-complete for $\mathrm{L}^{\#\mathrm{L}}$. This shows that the first statement is true for $\mathrm{L}^{\#\mathrm{L}}$.

Case 6: $C = \mathrm{SL}$. It is known [38, 45] that the complement of the undirected graph accessibility problem UGAP is $\leq_{\mathrm{m}}^{\log}$-complete for SL. We show that $\overline{\mathrm{UGAP}}$ has a prover in $\mathrm{FL}^{\overline{\mathrm{UGAP}}}$. It maps each input (G, s, t) to the set of all vertices reachable from s in G. Clearly, this prover is in $\mathrm{FL}^{\overline{\mathrm{UGAP}}}$. On input $\left(\langle G, s, t\rangle, I\right)$ the verifier checks whether $s \in I$, whether $t \notin I$, and whether I is closed under reachability. If so, it outputs "unreachable" and accepts. Thus, the second statement is true.

Case 7: $C \in \{\mathrm{UL}, \mathrm{FewL}\}$. We show the second statement to be true for UL. Let $A \in \mathrm{UL}$ via M. Define $B := \{\langle x, c\rangle \mid x$ is an input for M, c is a vertex in the topologically-sorted configuration dag of M, and there is a path from the initial configuration of M on input x to the accepting configuration through $c\}$. Then $B \in \mathrm{UL}$. On input x the prover queries (x, c) for every configuration c of M on input x. It passes the characteristic string of these queries with respect to B as its certificate to the verifier. On input (x, b) the verifier simulates M on input x and uses the bit string b to decide non-deterministic choices. If it reaches the accepting configuration under the "guidance" of b, it accepts. For $C = \mathrm{FewL}$ we construct the same set $B \in \mathrm{FewL}$ for $A \in \mathrm{FewL}$. Once more A has a prover in FL^B, only this time the verifier may find multiple legal nondeterministic choices in b. But then it can simply pick,

say, the smallest one. □

Proof of Theorem 7. Assume $\det \in \mathrm{FL}^{\mathrm{BH(L\text{-}sel)}}$. Then $\mathrm{L}^{\det} \subseteq \mathrm{L}^{\mathrm{BH(L\text{-}sel)}}$. The $\leq^{\log}_{\mathrm{m}}$-reduction closure of det is known [21, 57, 59, 63] to be exactly GapL, the class of functions that are the difference of two functions in #L. Hence $\mathrm{L}^{\det} = \mathrm{L}^{\mathrm{GapL}} = \mathrm{L}^{\#\mathrm{L}}$. But then $\mathrm{L}^{\#\mathrm{L}} \subseteq \mathrm{L}^{\mathrm{BH(L\text{-}sel)}}$, which yields $\mathrm{L}^{\#\mathrm{L}} = \mathrm{L}$ by Theorem 6, which in turn yields $\det \in \mathrm{FL}$. □

Proof of Theorem 12. Fix some ordering of the set Σ of symbols and let $\leq_{\mathrm{lex}}$ be the induced ordering of words. A *branch* $B \subseteq \Sigma^*$ is the set of all prefixes of an infinite string $s_1 s_2 s_3 \ldots$ of symbols $s_i \in \Sigma$. Such a branch B is the intersection of the two L-selective sets $\left\{ c \in \Sigma^* \mid c \leq_{\mathrm{lex}} s_1 \ldots s_{|c|} \right\}$ and $\left\{ c \in \Sigma^* \mid c \geq_{\mathrm{lex}} s_1 \ldots s_{|c|} \right\}$. Thus every branch B is in the boolean hierarchy over L-sel.

Any set that has only a bounded number of words per level and is closed under prefix is the union of a finite number of branches. Hence any such language is also in the boolean hierarchy over L-sel. By Theorem 6 we get the claim. □

Proof of Theorem 9. If GAP is in L-mc(k) then so is the reachability problem for dags with out-degrees at most 2, which is easily seen to be NL-complete. Let (G, s, t) be an input for this problem. We start a search from the source s, and keep track of a list of at most 2^k many vertices, which fulfills two requirements. Firstly, all vertices in the list are reachable from s. Secondly, if there is a path from s to t, the list will contain at least one vertex v from which there is a path to t.

Initially, the list contains only the source, which clearly fulfills the requirements. As long as the list has not grown to size 2^k, we remove the first element of the list and add its successors to the list. This, too, does not violate any requirement. If we ever put the target into the list, we accept.

If the list grows to size 2^k, we remove one vertex from the list having the property that it is not the only vertex from which t is reachable. To obtain such an element, we use a method also employed in [2, 13, 46]. Let v_b with $b \in \{0,1\}^k$ be the vertices in the list. We build k dags $D_1, \ldots, D_k$ as follows: each D_i consists of 2^k many copies of G. We add a new target and a new source. The target in every copy of G is connected to the new target. If the i-th bit of b is 1, the new source is connected to the vertex v_b in the b-th copy of G. If it is not, the new source is not connected to any vertex in the b-th copy. With this construction the new target of D_i is reachable from the new source of D_i, iff t is reachable in G from a vertex v_b where the i-th bit of b is 1.

We run the k-membership comparing function on $D_1, \ldots, D_k$. Note that we do not actually write these dags down anywhere, but rather dynamically calculate any bit of the code of the dags that is needed by the k-membership comparing function. Then the bit string b output by the function will be the index of a vertex that is not the only vertex from which the target t is reachable in G. We can hence remove v_b from our list.

As it is easily seen that we never reach the same list configuration twice, we get the claim. □

6 Conclusion

We have shown that a wide variety of computationally complex problems cannot be reduced to sets with low information content, unless unlikely collapses of complexity classes occur. We started by showing that the graph automorphism problem cannot be $\leq^{\mathrm{p}}_{\mathrm{tt}}$-reduced to any P-selective set, unless it can be decided in polynomial time. This is a significant improvement over the previously known results. Using the same proof technique we showed that if (1SAT, SAT) has a solution that is $\leq^{\mathrm{p}}_{\mathrm{tt}}$-reducible to an $\mathcal{O}(1)$-mc set, then RP = NP. This result is "orthogonal" to Fact 3. Our proof technique does not seem to be applicable to the graph isomorphism problem, but we still made progress on the question of whether GI is $\mathcal{O}(\log n)$-mc. If it is, then GI $\in$ RP.

A persisting open problem is the question whether SAT being $\leq^{\mathrm{p}}_{\mathrm{tt}}$-reducible to P-sel implies SAT $\in$ P, and likewise for GI. As we showed that GI $\in$ P-mc(log) implies GI $\in$ RP, we implicitly get that GI being $\leq^{\mathrm{p}}_{\mathrm{tt}}$-reducible to P-sel implies GI $\in$ RP. Interestingly, we were not able to show that GA $\in$ P-mc(log) implies GA $\in$ RP.

We also considered logspace computations, where we were able to prove even stronger results. For example, none of the three problems CVP, GAP and UGAP can be $\leq^{\log}_{\mathrm{T}}$-reducible to an L-selective set, unless they are in L. The determinant function cannot be $\leq^{\log}_{\mathrm{T}}$-reduced to an L-selective set, unless it can be computed in logarithmic space.

Many of our logspace results boil down to our characterization of L as the class of logspace self-reducible sets in $\mathrm{L}^{\mathrm{BH(L\text{-}sel)}}$, the $\leq^{\log}_{\mathrm{T}}$-reduction closure of the boolean hierarchy over L-sel. A more general form of this characterization was that L is the class of logspace self-reducible and membership enumerable sets. A much less general form is the trivial corollary that L is also the class of logspace self-reducible L-selective sets. This is the typical kind of result available in the polynomial time setting. For example, P is known to be the class of self-reducible P-selective sets, but not known to be the class of self-reducible sets in BH(P-sel), let alone in $\mathrm{P}^{\mathrm{BH(P\text{-}sel)}} = \mathrm{P/poly}$.

The presence of the Turing-reduction in our characterization of L makes it somewhat "robust." For example, Balcázar has shown a result similar to ours, namely that LOGCFL $\subseteq$ L/log implies CFL $\subseteq$ L. However, he points out that the assumption CFL $\subseteq$ L/log is insufficient to arrive at the conclusion CFL $\subseteq$ L, as L/log is not known to be closed under $\leq^{\log}_{\mathrm{m}}$-reductions. Our Theorem 6 does not suffer this problem. We arrive at the conclusion CFL $\subseteq$ L already from the assumption CFL $\subseteq \mathrm{L}^{\mathrm{BH(L\text{-}sel)}}$. Relatedly Austinat, Diekert, and Hertrampf [6, 7] have shown—unconditionally—that no inherently context-free languages is finite automaton $\mathcal{O}(1)$-membership comparable, see [6, 7] for detailed definitions. In particular, no inherently context-free language is in the boolean hierarchy over the finite automata selective sets.

We have not claimed that $\leq^{\log}_{\mathrm{T}}$-reducibility to L-mc(k) sets has any dramatic consequences. The reason is that although we could show L-sel $\subseteq$ L-men, it is not known whether L-mc(k) $\subseteq$ L-men for all k. Nickelsen [42] showed L-mc(2) $\subseteq$ L-men, but the general case remains open.

Acknowledgments

The authors are very grateful to Lane Hemaspaandra for useful suggestions and for pointers to literature.

References

[1] M. Agrawal and V. Arvind. Polynomial time truth-table reductions to P-selective sets. In *Proc. 9th Structure in Complexity Theory Conference*, pages 24–30. IEEE Computer Society Press, 1994.

[2] M. Agrawal and V. Arvind. Quasi-linear truth-table reductions to p-selective sets. *Theoretical Comput. Sci.*, 158(1/2):361–370, 1996.

[3] E. Allender, D. A. M. Barrington, and W. Hesse. Uniform circuits for division: consequences and problems. In *Proc. 16th Conf. on Computational Complexity*, pages 150–159, Los Alamitos, CA, USA, 2001. IEEE Computer Society Press.

[4] C. Àlvarez and B. Jenner. A very hard log-space counting class. *Theoretical Comput. Sci.*, 107:3–30, 1993.

[5] V. Arvind, Y. Han, L. Hemachandra, J. Köbler, A. Lozano, M. Mundhenk, M. Ogiwara, U. Schöning, R. Silvestri, and T. Thierauf. Reductions to sets of low information content. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory*, pages 1–45. Cambridge University Press, 1993.

[6] H. Austinat, V. Diekert, and U. Hertrampf. A structural property of regular frequency classes. *Theoretical Comput. Sci.*, 2001. To appear.

[7] H. Austinat, V. Diekert, U. Hertrampf, and H. Petersen. Regular frequency computations. In *Proc. RIMS Symp. on Algebraic Systems, Formal Languages and Computation*, Kyoto, 2000.

[8] J. Balcázar. Logspace self-reducibility. In *Proc. 3rd Conference on Structure in Complexity Theory*, pages 40–46. IEEE Computer Society Press, 1988.

[9] R. Beals, R. Chang, W. Gasarch, and J. Torán. On finding the number of graph automorphisms. *Chicago J. Theoretical Comp. Sci.*, 1999(1), Feb. 1999.

[10] R. Beigel. *Query-limited reducibilities.* PhD thesis, Stanford University, Stanford, USA, 1987.

[11] R. Beigel. A structural theorem that depends quantitatively on the complexity of SAT. In *Proc. 2nd Conference on Structure in Complexity Theory*, pages 28–32. IEEE Computer Society Press, 1987.

[12] R. Beigel. NP-hard sets are P-superterse unless R = NP. Technical Report 88-04, Dept. of Computer Science, Johns Hopkins University, 1988.

[13] R. Beigel, M. Kummer, and F. Stephan. Approximable sets. *Inform. Computation*, 120(2):304–314, 1995.

[14] R. Beigel, M. Kummer, and F. Stephan. Quantifying the amount of verboseness. *Inform. Computation*, 118(1):73–90, Apr. 1995.

[15] A. Bertoni, D. Bruschi, D. Joseqh, M. Sitharam, and P. Young. Generalized boolean hierarchies and boolean hierarchies over RP. In *Proc. 7th Conf. on Fundamentals of Computation Theory*, volume 380 of *Lecture Notes in Computer Science*, pages 35–46. Springer-Verlag, 1989.

[16] A. Borodin, S. Cook, P. Dymond, W. Ruzzo, and M. Tompa. Two applications of inductive counting for complemetation problems. *SIAM J. Computing*, 18(3):559–578, 1989.

[17] H. Buhrman, P. van Helden, and L. Torenvliet. P-selective self-reducible sets: A new characterization of P. In *Proc. 8th Conf. on Structure in Complexity Theory*, pages 44–51, Los Alamitos, CA, USA, 1993. IEEE Computer Society Press.

[18] G. Buntrock, C. Damm, U. Hertrampf, and C. Meinel. Structure and importance of Logspace-MOD class. *Math. Systems Theory*, 25(3):223–237, 1992.

[19] J. Cai and L. A. Hemachandra. Enumerative counting is hard. *Inform. Computation*, 82(1):34–44, July 1989.

[20] S. Clarke, J. C. Owings, Jr., and J. Spriggs. Trees with full subtrees. In *Proc. of the Sixth Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 169–172, Winnipeg, Canada, 1975.

[21] C. Damm. DET $= \mathrm{L}^{(\#\mathrm{L})}$. Technical Report Informatik-Preprint 8, Fachbereich Informatik der Humboldt-Universität zu Berlin, 1991.

[22] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Inform. Control*, 61(2):159–173, 1984.

[23] S. Even and Y. Yacobi. Cryptocomplexity and NP-completeness. In *Proc. Int. Coll. Automata, Languages and Programming (ICALP)*, volume 85 of *Lecture Notes in Computer Science*, pages 195–207. Springer-Verlag, July 1980.

[24] O. Goldreich, S. Micali, and A. Wigderson. Proofs the yield nothing but their validity and a methodology of cryptographic protocol design. In *Proc. 27th Symp. on Foundations of Computer Science*, pages 174–187, 1985.

[25] J. Hartmanis. On sparse sets in NP − P. *Inf. Process. Lett.*, 16(2):55–60, 1983.

[26] J. Hartmanis, N. Immerman, and V. Sewelson. Sparse sets in NP − P: EXPTIME versus NEXPTIME. *Inform. Control*, 65:159–181, 1985.

[27] L. Hemaspaandra, A. Hoene, A. Naik, M. Ogihara, A. Selman, T. Thierauf, , and J. Wang. Nondeterministically selective sets. *Int'l J. Found. Comp. Sci.*, 6(4):403–416, 1995.

[28] L. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. Computing solutions uniquely collapses the polynomial hierarchy. *SIAM J. Computing*, 25(4):697–708, 1996.

[29] L. Hemaspaandra, C. Nasipak, and K. Parkins. A note linear-nondeterminism, linear-sized, karp-lipton advice for the P-selective sets. *J. Universal Comp. Sci.*, 4(8):670–674, Aug. 1998.

[30] L. Hemaspaandra and L. Torenvliet. Optimal advice. *Theoretical Comput. Sci.*, 154(2):367–377, Feb. 1996.

[31] L. A. Hemaspaandra, H. Hempel, and A. Nickelsen. Algebraic properties for P-selectivity. In *COCOON 2001, International Computing and Combinatorics Conference*, number 2108 in Lecture Notes in Computer Science, Guilin, Aug. 2001.

[32] N. Immerman. *Descriptive Complexity.* Springer-Verlag, 1998.

[33] R. Karp and R. Lipton. Some connections between uniform and non-uniform complexity classes. In *Proc. 12th ACM Symp. on Theory of Computing*, pages 302–309, 1980.

[34] K.-I. Ko. On self-reducibility and weak P-selectivity. *J. Comput. Syst. Sci.*, 26:209–221, 1983.

[35] J. Köbler, U. Schöning, and J. Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhäuser, 1993.

[36] R. Ladner and N. Lynch. Relativization of questions about logspace computability. *Math. Systems Theory*, 10(1):19–32, 1976.

[37] R. E. Ladner, N. A. Lynch, and A. L. Selman. A comparison of polynomial time reducibilities. *Theoretical Comput. Sci.*, 1(2):103–123, Dec. 1975.

[38] H. Lewis and C. Papadimitriou. Symmetric space-bounded computation. *Theoretical Comput. Sci.*, 19:161–187, 1982.

[39] A. Lozano and J. Torán. On the nonuniform complexity of the graph isomorphism problem. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory, current research*. Cambridge University Press, 1993.

[40] R. Mathon. A note on the graph isomorphism counting problem. *Inf. Process. Lett.*, 8:131–132, 1979.

[41] J. W. Moon. *Topics on Tournaments*. Holt, Rinehart, and Winston, 1968.

[42] A. Nickelsen. Personal communication. 2001.

[43] A. Nickelsen. *Polynomial Time Partial Information Classes*. PhD thesis, Technische Universität Berlin, Germany, 1999.

[44] A. Nickelsen and T. Tantau. On reachability in graphs with bounded independence number. In *International Computing and Combinatorics Conference*, Lecture Notes on Computer Science. Springer-Verlag, 2002.

[45] N. Nisan and A. Ta-Shma. Symmetric logspace is closed under complement. *Chicago J. Theoretical Comp. Sci.*, 1995(1), 1995.

[46] M. Ogihara. Polynomial-time membership comparable sets. *SIAM J. Comput.*, 24(5):1068–1081, Oct. 1995.

[47] M. Ogiwara and O. Watanabe. On polynomial-time bounded truth-table reducibility of NP sets to sparse sets. *SIAM J. Comput.*, 20(3):471–483, June 1991.

[48] C. H. Papadimitriou. *Computational Complexity*. Addison–Wesley, 1994.

[49] D. Ranjan and P. Rohatgi. On randomized reductions to sparse sets. In *Structure in Complexity Theory Conference*, pages 239–242, 1992.

[50] D. Ronneburger. Upper and lower bounds for token advice for partial information classes. Master's thesis, Technische Universität Berlin, Germany, Nov. 1998.

[51] J. Rosser and L. Schoenfeld. Approximate formulas for some funcitons of prime numbers. *Illinois J. Math.*, 6:64–94, 1962.

[52] W. Ruzzo. On uniform circuit complexity. *J. Comput. Syst. Sci.*, 22:365–383, 1981.

[53] A. L. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Math. Systems Theory*, 13:55–65, 1979.

[54] D. Sivakumar. On membership comparable sets. *J. Comput. Syst. Sci.*, 59(2):270–280, 1999.

[55] T. Tantau. A note on the complexity of the reachability problem for tournaments. Technical Report TR01-092, Electronic Colloquium on Computational Complexity, www.eccc.uni-trier.de/eccc, 2001.

[56] T. Tantau. A note on the power of extra queries to membership comparable sets. Technical Report TR02-004, Electronic Colloquium on Computational Complexity, www.eccc.uni-trier.de/eccc, 2002.

[57] S. Toda. Counting problems computationally equivalent to computing the determinant. Technical Report CSIM 91-07, Department of Computer Science, University of Electro-Communications, Tokyo, Japan, May 1991.

[58] S. Toda. On polynomial-time truth-table reducibility of intractable sets to p-selective sets. *Math. Systems Theory*, 24:69–82, 1991.

[59] L. Valiant. Why is boolean complexity theory difficult? In M. Paterson, editor, *Boolean Function Complexity*, pages 84–94. London Mathematical Society, Lecture Note Series 169, Cambridge University Press, 1992.

[60] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Comput. Sci.*, 47:85–93, 1986.

[61] D. van Melkebeek. Deterministic and randomized bounded truth-table reductions of P, NL, and L to sparse sets. *Journal of Computer and System Sciences*, 58(2):213–232, 1998.

[62] H. Venkateswaran. Properties that characterize LOGCFL. *J. Comput. Syst. Sci.*, 43(2):380–404, 1991.

[63] V. Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proc. 6th Structure in Complexity Theory Conference*, volume 223 of *Lecture Notes in Computer Science*, pages 270–284. Springer-Verlag, 1991.