SIGACT News Complexity Theory Column 64

Lane A. Hemaspaandra
Dept. of Computer Science, University of Rochester
Rochester, NY 14627, USA

Introduction to Complexity Theory Column 64

Warmest thanks to Christian Glaßer, Mitsunori Ogihara, Aduri Pavan, Alan L. Selman, and Liyu Zhang for this issue's column. I find the advances they are describing inspiring. These authors had the courage to approach fundamental, long-standing open issues on what properties many-one complete sets for crucial classes, such as NP, possess, and these authors quite beautifully resolved those issues. In particular, their work shows that for a wide range of central complexity classes (including NP), the nontrivial many-one complete sets are always many-one autoreducible, and their work also (combined with earlier work of Ambos-Spies) shows that for nontrivial sets, being many-one mitotic (basically, being a set that some P set can slice into two parts, each of which is equivalent to the other and to the original set) and being many-one autoreducible coincide. These are two separate advances, each impressive, and in their wake, the field now knows that all nontrivial NP-complete sets are many-one autoreducible and many-one mitotic. The work of these authors, and others, has established a broad range of insights into what redundancy properties complete sets inherently posses, and this article presents that exciting stream of work. Enjoy!

Please stay tuned to future issues for what I suspect is going to be a particularly exciting set of articles in this column (note: in all cases the topics/titles are tentative): Zeev Dvir (on "From Randomness Extraction to Rotating Needles"), Scott Aaronson (topic TBA), Ronen Shaltiel (writing on derandomization), and Marius Zimand (writing on Kolmogorov complexity extraction).

Finally, as this article will with luck get to people's mailboxes slightly before the submission deadline for the new conference Innovations in Computer Science (ICS), let me please point everyone to ICS 2009 and to the possibility of submitting there. The conference is aimed at showcasing big-picture, conceptual, and novel work.

Most readers of this introduction will by now have already seen some of the blog/comment discussions of what might be on the other side of divide (from ICS), but just in case anyone didn't yet see the web site for the fictitious SLOGN conference, let me, with thanks for Felix Brandt for pointing me to this, mention that the web site `slogn.org` is pretty funny. Scott Aaronson in his blog outed the two people behind that web site, and I'd suggest, despite that conference's stated “triple-bind” policy of even an anonymous PC, that those two people be made SLOGN PC chairs for life. (If that humorous web site wanted to be even more provocative, I suppose it could have suggested that even the *content* of the submissions would be hidden from the anonymous PC and the anonymous reviewers, and that the PC/reviewers would be shown, for a paper, *only* the authors' names. This would encourage people to put in the time to build career reputations as Master Sloggers.)

But, seriously now, my own two cents are that I think it is utterly wonderful for a conference to focus on innovation and creativity; ICS seems a great idea. But I suspect that everyone from Master Innovators to Master Sloggers (note: collect both T-shirts and win a free set of steak knives!) agrees that what one most loves to see is conference papers that are interesting, important, and open new directions, and doing any one of those three (admittedly highly subjective and overlapping) things very well is very good (and perhaps far too rare)... both for the cases when achieving that takes real slogging (and ideally building techniques that not only answer up front something interesting, but also may find future use elsewhere) and for the cases when achieving that is done by some simple yet stunning bit of poetry (for example, the Immerman-Szelepcsényi Theorem's proof). Trite though it is to say, both types of Masters, as well as the rest of us, do have the same daunting goal: moving the field's knowledge forward (as best we can). And speaking of work moving the field's knowledge forward (in this case, very nicely indeed), here is this issue's article.

Guest Column: Autoreducibility and Mitoticity[1]

***Christian Glaßer*[2], *Mitsunori Ogihara*[3], *A. Pavan*[4], *Alan L. Selman*[5], *Liyu Zhang*[6]**

Abstract

Autoreducibility and mitoticity express weak forms of redundancy of information that a set might possess. We describe many results known about these concepts. Mitoticity always implies autoreducibility, but the converse holds in some situations and fails in others.

Among the results we describe are that NP-complete sets are many-one autoreducible and that every many-one autoreducible set is many-one mitotic. In particular, every infinite NP-complete set splits by a set in P into two disjoint, infinite NP-complete sets.

[1]© C. Glaßer, M. Ogihara, A. Pavan, A. L. Selman, and L. Zhang, 2009.
[2]University at Würzburg, Germany.
[3]University of Miami.
[4]Iowa State University. Research Supported in part by NSF grant CCF-0830479.
[5]University at Buffalo.
[6]University of Texas at Brownsville.

1 Introduction

Many natural sets (i.e., positive instances of naturally occurring decision problems) contain redundant information. The membership value of an element is often coordinated with that of others, in the sense that the value can be determined by knowing the membership value of some others. This property, which we call membership redundancy, is manifested in many forms. For example, in SAT, a Boolean formula $\phi(x_1, x_2, \cdots, x_n)$ is satisfiable if and only if at least one of $\phi(1, x_2, \cdots, x_n)$ or $\phi(0, x_2, \cdots, x_n)$ is satisfiable. This property, known as self-reducibility, has played a central role in understanding the structure of the class NP [Mah82, OW91, OKSW94, BKS95, Ogi95, AA96].

Another well-studied form of membership redundancy is random self-reducibility, where the membership value of an instance can be determined from the membership value of randomly chosen others. An example of random self-reducibility is the quadratic residuosity problem. For natural numbers N and $a, 0 \leq a \leq N-1$, a is a quadratic residue modulo N, that is, $X^2 \equiv a \pmod{N}$ has a root, if and only if ar^2 is a quadratic residue modulo N for any randomly chosen r, where the multiplication is taken modulo N. Random self-reducibility has played a crucial role in establishing worst-case to average-case reductions [Lip91, FL92, GS92, CPS99, GRS99, TV02] and designing program checkers and interactive proofs [BK89, LFKN92, Sha92].

In this article our focus is on two of the simplest forms of membership redundancy: autoreducibility and mitoticity. Autoreducibility captures the very notion of membership redundancy: a set A is autoreducible if the membership in A of x can be determined by examining the membership in A of y for any number of y other than x. Trakhtenbrot [Tra70] introduced the notion of autoreducibility in the recursion-theoretic setting. Ambos-Spies [AS84] translated this notion to the polynomial-time setting, and Yao [Yao90] considered the probabilistic polynomial-time setting of this notion, which he called coherence. In this article we mainly consider the deterministic polynomial-time autoreducibility.

By controlling the manner in which an instance of a language can reduce to other instances, we obtain several variants of autoreducibility: many-one autoreducibility, Turing autoreducibility, and truth-table autoreducibility. Given a reducibility $\leq_r$, we want to know whether $\leq_r$-complete sets for a class are $\leq_r$-autoreducible.

Many-one complete sets are many-one autoreducible for almost all natural complexity classes. When we consider Turing complete sets this is provably not the case. For many complexity classes below EXP, such as NP, PSPACE, and EXP itself, their Turing complete sets turn out to be Turing autoreducible, whereas for classes higher than EXP, such as EEXPSPACE, there exist Turing complete sets that are not Turing autoreducible. Resolving the question of whether every Turing complete set is Turing autoreducible for classes in between, such as EXPSPACE and EEXP would lead to interesting separations of complexity classes.

Similarly, we have limited insight about the analogous question with respect to truth-table reducibility. Again, for many complexity classes, resolving whether truth-table complete sets are truth-table autoreducible or not will lead to separations of complexity classes.

For any set A, the set $0A \cup 1A$ is trivially autoreducible. This set contains two disjoint subsets $0A$ and $1A$ that contain the exact same information as the original set $0A \cup 1A$. The notion of mitoticity, originally introduced by Lachlan [Lac67] in the recursion-theoretic setting, captures this stronger form of membership redundancy. Ambos-Spies translated it into the polynomial-time setting as well as introduced the weak version of the notion. A set S is *mitotic* if there is a set A in P such that the sets S, $S_1 = S \cap A$, and $S_2 = S \cap \overline{A}$ are all polynomial-time reducible to each other. A set is S *weakly mitotic* if it can be partitioned into two disjoint sets S_1 and S_2 such that the sets

S, S_1, and S_2 are all polynomial-time reducible to each other. Thus mitoticity requires that the partition of S can be done in polynomial-time. Again, by changing the underlying reduction we obtain the notions of many-one mitoticity, Turing mitoticity, and truth-table mitoticity.

For a reducibility $\leq_r$, it is known that $\leq_r$-mitoticity implies $\leq_r$-autoreducibility [AS84]. The converse does not hold for Turing or truth-table reductions. It is known that there is a truth-table autoreducible set that is not even Turing mitotic [GPSZ08]. However, for the case of many-one reductions, somewhat surprisingly, many-one autoreducibility implies many-one mitoticity. Thus the two notions are equivalent for the case of many-one reductions. From this it immediately follows that for many complexity classes many-one complete sets are many-one mitotic. However, when we consider truth-table or Turing mitoticity we know very little. We do not know of any interesting complexity class whose Turing/truth-table complete sets are Turing/truth-table mitotic.

In addition to be able to capture membership redundancy in the simplest forms, there are other reasons to study the notions of autoreducibility and mitoticity. Buhrman et al. [BFvMT00] showed that while all Turing complete sets are Turing autoreducible, there is a relativized world where not all Turing complete sets for EXP are Turing autoreducible. Thus autoreducibility is a nonrelativizing property in the exponential time realm. Buhrman et al. also suggested to use autoreducibility as a tool to separate complexity classes. Two complexity classes can be separated by showing that all complete sets for one class are autoreducible whereas the other complexity class has complete sets that are not autoreducible. Using this approach they showed that the polynomial-time hierarchy differs from exponential space. Although this separation is known to follow already using diagonalization, its proof is attractive because it separates the two classes by showing that one class has a property that the other class does not have. We refer the reader to the survey by Buhrman and Torenvliet [BT05] for more on this approach.

The study of autoreducibility and mitoticity can deepen our understanding of the Isomorphism Conjecture [BH77]. This conjecture states that every pair of NP-complete sets are polynomial-time isomorphic, i.e., they are reducible to each other by a polynomial-time computable, polynomial-time invertible bijection. For a long time, it has been believed that if one-way functions exist, then the Isomorphism Conjecture is false. However, recently Agrawal and Watanabe [AW09] made some exciting progress that contradicts this belief. They observed that all known one-way functions contain "easy cylinders", i.e., they can be easily inverted on sub-exponentially many strings. They showed that if every one-way function has this property, then the Isomorphism Conjecture holds under nonuniform reductions. Although Goldreich [Gol09] has presented a one-way function that may not have easy cylinders and thus can be a counterexample to the possibility that the Isomorphism Conjecture is proven along this direction, Agrawal and Watanabe's result suggests that we may have to turn around and collect positive evidence for the conjecture.

One way to obtain positive evidence is to consider properties of NP-complete sets that are immediately implied by the Isomorphism Conjecture and prove that NP-complete sets indeed have those properties. Autoreducibility and mitoticity are two such properties. The notions of autoreducibility and mitoticity are invariant under polynomial-time isomorphisms. Since SAT is trivially mitotic, if the Isomorphism Conjecture is true, then all NP-complete sets are mitotic. Thus the unconditional result that all NP-complete sets are many-one mitotic could be taken as (weak) evidence in support of the conjecture.

2 Preliminaries

Let $\Sigma = \{0, 1\}$ be the alphabet. Σ^* denotes the set of all words over Σ, while $\Sigma^{\leq m}$ denotes the words over Σ that are of length $\leq m$. As usual we identify Σ^* with $\mathbb{N}$. For an integer n, $|n|$ denotes the length of n in binary representation. In contrast, the absolute value of n is denoted by $\mathrm{abs}(n)$. We call a set L *nontrivial* if $\|L\| \geq 2$ and $\|\overline{L}\| \geq 2$.

Definition 2.1 ([AS84]) *A set A is* polynomial-time Turing autoreducible *if there exists a polynomial-time-bounded oracle Turing machine M such that $A = L(M^A)$ and for all x, M on input x never queries x. If the oracle Turing machine M makes nonadaptive queries, then A is polynomial-time truth-table autoreducible.*

A set A is polynomial-time many-one autoreducible *if $A \leq^{\mathrm{p}}_{\mathrm{m}} A$ via a polynomial-time computable reduction function f such that for all x, $f(x) \neq x$.*

Given a reducibility $\leq_r$, two sets A and B are $\leq_r$-equivalent if $A \leq_r B$ and $B \leq_r A$.

Definition 2.2 ([AS84]) *Given a polynomial-time reducibility $\leq_r$, a set A is* polynomial-time $\leq_r$-mitotic *if there exists a set $B \in \mathrm{P}$ such that A, $A \cap B$ and $A \cap \overline{B}$ are $\leq_r$ equivalent to each other.*

All reductions considered in this paper are polynomial-time computable. Therefore, for simplicity we sometimes leave out the modifier "polynomial-time" and simply write many-one reduction, Turing reduction, many-one autoreducible, Turing mitotic and so on.

3 Autoreducibility

3.1 Many-One Autoreducibility

Berman [Ber77] showed that all many-one complete sets for EXP are complete via length-increasing reductions, thus all many-one complete sets for EXP are many-one autoreducible. All many-one complete sets for NEXP are complete via one-one reductions [GH92]. Let A be any NEXP-complete set. There is a one-one reduction f from $A \times \{0, 1\}$ to A. Since f is one-one, at least one of $f(\langle x, 0\rangle)$ and $f(\langle x, 1\rangle)$ must differ from x. Thus all NEXP-complete sets are many-one autoreducible. Berman's result can be extended to all deterministic classes that are above EXP, and Ganesan and Homer's result can be extended to all nondeterministic classes above NEXP. Thus many-one complete sets for all these classes are many-one autoreducible.

Theorem 3.1 *If $\mathcal{C}$ is any deterministic (nondeterministic) complexity class that contains* EXP *(*NEXP*), then all nontrivial $\leq^{\mathrm{p}}_{\mathrm{m}}$-complete sets for $\mathcal{C}$ are $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreducible.*

When we consider classes below EXP, such as NP and PSPACE, we do not know whether many-one complete sets for these classes are complete via length-increasing or via one-one reductions. Thus we need different ideas.

Theorem 3.2 ([GOP$^+$07]) *All nontrivial $\leq^{\mathrm{p}}_{\mathrm{m}}$-complete sets for* NP *are $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreducible.*

Proof The proof uses left-sets of Ogiwara and Watanabe [OW91]. Let L be any many-one complete set for NP and let $R(.,.)$ be a polynomial-time verifiable relation for L. Without loss of generality, assume that every x in L has a witness of length exactly $p(|x|)$ for some polynomial p. Consider the following language L':

$$L' = \{\langle x, y\rangle \mid |y| = p(|x|), \exists w\ R(x, w),\ \text{and } y \leq w\}.$$

Since L' is in NP, there is a many-one reduction f from L' to L. Our proof relies on the following simple properties that relate L to L'. Consider any string x of length n, let $p(n) = m$.

1. x is in L if and only if $\langle x, 0^m\rangle$ is in L'. Thus x is in L if and only if $f(\langle x, 0^m\rangle)$ is in L.
2. Suppose $f(\langle x, 1^m\rangle) = x$. Then x is in L if and only if 1^m is a witness of x. Thus we can decide the membership of x in L in polynomial-time.
3. If a string z is not a witness of x, then $\langle x, z\rangle$ is in L' if and only if $\langle x, z+1\rangle$ is in L'.

Since L is nontrivial there exist strings a_1 and a_2 that are in L and strings b_1 and b_2 that are not in L.

Given an input string x of length n, the autoreduction works as follows. If $f(\langle x, 0^m\rangle) \neq x$, then output $f(\langle x, 0^m\rangle))$. Property 1 ensures that the reduction is correct in this case. Else, if $f(\langle x, 1^m\rangle) = x$, then by checking whether 1^m is a witness of x, decides the membership of x in L. Based on whether $x \in L$ or not, output an appropriate fixed string from $\{a_1, a_2, b_1, b_2\}$ that is different from x. Property 2 ensures that the reduction is correct in this case.

If $f(\langle x, 0^m\rangle) = x$, and $f(\langle x, 1^m\rangle) \neq x$, then there exists a string z such that $f(\langle x, z\rangle) = x$ and $f(\langle x, z+1\rangle) \neq x$. Such a string z can be found in polynomial-time via binary search. If z is a witness of x, then x is in L, and the autoreduction outputs a fixed string from $\{a_1, a_2\}$ that is different from x.

Suppose z is not a witness of x. By Property 3, we have $\langle x, z\rangle \in L'$ if and only if $\langle x, z+1\rangle \in L'$. Since $f(\langle x, z\rangle) = x$, it follows that x is in L if and only if $f(\langle x, z+1\rangle)$ is in L. Since $f(\langle x, z+1\rangle) \neq x$, the autoreduction outputs $f(\langle x, z+1\rangle)$. □

The above proof makes crucial use of the ability to define left-sets within NP. We can use Cai and Furst's [CF91] bottleneck characterization of PSPACE to define left-set like languages and establish many-one autoreducibility of PSPACE-complete sets. This characterization states that for any language L in PSPACE, there is a polynomial p and a polynomial-time computable function f from $\Sigma^* \times \Sigma^*$ to S_5 such that for any x, $x \in L$ if and only if

$$f(x, 1^m) \circ \cdots \circ f(x, 0^m) = I_5,$$

where $\circ$ is the product of the permutations calculated from left to right, I_5 is the identity permutation, and $p(|x|) = m$.

Let L be a many-one complete language for PSPACE and suppose that the membership of L in PSPACE is characterized by the polynomial p and the function f. Define an intermediate language L' as follows.

$$L' = \{\langle x, y, \pi\rangle \mid |y| = p(|x|), y \in S_5,\ \text{and } f(x, 1^{p(|x|)}) \circ \cdots \circ f(x, y) = \pi\}$$

Since L' is in PSPACE, there is a many-one reduction g from L' to L. Similar to the case of NP, the language L' has the following useful properties. Let x be a string of length n and $p(n) = m$.

1. x is in L if and only if $\langle x, 0^m, I_5\rangle$ is in L'. So x is in L if and only if $g(\langle x, 0^m, I_5\rangle)$ is in L.

2. Suppose for some $\pi \in S_5$, $g(\langle x, 1^m, \pi\rangle = x$. This means that x is in L if and only if $f(x, 1^m) = \pi$. Thus we can decide the membership of x in L in polynomial-time.

3. Let y be string and γ be a permutation. Let ρ be the unique inverse of the permutation $f(x, y)$. Let $\gamma' = \gamma \circ \rho$. Now, $f(x, 1^m) \circ \cdots \circ f(x, y) = \gamma$ if and only if $f(x, 1^m) \circ \cdots \circ f(x, y+1) = \gamma'$. Thus $\langle x, y, \gamma\rangle$ is in L' if and only if $\langle x, y+1, \gamma'\rangle$ is in L'. Moreover γ' can be computed in polynomial-time from x, y and γ.

We can use these properties to design a many-one autoreduction for L. If $g(\langle x, 0^m, I_5\rangle) = z \neq x$, then the autoreduction outputs z. If for some $\pi \in S_5$, $g(\langle x, 1^m, \pi\rangle = x$, then decide the membership of x in L, and output an appropriate string that is different from x from a set of fixed members and nonmembers. If both cases are not true, then there exists a string y such that for some $\pi \in S_5$, $f(\langle x, y, \pi\rangle) = x$, and for every $\pi \in S_5$, $f(\langle x, y+1, \pi\rangle) \neq x$. Such a string y and permutation π can be found using binary search. Now we can use the third property to output a string z whose membership in L is the same as the membership of x in L. Thus we have the following theorem.

Theorem 3.3 **([GOP⁺07])** *All nontrivial $\leq_{\mathrm{m}}^{\mathrm{p}}$-complete sets for* PSPACE *are $\leq_{\mathrm{m}}^{\mathrm{p}}$-autoreducible.*

These theorems can be generalized to several interesting classes that lie between NP and PSPACE. Again, this is done by using an appropriate characterization of these classes that enable us to define intermediate languages that look like left-sets.

A language A is *polynomial-time bit-reducible* to a language B, if there exist polynomial-time computable functions $f : \Sigma^* \times \mathbb{N} \to \Sigma$ and $g : \Sigma^* \to \mathbb{N}$ such that for every x, x is in A if and only if the string $f(x, 1)f(x, 2) \cdots f(x, g(x))$ is in B.

Let R be a regular language and let $\mathcal{C}$ be the class of languages that are polynomial-time bit-reducible to R. Let L be a many-one complete language for $\mathcal{C}$. As in the case of NP and PSPACE, we can define an appropriate "left-set like" language L'. This enable us to show the many-one autoreducibility of L.

It is known that all Σ_k, Π_k, Δ_k levels of the polynomial-time hierarchy are polynomial-time bit-reducible to appropriate regular languages [BSS99, HLS⁺93, Tra02, BLS⁺04]. Thus many-one complete sets for all of these classes are many-one autoreducible.

Theorem 3.4 **([GOP⁺07])** *All nontrivial $\leq_{\mathrm{m}}^{\mathrm{p}}$-complete sets of the following classes are $\leq_{\mathrm{m}}^{\mathrm{p}}$-autoreducible: Σ_k^{P}, Π_k^{P}, and $\Delta_{\mathrm{k+1}}^{\mathrm{P}}$ for $k \geq 1$.*

3.2 Turing Autoreducibility

Now we consider Turing complete sets. Beigel and Feigenbaum [BF92] showed that for most of the classes that lie below PSPACE, Turing complete sets turn out to be Turing autoreducible. Below we provide the proof for NP.

Theorem 3.5 *All $\leq_{\mathrm{T}}^{\mathrm{p}}$-complete sets for* NP *are $\leq_{\mathrm{T}}^{\mathrm{p}}$-autoreducible.*

Proof Let L be any Turing complete set for NP. Let f be a many-one reduction from L to SAT and M be a polynomial-time Turing machine such that $\mathrm{SAT} = M^L$.

Let x be an input string. Compute $f(x)$ to obtain a Boolean formula $\phi(y_1, \cdots, y_m)$. Knowing the membership of $\phi(y_1, \cdots, y_m)$ in SAT tells us the membership of x in L. Now the autoreduction works as follows.

Simulate the oracle machine M on input $\phi(0, y_2, \cdots, y_m)$ with L as oracle. When M makes the query x, split the simulation into two paths with one path proceeding with answer "Yes" and the other path proceeding with answer "No". If both paths accept or both paths reject, or if the machine never queries x, then we know whether $\phi(0, y_2, \cdots, y_m)$ is satisfiable or not. If it is satisfiable, then x is in L and so the autoreduction terminates. If $\phi(0, y_2, \cdots, y_m)$ is not in SAT, then do a similar simulation for $\phi(1, y_2, \cdots, y_m)$.

The above process fails to determine the membership of x in L when then there is a $b \in \{0, 1\}$ such that while simulating M on input $\phi(b, y_2, \cdots, y_m)$, it makes the query x, and one path accepts and the other path rejects. From this we will arrive at one of the following conclusions: $x \in L \Leftrightarrow \phi(b, y_2, \cdots y_m) \in \text{SAT}$ or $x \in L \Leftrightarrow \phi(b, y_2, \cdots, y_m) \notin \text{SAT}$. In either case, knowing the membership of $\phi(b, y_2, \cdots, y_m)$ in SAT tells the membership of x in L.

Observe that we are able to produce the formula $\phi(b, y_2, \cdots, y_m)$ by running the machine M with L as oracle without querying x. The formula $\phi(b, y_2, \cdots, y_m)$ has one variable less than the original formula $\phi(y_1, \cdots, y_m)$. So we have made progress. By repeating the above process, we can determine the membership of x in L without ever querying x. This shows that L is Turing autoreducible. $\square$

The above proof relies on the fact that SAT is length-decreasing self-reducible. This idea can be applied to any complexity class that has a complete language that is length-decreasing self-reducible. This gives the following result.

Theorem 3.6 ([BF92]) *All $\leq^{\text{p}}_{\text{T}}$-complete sets of the following classes are $\leq^{\text{p}}_{\text{T}}$-autoreducible:* NP, PSPACE, $\Sigma^{\text{P}}_{\text{i}}$, $\Pi^{\text{P}}_{\text{i}}$, *and* $\Delta^{\text{P}}_{\text{i+1}}$ *for* $i \geq 1$.

Since every self-reducible language belongs to PSPACE, the above proof idea does not work for classes such as EXP. Nevertheless Buhrman et al. showed that all Turing complete sets for EXP are Turing autoreducible.

Theorem 3.7 ([BFvMT00]) *All $\leq^{\text{p}}_{\text{T}}$-complete sets for* EXP *are $\leq^{\text{p}}_{\text{T}}$-autoreducible.*

Proof Let L be a $\leq^{\text{p}}_{\text{T}}$-complete set for EXP that is accepted by a Turing machine M with running time $2^{p(n)}$. The following set belongs to EXP.

$$L_1 \overset{df}{=} \{\langle x, i, j\rangle | \text{ at the } i\text{-th step of } M \text{ on } x, \text{ the } j\text{-th cell of the Turing tape contains a } 1\}$$

So there exists a polynomial-time oracle machine R such that $L_1 \leq^{\text{p}}_{\text{T}} L$ via R.

$$L_2 \overset{df}{=} \{\langle x, a, b\rangle | \exists i \leq a\ \exists j \leq b \text{ such that } R^{L-\{x\}}(x, i, j) \text{ and } R^{L\cup\{x\}}(x, i, j) \text{ have different outcomes}\}$$

Observe that $L_2 \in \text{EXP}$, since $L \in \text{EXP}$. Hence $L_2 \leq^{\text{p}}_{\text{T}} L$. The following algorithm is a $\leq^{\text{p}}_{\text{T}}$-autoreduction for L where x is the input.

```
1  Determine the smallest i and j such that $R^{L-\{x\}}(x,i,j)$ and $R^{L\cup\{x\}}(x,i,j)$ have
   different outcomes.
2  If such i,j do not exist, then output the result of $R^{L-\{x\}}(x, 2^{p(|x|)}, 1)$ and stop.
3  Determine the content of the cells $j-1$, j, and $j+1$ in step $i-1$ of M(x).
4  If the outcome of $R^{L-\{x\}}(x,i,j)$ is inconsistent with the content of the cells
   in step $i-1$, then accept else reject.
```

We argue that step 1 is possible in polynomial time by asking queries different from x: In this step it suffices to determine the smallest i and j such that $\langle x, i, j\rangle \in L_2$. This is possible in polynomial time by asking binary-search queries to L_2. It suffices to ask queries to L, since $L_2 \leq_{\mathrm{T}}^{\mathrm{p}} L$. Let M' be the machine that determines i and j by asking queries to L. We simulate M' once with oracle $L - \{x\}$ and once with oracle $L \cup \{x\}$. One of the simulations yields the correct values for i and j. We can determine the correct values by verifying that $R^{L-\{x\}}(x, i, j)$ and $R^{L\cup\{x\}}(x, i, j)$ have different outcomes. (If the values of both simulations pass this test, then the smaller values are the correct ones.)

If i and j do not exist, then in particular $R^{L-\{x\}}(x, 2^{p(|x|)}, 1)$ and $R^{L\cup\{x\}}(x, 2^{p(|x|)}, 1)$ have the same outcome. So in step 2 we output the result of $R^{L}(x, 2^{p(|x|)}, 1)$. This means that we accept if and only if after $2^{p(|x|)}$ steps of the computation $M(x)$, the first cell of the Turing tape contains 1. Hence we accept in step 2 if and only if $M(x)$ accepts.

Step 3 is done by simulating $R^{L-\{x\}}(x, i-1, j-1)$, $R^{L-\{x\}}(x, i-1, j)$ and $R^{L-\{x\}}(x, i-1, j+1)$. This yields the right results, since by the minimal choice of i and j, these computations have the same outcome for both oracles, $L - \{x\}$ and $L \cup \{x\}$.

In step 4, exactly one of the outcomes of $R^{L-\{x\}}(x, i, j)$ and $R^{L\cup\{x\}}(x, i, j)$ is inconsistent with the cells $j-1$, j, and $j+1$ in step $i-1$. If $R^{L-\{x\}}(x, i, j)$ is inconsistent, then $L - \{x\} \neq L$ and hence $x \in L$. Otherwise $L \cup \{x\} \neq L$ and hence $x \notin L$. $\square$

This theorem can be generalized to show that all Turing complete sets of $\Delta_{k+1}^{\mathrm{EXP}}$-levels of the exponential-time hierarchy are Turing autoreducible [BFvMT00]. Interestingly, the above theorem does not relativize. This is because an exponential-time machine can query about strings that a polynomial-time autoreduction cannot.

Theorem 3.8 ([BFvMT00]) *There is an oracle relative to which* EXP *has a* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete set that is not* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible.*

As opposed to the case of many-one reductions, there are complexity classes whose Turing complete sets are not Turing autoreducible.

Theorem 3.9 ([BFvMT00]) EEXPSPACE *has a* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete set that is not* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible.*

When we consider intermediate classes such as EEXP, EXPSPACE, and NEXP we do not know whether Turing complete sets for these classes are Turing autoreducible. Resolving this question for the former two complexity classes yields separation results.

Theorem 3.10 ([BFvMT00])

- *If all* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete sets for* EEXP *are* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible, then* $\mathrm{NL} \neq \mathrm{NP}$ *and* $\mathrm{P} \neq \mathrm{PSPACE}$*.*
- *If not all* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete sets for* EEXP *are* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible, then* $\mathrm{PH} \neq \mathrm{EXP}$*.*
- *If all* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete sets for* EXPSPACE *are* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible, then* $\mathrm{NL} \neq \mathrm{NP}$*.*
- *If not all* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-complete sets for* EXPSPACE *are* $\leq_{\mathrm{T}}^{\mathrm{p}}$*-autoreducible, then* $\mathrm{PH} \neq \mathrm{PSPACE}$*.*

If there exist Turing complete sets for NEXP that are not Turing autoreducible, then EXP differs from NEXP. However, we do not know if proving all Turing complete sets for NEXP are Turing autoreducible yields any separation result. Buhrman et al. showed that all Turing complete sets for NEXP are nonuniform Turing autoreducible. Thus it is conceivable that all Turing complete sets for NEXP are Turing autoreducible. Proving this might be well within our reach.

3.3 Truth-Table Autoreducibility

For most of the complexity classes, we know whether many-one/Turing complete sets are many-one/Turing autoreducible or not. However, when we consider truth-table complete sets, our knowledge is limited. It turns out that, for many classes, resolving the question of whether truth-table complete sets are truth-table autoreducible or not will lead to separations of complexity classes.

When we consider truth-table reductions that make only one query, the proofs of Section 3.1 can be easily modified to obtain the following result.

Theorem 3.11 ([GOP$^+$07]) *All $\leq^{\mathrm{p}}_{1\text{-tt}}$-complete sets of the following classes are $\leq^{\mathrm{p}}_{1\text{-tt}}$-autoreducible:* NP*,* PSPACE*,* $\Sigma^{\mathrm{P}}_{\mathrm{i}}$*,* $\Pi^{\mathrm{P}}_{\mathrm{i}}$*, and* $\Delta^{\mathrm{P}}_{\mathrm{i+1}}$ *for* $i \geq 1$*,* EXP*, and* NEXP*.*

With respect to slightly more powerful reducibilities we lose the autoreducibility of EXP-complete sets: Buhrman at al. [BFvMT00] show that for $k \geq 3$ there exist $\leq^{\mathrm{p}}_{k\text{-tt}}$-complete sets in EXP that are not $\leq^{\mathrm{p}}_{k\text{-tt}}$-autoreducible. When we consider general truth-table reductions our knowledge is limited. Using ideas that are similar to the proof of Theorem 3.7, Buhrman et al. showed the following result.

Theorem 3.12 ([BFvMT00]) *All $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets in $\Delta^{\mathrm{P}}_{\mathrm{i+1}}$ for $i \geq 1$ are $\leq^{\mathrm{p}}_{\mathrm{tt}}$-autoreducible.*

For classes, NP and EXP, truth-table complete sets turn out to be probabilistic truth-table autoreducible.

Theorem 3.13 ([BFvMT00]) *All $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets in* NP *and* EXP *are probabilistically polynomial-time truth-table autoreducible.*

The proof for NP uses the witness isolation theorem of Valiant and Vazirani [VV86]. The proof for EXP uses the PCP characterization of EXP [ALM$^+$92]. Using the interactive proofs characterization of PSPACE, one can also show that truth-table complete sets for PSPACE are probabilistic truth-table autoreducible. Making these autoreductions deterministic remains a challenge. Achieving this for PSPACE and EXP yields separation results.

Theorem 3.14 ([BFvMT00])

- *If all $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets for* PSPACE *are $\leq^{\mathrm{p}}_{\mathrm{tt}}$-autoreducible, then* NL $\neq$ NP*.*
- *If not all $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets for* PSPACE *are $\leq^{\mathrm{p}}_{\mathrm{tt}}$-autoreducible, then* PH $\neq$ PSPACE*.*
- *If all $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets for* EXP *are $\leq^{\mathrm{p}}_{\mathrm{tt}}$-autoreducible, then* P $\neq$ PSPACE*.*
- *If not all $\leq^{\mathrm{p}}_{\mathrm{tt}}$-complete sets for* EXP *are $\leq^{\mathrm{p}}_{\mathrm{tt}}$-autoreducible, then* PH $\neq$ EXP*.*

4 Mitoticity

In this section we want to learn more about the relationship between autoreducibility and mitoticity. For arbitrary polynomial-time reducibilities it holds that mitoticity implies autoreducibility. The converse implication can be disproved for all reducibilities between polynomial-time 2-truth-table and polynomial-time Turing reducibility.

In contrast, polynomial-time many-one autoreducibility and polynomial-time many-one mitoticity are equivalent, which will be the central result of this section. First we quickly prove that mitoticity implies autoreducibility. Then we show that a restriction of polynomial-time many-one autoreducibility translates to polynomial-time many-one mitoticity. The idea of this proof can be technically extended to general polynomial-time many-one autoreducibility, but for the sake of clarity we skip this extension. Finally, with the equivalence at hand, we obtain several results about the mitoticity of complete sets.

Theorem 4.1 ([AS84]) *If a nontrivial set L is $\leq^{\mathrm{p}}_{\mathrm{m}}$-mitotic, then it is $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreducible.*

Proof If L is $\leq^{\mathrm{p}}_{\mathrm{m}}$-mitotic, then there exist $S \in \mathrm{P}$ and polynomial-time computable functions f_1, f_2 such that $L \cap S \leq^{\mathrm{p}}_{\mathrm{m}} L \cap \overline{S}$ via f_1 and $L \cap \overline{S} \leq^{\mathrm{p}}_{\mathrm{m}} L \cap S$ via f_2. By assumption, there exist different words $v, w \in \overline{L}$. The following function is a $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreduction for L.

$$f'(x) \overset{df}{=} \begin{cases} f_1(x) & : \text{ if } x \in S \text{ and } f_1(x) \notin S \\ f_2(x) & : \text{ if } x \notin S \text{ and } f_2(x) \in S \\ \min(\{v, w\} - \{x\}) & : \text{ otherwise} \end{cases}$$

□

This result actually holds for arbitrary reducibilities.

Corollary 4.2 ([AS84]) *For every nontrivial set L and every $\leq \in \{\leq^{\mathrm{p}}_{\mathrm{m}}, \leq^{\mathrm{p}}_{\mathrm{tt}}, \leq^{\mathrm{p}}_{\mathrm{T}}\}$,*

$$L \text{ is } \leq\text{-mitotic} \quad \Rightarrow \quad L \text{ is } \leq\text{-autoreducible.}$$

Now we want to understand the more difficult direction, i.e., that $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreducibility implies $\leq^{\mathrm{p}}_{\mathrm{m}}$-mitoticity. In order to make the approach comprehensible, we prove the weaker result where we additionally assume that the $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreduction is made by a function that does not increase the length. The presented technique can then be technically extended so that it works for general $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreductions. We will omit the details of this extension, but will describe its rough idea.

Let sgn() denote the signum function, and let log() denote the logarithm with base 2, where log(0) is defined as 0. We use the following polynomial-time computable distance function.

$$d(x, y) \overset{df}{=} \mathrm{sgn}(y - x) \cdot \lfloor \log(\mathrm{abs}(y - x)) \rfloor$$

The following two propositions are immediate consequences of this definition.

Proposition 4.3 *For all $m \in \mathbb{N}$ and all $x, y \in \Sigma^{\leq m}$ it holds that $-m \leq d(x, y) \leq m$.*

Proposition 4.4 *For all $x, y \in \mathbb{N}$ it holds that:*

$$\begin{aligned} d(x, y) = 0 \quad &\Rightarrow \quad y - x \in \{-1, 0, 1\} \\ d(x, y) = r > 0 \quad &\Rightarrow \quad 2^r \leq y - x < 2^{r+1} \\ d(x, y) = r < 0 \quad &\Rightarrow \quad 2^{-r} \leq x - y < 2^{-r+1} \end{aligned}$$

Proposition 4.3 has an important consequence: If we have a sufficiently large list of words from $\Sigma^{\leq m}$ and if we compare the distances between neighboring words, then the distances are a non-monotone sequence of numbers or we find three places in a row where the distances are the same. The following proposition makes this precise.

Proposition 4.5 *Let $n \geq 1$ and $x_0, \dots, x_{6n+4} \in \Sigma^{\leq n}$. Then at least one of the following holds:*

1. *There exist $i, j \in [0, 6n+2]$ such that $d(x_i, x_{i+1}) < d(x_{i+1}, x_{i+2})$ and $d(x_j, x_{j+1}) > d(x_{j+1}, x_{j+2})$.*
2. *There exists $j \in [0, 6n]$ such that $d(x_j, x_{j+1}) = d(x_{j+1}, x_{j+2}) = d(x_{j+2}, x_{j+3}) = d(x_{j+3}, x_{j+4})$.*

Proof Assume that the first statement does not hold. Without loss of generality we may assume $d(x_0, x_1) \leq d(x_1, x_2) \leq \cdots \leq d(x_{6n+3}, x_{6n+4})$. By Proposition 4.3, we have $-n \leq d(x_0, x_1)$ and $d(x_{6n+3}, x_{6n+4}) \leq n$. Since the values $d(\cdot, \cdot)$ are natural numbers, there are at most $2n$ elements $i \in [0, 6n+2]$ such that $d(x_i, x_{i+1}) < d(x_{i+1}, x_{i+2})$. For the remaining elements $j \in [0, 6n+2]$ it holds that $d(x_j, x_{j+1}) = d(x_{j+1}, x_{j+2})$. There are at least $4n+3$ such elements j and they distribute over at most $2n+1$ parts that are separated by the elements i. By the pigeon-hole principle, there exists one part that contains at least 3 elements j. So the second statement holds. □

Theorem 4.6 *Let L be a nontrivial set. If L has an $\leq_{\mathrm{m}}^{\mathrm{p}}$-autoreduction f such that $|f(x)| \leq |x|$ for all x, then L is $\leq_{\mathrm{m}}^{\mathrm{p}}$-mitotic.*

Proof Note that for all x, $f(x) \neq x$, $|f(x)| \leq |x|$, and $(x \in L \Leftrightarrow f(x) \in L)$. We construct a total, polynomial-time-computable function g and a set $S \in \mathrm{P}$ such that for all x,

$$x \in L \Leftrightarrow g(x) \in L \tag{1}$$

and

$$x \in S \Leftrightarrow g(x) \notin S. \tag{2}$$

With help of these equivalences one easily verifies that $L \cap S \equiv_{\mathrm{m}}^{\mathrm{p}} L \cap \overline{S} \equiv_{\mathrm{m}}^{\mathrm{p}} L$. Hence L is $\leq_{\mathrm{m}}^{\mathrm{p}}$-mitotic. So it remains to construct g and S, and to prove (1) and (2).

The set S is defined by the following polynomial-time algorithm.

```
0   // decision algorithm for the set S; works on input x
1   y := f(x), z := f(f(x))
2   if x = z then (if x > f(x) then accept else reject)
3   // here x, y, and z are pairwise different
4   if d(x,y) > d(y,z) then reject
5   if d(x,y) < d(y,z) then accept
6   r := d(x,y)
7   if ⌊y/2^{abs(r)+1}⌋ is even then accept else reject
```

S has the following property: If we start with an arbitrary word x and follow its trajectory $x, f(x), f(f(x)), \dots$, then after at most $6|x|+3$ steps we find an $f^i(x)$ such that $x \in S \Leftrightarrow f^i(x) \notin S$.

Claim 4.7 *For every $x \in \Sigma^*$ there exists some $i \in [0, 6|x|+4]$ such that*

$$x \in S \Leftrightarrow f^i(x) \notin S.$$

For the moment we postpone the proof of this claim and continue with the proof of the theorem. The function g is defined by the following algorithm.

```
0   // algorithm for the function g; works on input x
1   determine an i ∈ [1, 6|x| + 4] such that x ∈ S ⇔ f^i(x) ∉ S
2   return f^i(x)
```

Note that in this algorithm, the values $f^i(x)$ can be computed in polynomial time, since f does not increase the length. By Claim 4.7, the i in line 1 exists and hence g is a total, polynomial-time-computable function. From the definition of g, (2) immediately follows. Moreover, (1) holds, because $g(x) = f^i(x)$ and f is an autoreduction for L. This finishes the proof of the theorem.

Proof of Claim 4.7 Assume that the claim does not hold. So there is some $x \in \Sigma^*$ such that

$$\forall i \in [0, 6|x| + 4],\ x \in S \Leftrightarrow f^i(x) \in S.$$

Let $n \overset{df}{=} |x|$ and let $x_i \overset{df}{=} f^i(x)$ for $i \in [0, 6n + 4]$. Without loss of generality we assume $x_i \in S$ for all $i \in [0, 6n + 4]$.

Consider the algorithm for S on input x_i where $i \in [0, 6n + 2]$ and let us analyze in which line the algorithm stops. It cannot stop in line 4, since by assumption $x_i \in S$. We show that it cannot stop in line 2: Assume that there exists some $i \in [0, 6n + 2]$ such that the algorithm on input x_i stops in line 2. So $x_i = f(f(x_i))$ and hence $x_{i+1} = f(f(x_{i+1}))$. Therefore, on both inputs, x_i and x_{i+1}, the algorithm stops in line 2. Note that $x_i \neq x_{i+1}$, since f is an $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreduction. Hence by line 2, $x_i \in S \Leftrightarrow x_{i+1} \notin S$, which contradicts our assumption. So we have shown:

For $i \in [0, 6n + 2]$, the algorithm for S on input x_i stops either in line 5 or in line 7. (3)

It follows that

$$\forall i \in [0, 6n + 2],\ d(x_i, x_{i+1}) \leq d(x_{i+1}, x_{i+2}), \tag{4}$$

since otherwise the algorithm stops in line 4. We apply Lemma 4.5 to the list $x_0, x_1, \ldots, x_{6n+4}$. By (4), the first statement of the lemma cannot hold, and hence there exists $i \in [0, 6n]$ such that

$$d(x_i, x_{i+1}) = d(x_{i+1}, x_{i+2}) = d(x_{i+2}, x_{i+3}) = d(x_{i+3}, x_{i+4}). \tag{5}$$

Hence on input of the words x_i, x_{i+1}, x_{i+2}, the algorithm stops in line 7. Let $r \overset{df}{=} d(x_i, x_{i+1})$ and recall that $x_i, x_{i+1}, x_{i+2} \in S$. Therefore, the following holds:

$$a_1 \overset{df}{=} \lfloor x_{i+1}/2^{\mathrm{abs}(r)+1} \rfloor \text{ is even} \tag{6}$$
$$a_2 \overset{df}{=} \lfloor x_{i+2}/2^{\mathrm{abs}(r)+1} \rfloor \text{ is even} \tag{7}$$
$$a_3 \overset{df}{=} \lfloor x_{i+3}/2^{\mathrm{abs}(r)+1} \rfloor \text{ is even} \tag{8}$$

We show that this implies a contradiction.

Case 1: $r = 0$. Since f is an autoreduction, $x_{i+1} \neq x_{i+2}$ and $x_{i+2} \neq x_{i+3}$. By Proposition 4.4, $x_{i+1} - x_{i+2}$ and $x_{i+2} - x_{i+3}$ are in $\{-1, 1\}$. Both values are the same, since otherwise $x_{i+1} = x_{i+3}$ and the algorithm on x_{i+1} stops in line 2. So $x_{i+1} = x_{i+2} - 1 = x_{i+3} - 2$ or $x_{i+3} = x_{i+2} - 1 = x_{i+1} - 2$. Hence $x_{i+3} - x_{i+1} \in \{-2, 2\}$ and $a_3 - a_1 \in \{-1, 1\}$. This contradicts (6) and (8).

Case 2: $r \neq 0$. Without loss of generality we assume $r > 0$. So $x_i < x_{i+1} < x_{i+2} < x_{i+3}$ and $a_1 \leq a_2 \leq a_3$. By Proposition 4.4, the values $x_{i+2} - x_{i+1}$ and $x_{i+3} - x_{i+2}$ are in $[2^r, 2^{r+1})$. Hence $a_2 - a_1 \leq 1$ and $a_3 - a_2 \leq 1$. Moreover, $x_{i+3} - x_{i+1} \geq 2^{r+1}$ and $a_3 \geq a_1 + 1$. The latter implies $a_3 - a_1 \geq 2$, since a_1 and a_3 are even. Since a_2 is even as well, we obtain $a_2 - a_1 \geq 2$ or $a_3 - a_2 \geq 2$. This is a contradiction.

This proves Claim 4.7 and finishes the proof of Theorem 4.6. □

A technical extension of the proof of Theorem 4.6 yields the central result of this section.

Theorem 4.8 ([GPSZ08]) *A nontrivial set is $\leq^{\mathrm{p}}_{\mathrm{m}}$-mitotic if and only if it is $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreducible.*

Proof The implication from left to right holds by Theorem 4.1. For the other direction one has to extend the proof of Theorem 4.6 so that it works for arbitrary $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreductions f. This extension makes the proof technically more difficult, but does not contain new ideas. Therefore, we omit this extension and refer to [GPSZ08] for the detailed version.

The rough idea of the extension is as follows: The algorithm for S has to be modified such that if the autoreduction increases the length, then the input is accepted/rejected according to their length. By doing so we can ensure that if the length of words in the trajectory $x, f(x), f(f(x)), \ldots$ increases rapidly, then after a constant number of such increases one reaches a length where the algorithm for S flips its acceptance behavior. This results in a small i such that $x \in S \Leftrightarrow f^i(x) \notin S$, which generalizes Claim 4.7 to arbitrary $\leq^{\mathrm{p}}_{\mathrm{m}}$-autoreductions. □

The proof above also goes through for $\leq^{\mathrm{p}}_{1\text{-tt}}$-reducibility.

Theorem 4.9 ([GPSZ08]) *A nontrivial set is $\leq^{\mathrm{p}}_{1\text{-tt}}$-mitotic if and only if it is $\leq^{\mathrm{p}}_{1\text{-tt}}$-autoreducible.*

However, autoreducibility and mitoticity differ with respect to more powerful reducibilities.

Theorem 4.10 ([AS84, GPSZ08, GSTZ09]) *There exist $L_1, L_2 \in \mathrm{SPARSE} \cap \mathrm{EXP}$ such that:*

- *L_1 is $\leq^{\mathrm{p}}_{2\text{-tt}}$-autoreducible, but not $\leq^{\mathrm{p}}_{2\text{-tt}}$-mitotic*
- *L_1 is $\leq^{\mathrm{p}}_{3\text{-tt}}$-autoreducible, but not $\leq^{\mathrm{p}}_{\mathrm{T}}$-mitotic*

Hence autoreducibility and mitoticity are inequivalent for all reducibilities between $\leq^{\mathrm{p}}_{2\text{-tt}}$ and $\leq^{\mathrm{p}}_{\mathrm{T}}$.

The Theorems 4.8 and 4.9 allow us to reformulate several autoreducibility results in terms of mitoticity.

Corollary 4.11 *1. All nontrivial $\leq^{\mathrm{p}}_{\mathrm{m}}$-complete sets of the following classes are $\leq^{\mathrm{p}}_{\mathrm{m}}$-mitotic:* NP, PSPACE, EXP, NEXP, $\Sigma^{\mathrm{P}}_{\mathrm{i}}$, $\Pi^{\mathrm{P}}_{\mathrm{i}}$, *and* $\Delta^{\mathrm{P}}_{\mathrm{i+1}}$ *for $i \geq 1$.*

2. All nontrivial $\leq^{\mathrm{p}}_{1\text{-tt}}$-complete sets of the following classes are $\leq^{\mathrm{p}}_{1\text{-tt}}$-mitotic: NP, PSPACE, $\Sigma^{\mathrm{P}}_{\mathrm{i}}$, $\Pi^{\mathrm{P}}_{\mathrm{i}}$, *and* $\Delta^{\mathrm{P}}_{\mathrm{i+1}}$ *for $i \geq 1$.*

In contrast to the results in Corollary 4.11, with respect to slightly more powerful reducibilities it might well be that complete sets of complexity classes are not mitotic. For $k \geq 3$ there exist $\leq^{\mathrm{p}}_{k\text{-tt}}$-complete sets in EXP that are not $\leq^{\mathrm{p}}_{k\text{-tt}}$-autoreducible and hence not $\leq^{\mathrm{p}}_{k\text{-tt}}$-mitotic [BFvMT00].

Autoreducibility

	$\leq^p_m$	$\leq^p_{1\text{-tt}}$	$\leq^p_{tt}$	$\leq^p_T$
NP	yes	yes	open	yes
Σ^P_i	yes	yes	open	yes
Π^P_i	yes	yes	open	yes
Δ^P_i	yes	yes	yes	yes
PSPACE	yes	yes	open[7]	yes
EXP	yes	yes	open[7]	yes
NEXP	yes	yes	open	open

Mitoticity

	$\leq^p_m$	$\leq^p_{1\text{-tt}}$	$\leq^p_{tt}$	$\leq^p_T$
NP	yes	yes	open	open
Σ^P_i	yes	yes	open	open
Π^P_i	yes	yes	open	open
Δ^P_i	yes	yes	open	open
PSPACE	yes	yes	open	open
EXP	yes	yes	open	open
NEXP	yes	yes	open	open

Table 1: For a complexity class $\mathcal{C}$ and a reducibility $\leq$, the corresponding entry shows the status of the question of whether all nontrivial $\leq$-complete sets in $\mathcal{C}$ are $\leq$-autoreducible/$\leq$-mitotic.

5 Summary

A summary of our current knowledge regarding the autoreducibility and mitoticity of sets that are complete for various complexity classes can be found in Table 1.

References

[AA96] M. Agrawal and V. Arvind. Quasi-linear truth-table reductions to p-selective sets. *Theoretical computer Science*, 158:361–370, 1996.

[ALM$^+$92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the intractability of approximation problems. In *Proceedings 33rd Symposium on the Foundations of Computer Science*, pages 14–23. IEEE Computer Society Press, 1992.

[AS84] K. Ambos-Spies. P-mitotic sets. In E. Börger, G. Hasenjäger, and D. Roding, editors, *Logic and Machines, Lecture Notes in Computer Science 177*, pages 1–23. Springer-Verlag, 1984.

[AW09] M. Agrawal and O. Watanabe. One-way functions and the isomorphism conjecture. Technical Report TR09-019, ECCC, 2009. To Appear in CCC 2009.

[Ber77] L. Berman. *Polynomial Reducibilities and Complete Sets*. PhD thesis, Cornell University, Ithaca, NY, 1977.

[BF92] R. Beigel and J. Feigenbaum. On being incoherent without being very hard. *Computational Complexity*, 2:1–17, 1992.

[BFvMT00] H. Buhrman, L. Fortnow, D. van Melkebeek, and L. Torenvliet. Separating complexity classes using autoreducibility. *SIAM Journal on Computing*, 29(5):1497–1520, 2000.

[BH77] L. Berman and J. Hartmanis. On isomorphism and density of NP and other complete sets. *SIAM Journal on Computing*, 6:305–322, 1977.

[7]By Theorem 3.14, any answer of this question results in a nontrivial separation of complexity classes.

[BK89] M. Blum and S. Kannan. Designing programs that check their work. In *Proceedings of the 21st ACM Symposium on Theory of Computing*, pages 86–97, 1989.

[BKS95] R. Beigel, M. Kummer, and F. Stephan. Approximable sets. *Information and Computation*, 120:304–314, 1995.

[BLS$^+$04] B. Borchert, K. Lange, F. Stephan, P. Tesson, and D. Thérien. The dot-depth and the polynomial hierarchy correspond on the delta levels. In *Developments in Language Theory*, pages 89–101, 2004.

[BSS99] B. Borchert, H. Schmitz, and F. Stephan. Unpublished manuscript, 1999.

[BT05] H. Buhrman and L. Torenvliet. A Post's program for complexity theory. *Bulletin of the EATCS*, 85:41–51, 2005.

[CF91] J.-Y. Cai and M. Furst. PSPACE survives constant-width bottlenecks. *International Journal of Foundations of Computer Science*, 2:67–76, 1991.

[CPS99] J. Cai, A. Pavan, and D. Sivakumar. On the hardness of permanent. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, volume LNCS, 1627, pages 90–99, 1999.

[FL92] U. Feige and C. Lund. On the hardness of computing permanent of random matrices. In *Proceedings of 24th Annual ACM Symposium on Theory of Computing*, pages 643–654, 1992.

[GH92] K. Ganesan and S. Homer. Complete problems and strong polynomial reducibilities. *SIAM Journal on Computing*, 21:733–742, 1992.

[Gol09] O. Goldreich. A candidate counterexample to the easy cylinders conjecture. Technical Report TR09-028, ECCC, 2009.

[GOP$^+$07] C. Glaßer, M. Ogihara, A. Pavan, A. L. Selman, and L. Zhang. Autoreducibility, mitoticity, and immunity. *Journal of Computer and System Sciences*, 73(5):735–754, 2007.

[GPSZ08] C. Glaßer, A. Pavan, A. L. Selman, and L. Zhang. Splitting NP-complete sets. *SIAM Journal on Computing*, 37(5):1517–1535, 2008.

[GRS99] O. Goldreich, D. Ron, and M. Sudan. Chinese remaindering with errors. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC'99)*, pages 225–234, 1999.

[GS92] P. Gemmel and M. Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43:169–174, 1992.

[GSTZ09] C. Glaßer, A. L. Selman, S. Travers, and L. Zhang. Non-mitotic sets. *Theoretical Computer Science*, 410(21-23):2011–2023, 2009.

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

[Lac67] A. H. Lachlan. The priority method I. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 13:1–10, 1967.

[LFKN92] C. Lund, L. Fortnow, H. Karloff, and N.Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

[Lip91] R. Lipton. New directions in testing. In *Distributed Computing and Cryptography*, volume 2 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 191–202. American Mathematics Society, 1991.

[Mah82] S. Mahaney. Sparse complete sets for NP: Solution of a conjecture of Berman and Hartmanis. *Journal of Computer and Systems Sciences*, 25(2):130–143, 1982.

[Ogi95] M. Ogihara. Polynomial-time membership comparable sets. *SIAM Journal on Computing*, 24:1168–1181, 1995.

[OKSW94] P. Orponen, K. Ko, U. Schöning, and O. Watanabe. Instance complexity. *Journal of the ACM*, 41(1):96–121, 1994.

[OW91] M. Ogiwara and O. Watanabe. On polynomial-time bounded truth-table reducibility of NP sets to sparse sets. *SIAM Journal on Computing*, 20(3):471–483, 1991.

[Sha92] A. Shamir. IP = PSPACE. *Journal of the Association for Computing Machinery*, 39:869–877, 1992.

[Tra70] B. Trakhtenbrot. On autoreducibility. *Dokl. Akad. Nauk SSSR*, 192(6):1224–1227, 1970. Translation in Soviet Math. Dokl. 11(3): 814–817, 1970.

[Tra02] S. Travers. Blattsprachen Komplexitätsklassen: Über Turing-Abschluss und Counting-Operatoren, Studienarbeit, 2002.

[TV02] L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. In *Annual IEEE Conference on Computational Complexity*, volume 17, pages 129–138, 2002.

[VV86] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986.

[Yao90] A. C. C. Yao. On ACC and threshold circuits. In *Proceedings 31th Foundations of Computer Science*, pages 619–627. IEEE Computer Society Press, 1990.