CUCS-343-88

Structure of Complexity Classes: Separations, Collapses, and Completeness

Lane A. Hemachandra[*]
Department of Computer Science
Columbia University
New York, NY 10027, USA

Abstract

During the last few years, unprecedented progress has been made in structural complexity theory; class inclusions and relativized separations were discovered, and hierarchies collapsed. We survey this progress, highlighting the central role of counting techniques. We also present a new result whose proof demonstrates the power of combinatorial arguments: there is a relativized world in which UP has no Turing complete sets.

1 Introduction

Two years ago, the hunting season opened. Quickly, the strong exponential hierarchy fell, followed by the linear-space, logspace, and logspace oracle hierarchies. Soon the LBA problem, a venerable precursor of P =? NP, had been added to the game sack. Today, open season has been declared on P =? NP. Many reasonable men express confidence that P =? NP will be resolved within a decade—a view that only a few years ago would have been heretical.

In the first part of this paper, we survey some highlights of recent progress in structural complexity theory, and identify the reason for this sudden progress. For many years, NP was viewed as a mysterious black box. During the last few years, we have started to open that box. Counting and combinatorial techniques have been used to explore, and exploit, the strengths and weaknesses of nondeterministic computation.

We survey recent progress on collapsing hierarchies, establishing complexity class containments, and proving relativized separations and non-completeness results.

The second part of this paper—Section 3—presents a result that extends the use of counting arguments to a new area—proving Turing non-completeness. We show that there is a relativized world in which the cryptographic class UP, unique polynomial time, has no Turing complete sets. This extends a long line of research on completeness that has been pursued by Ambos-Spies, Hartmanis, Immerman, and Sipser [Sip82,HI85,HH86, Amb86]. The goal of this research is to understand the structure of complexity classes by determining which classes have complete sets under which reducibilities.

[*]Supported by NSF grant CCR-8809174 and a Hewlett-Packard Corporation equipment grant.

2 Recent Progress in Structural Complexity Theory

2.1 Collapsing Hierarchies and Class Containments

The most noteworthy recent progress in structural complexity theory is the sudden collapse of complexity hierarchies. The most surprising aspect of these collapses is the simplicity and elegance of the techniques used. The LBA problem, which remained open for twenty-five years, has a four-page resolution.

The key technique in these collapses has been the use of *census functions*—functions that count. Usually, census functions count the number of elements in prefixes of a set having certain properties. Census functions are not new; Mahaney's proof that NP has no sparse complete sets unless P = NP is based on the use of census information [Mah82].

We start by presenting two of the earlier of recent uses of census functions to explore class inclusions. The first shows the relationship between Turing reductions and truth-table reductions, and the second strengthens a long line of "small circuit [KL80]" results.

$\mathrm{P}^{\mathrm{NP}[\log]}$ indicates the class of languages accepted by polynomial-time Turing machines that make $\mathcal{O}(\log n)$ calls to an NP oracle. $\mathrm{P}^{\mathrm{NP}}_{truth-table}$ indicates the class of languages that are polynomial-time truth-table reducible to NP [LLS75].

Theorem 2.1 [Hem87c]
$\mathrm{P}^{\mathrm{NP}[\log]} = \mathrm{P}^{\mathrm{NP}}_{truth-table}$.

The proof simply uses binary search to find, in $\mathcal{O}(\log n)$ queries to NP, the number m of queries of the truth-table reduction that receive the answer "yes," and then uses one further NP query to guess and check which m queries are answered "yes" and determine if the truth-table system accepts.

Proof Sketch $S \leq^{p}_{truth-table}$ NP means there is a polynomial-time machine that answers "$x \in S$?" by making queries to SAT [GJ79], such that the queries asked of SAT are independent of the answers received [LLS75]. To prove the $\subseteq$ part, we have P perform binary search, using its NP oracle, to find the *number* of yes answers, and with one final query to the oracle have NP guess which queries receive yes answers and simulate the action of the truth-table reducer. The $\supseteq$ part it trivial—the truth-table reducer asks all queries that might be formed by any of the $n^{\mathcal{O}(1)}$ possible sets of oracle answers of the run of $\mathrm{P}^{\mathrm{NP}[\log]}$. ∎

Detailed investigations of the interleaving of truth-table, oracle query, and boolean hierarchies can be found in [KSW86,AG87,Bei87].

The next example, due to Kadin, strengthens theorems of Karp, Lipton, Long, and Mahaney [KL80,Mah82,Lon82]. The Karp-Lipton "small circuits" theorem (so-called as the hypothesis is equivalent to "NP has small circuits") states:

Theorem 2.2 [KL80] If there is a sparse[1] set S such that NP $\subseteq \mathrm{P}^{S}$, then $\mathrm{NP}^{\mathrm{NP}} = \mathrm{PH}$, where PH is the polynomial hierarchy.

For the case of *simple* sparse sets, this was extended by Mahaney.

[1] A set A is sparse if it has only polynomially many elements of length at most n. That is, $(\exists k)(\forall n)$[there are at most $n^{k} + k$ elements of length $\leq n$ in A].

Theorem 2.3 [Mah82] If there is a sparse set $S \in \mathrm{NP}$ such that $\mathrm{NP} \subseteq \mathrm{P}^S$, then $\mathrm{P}^{\mathrm{NP}} = \mathrm{PH}$.

Kadin observed that Theorem 2.3 can be strengthened using census functions.

Theorem 2.4 [Kad87] If there is a sparse set $S \in \mathrm{NP}$ such that $\mathrm{NP} \subseteq \mathrm{P}^S$, then $\mathrm{P}^{\mathrm{NP}[\log]} = \mathrm{PH}$.

Proof Sketch Let $census_S(1^i)$ equal the number of elements in S of length at most i. If there is a sparse set S as described in the theorem, then a P machine can make $O(\log n)$ calls to an NP oracle to find—via binary search—$census_S(1^n)$. It is easy to see that, with $O(\log n)$ NP queries to calculate the census function, and one further query (to guess exactly which strings are in S up to a certain length), $\mathrm{P}^{\mathrm{NP}[\log]}$ can simulate P^S. So, using Theorem 2.3, $\mathrm{PH} \subseteq \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{P}^{(\mathrm{P}^S)} \subseteq \mathrm{P}^S \subseteq \mathrm{P}^{\mathrm{NP}[\log]}$. ∎

The first hierarchy to fall under attack by census functions was the strong exponential hierarchy. Nondeterministic exponential time, NE, is defined as $\bigcup_{c>0} \mathrm{NTIME}[2^{cn}]$. The following theorem states that the strong exponential hierarchy collapses.

Theorem 2.5 [Hem87c]
$\mathrm{P}^{\mathrm{NE}} = \mathrm{E} \cup \mathrm{NE} \cup \mathrm{NP}^{\mathrm{NE}} \cup \mathrm{NP}^{\mathrm{NP}^{\mathrm{NE}}} \cup \cdots$.

This follows immediately from the lemma that $\mathrm{P}^{\mathrm{NE}} = \mathrm{NP}^{\mathrm{NE}}$. The collapse was first proven by a census argument that—level by level in the $\mathrm{NP}^{\mathrm{NE}}$ computation tree—finds the number of yes answers to queries, and yields strengthenings of Theorem 2.5. Schöning and Wagner [SW88] developed the following elegant proof: consider an NP machine with an NE oracle A. The NP machine can query, for some k, at most n^k strings in A. Note, however, that P^{NE} can, by binary search, compute the function $census_A$, and thus simulate $\mathrm{NP}^{\mathrm{NE}}$.

The strong exponential hierarchy is a time hierarchy. However, the same techniques apply also to space hierarchies. The linear-space hierarchy, the logspace hierarchy, and the logspace hierarchy all collapsed under the application of census techniques [Tod87,LJK87,SW88]. These space results were strengthened by Szelepcsényi and Immerman [Sze87,Imm87].

Theorem 2.6 [Sze87,Imm87] For any space constructible $S(n) \geq \log n$, $\mathrm{NSPACE}[S(n)] = \mathrm{co}-\mathrm{NSPACE}[S(n)]$.

The proof has been widely circulated. It uses *inductive counting*—iteratively counting the number of reachable configurations at each distance from the start of a co-NSPACE$[S(n)]$ computation. In part, this comes full circle to the iterative use of census used to collapse the strong exponential hierarchy.

Inductive counting has since been used to show that the class of languages that logspace reduce to some context-free language, LOG(CFL), is closed under complement.

Theorem 2.7 [BCRT88]
LOG(CFL) = co-LOG(CFL).

A number of class containment results have been proven recently using the ability of nondeterministic machines to make extra copies of accepting paths. This approach has been used to evaluate the complexity of classes in the counting hierarchy—a hierarchy based on NP machines with altered acceptance mechanisms [CH86,GW, CGH*b]. More recently, this approach has been used to show that parity polynomial time is powerful enough to contain FewP—the subset of NP languages that are accepted by machines that never have many accepting paths.

Definition 2.8

1. [PZ83] (Parity Polynomial Time) $\oplus\mathrm{P} = \{L \mid$ there is a nondeterministic polynomial-time Turing machine N such that $x \in L$ if and only if $N(x)$ has an odd number of accepting paths$\}$.
2. [All86] $\mathrm{FewP} = \{L \mid$ there is a nondeterministic polynomial-time Turing machine N such that (1) $x \in L$ if and only if $N(x)$ has at least one accepting path and (2) $(\exists k)(\forall x)[N(x)$ has at most $|x|^k + k$ accepting paths$]\}$.
3. [CH87] Few is the class of all languages L such that there is a nondeterministic polynomial-time Turing machine N, a polynomial-time computable predicate $Q(\cdot,\cdot)$, and a polynomial $q(\cdot)$, such that (1) $x \in L$ if and only if $Q(x, ||N(x)||)$, and (2) $(\forall x)[||N(x)|| \leq q(|x|)]$, where $||N(x)||$ denotes the number of accepting paths of $N(x)$.

Theorem 2.9 [CH87] $\oplus\mathrm{P} \supseteq \mathrm{Few}$.

Corollary 2.10 [CH87] $\oplus\mathrm{P} \supseteq \mathrm{FewP}$.

A direct proof of the corollary is immediate. Given a FewP machine, N_i, that on inputs of length n never has more than $n^k + k$ accepting paths, we construct a new machine N, a nondeterministic machine with the parity acceptance mechanism, so that $N(x)$ has a path for each path of $N_i(x)$, and a path for each pair of paths of $N_i(x)$, $\ldots$, and a path for each $(n^k + k)$-tuple of paths of $N_i(x)$. Each path of $N(x)$ will accept if and only if all of the paths that it represents in $N_i(x)$ are accepting paths. Since $\sum_{1 \leq i \leq n^k+k} \binom{j}{i} = 2^j - 1$ is odd exactly when $j \neq 0$, it follows that $\mathrm{FewP} \subseteq \oplus\mathrm{P}$. The proof that $\mathrm{Few} \subseteq \oplus\mathrm{P}$ takes a bit more work, but follows the same lines.

2.2 Separations

The previous section described a number of collapsing hierarchies and class containments. Hierarchy separations, or even class separations, remain elusive. However, a number of hierarchies have been separated in relativized worlds. These include the boolean hierarchy, the counting hierarchy [CGH*b], and, most importantly, the polynomial hierarchy.

Definition 2.11

1. **Boolean Hierarchy** [Wec85] $\Sigma_0^{BH} = \mathrm{P}$. $\Sigma_1^{BH} = \mathrm{NP}$. $\Sigma_i^{BH} = \{L \mid (\exists L' \in \mathrm{NP})(\exists L'' \in \Sigma_{i-1}^{BH})[L = L' - L'']\}$, $i > 1$.
2. **Polynomial Hierarchy** [Sto77] $\Sigma_0^p = \mathrm{P}$. $\Sigma_1^p = \mathrm{NP}$. $\Sigma_i^p = \mathrm{NP}^{\Sigma_{i-1}^p}$, $i > 1$.

Theorem 2.12 [CGH*a]

1. There is a relativized world A in which $\Sigma_0^{BH}(A) \neq \Sigma_1^{BH}(A) \neq \cdots$.
2. For each k there is a relativized world A in which $\Sigma_0^{BH}(A) \neq \cdots \neq \Sigma_k^{BH}(A) = \Sigma_{k+1}^{BH}(A) = \cdots$.

Theorem 2.13 [Yao85,Ko88]

1. There is a relativized world A in which $\Sigma_0^p(A) \neq \Sigma_1^p(A) \neq \cdots$.
2. For each k there is a relativized world A in which $\Sigma_0^p(A) \neq \cdots \neq \Sigma_k^p(A) = \Sigma_{k+1}^p(A) = \cdots$.

Yao's and Ko's results exploit the connection between circuit theory and the theory of relativizations [FSS84].

The boolean hierarchy results are proven by exactly the approach we'll use in Section 3. An NP machine accepts, by definition, when *any* computation path accepts. This is an insensitive mechanism, and leaves NP open to relativized manipulation.

The crudeness of the acceptance mechanism of NP has also been used to show *robustness* [Sch85] results—results showing the difficulty of NP machines maintaining some computational invariant in all relativized worlds. The idea behind the proofs is that the crudeness of the acceptance mechanism tends to make NP machines oblivious to fine changes in the oracle. A representative result is that if a nondeterministic polynomial-time Turing machine N_i is categorical (i.e., has at most one accepting path on any input [Val76]) in every relativized world, then in every relativized world A, the language $L(N_i^A)$ is relatively simple. It falls into the second level of Schöning's [Sch83, Sch86] extended low hierarchy. The results below are from [HH87].

Theorem 2.14 (Robustly categorical machines accept simple languages)
$(\forall A)[N_i^A \text{ is categorical}] \Rightarrow (\forall A)[L(N_i^A) \in P^{NP \oplus A}]$.

Corollary 2.15 If $P = NP$ and N_i is robustly categorical (i.e., $(\forall A)[N_i^A$ is categorical]), then for every oracle A, $L(N_i^A) \in P^A$.

Theorem 2.16 (Robustly complementary machines accept simple languages)
$(\forall A)[L(N_i^A) = \overline{L(N_j^A)}] \Rightarrow (\forall A)[L(N_i^A) \in P^{NP \oplus A}]$.

Corollary 2.17 [2] If $P = NP$ and N_i and N_j are robustly complementary (i.e., $(\forall A)[L(N_i^A) = \overline{L(N_j^A)}]$), then for every oracle A $L(N_i^A) \in P^A$.

Theorem 2.18 (Machines robustly Σ^*-accepting on sparse oracles accept for transparent reasons)
$(\forall \text{ sparse } S)[L(N_i^S) = \Sigma^*] \Rightarrow (\forall \text{ sparse } S)(\exists f$ computable in $P^{NP \oplus S})(\forall x)[f(x)$ prints an accepting path of $N_i^S(x)]$.

Corollary 2.19 If $P = NP$ and N_i robustly accepts Σ^* on sparse oracles (i.e., $(\forall \text{ sparse } T)$ $[L(N_i^T) = \Sigma^*]$), then for every sparse oracle S, there is a function f computable in P^S so that on any input x, $f(x)$ prints an accepting path of $N_i^S(x)$.

Finally, we mention a lovely, combinatorially deep relativization of Torán [Tor88]. He has shown that there is an oracle A for which $NP^A - \oplus P^A \neq \emptyset$.

2.3 Completeness

The same obliviousness of NP machines that yielded the relativized separations mentioned in Section 2.2 has also led to proofs that some natural complexity classes may lack complete sets.

Complete languages have long been a useful tool in complexity theory. Much of our knowledge about NP comes from studying the NP-complete set SAT [Mah82,KL80,BH77]. Most common

[2]Manuel Blum and Russell Impagliazzo [BI87] independently proved Corollary 2.17 in their study of generic oracles. Tardos [Tar87] studies an analog where unbounded computational time is combined with restricted oracle access.

complexity classes—NP, coNP, PSPACE, etc.—have many-one complete sets that help us study them.

Sipser noted, however, that some classes may lack complete sets [Sip82]. His paper sparked much research into which classes have complete languages, and what strengths of completeness results (e.g., many-one or Turing) can be obtained. Of course, if P = PSPACE, then *all* classes between P and PSPACE have many-one complete languages. Thus, incompleteness results are typically displayed in relativized worlds [BGS75,Sip82].

Sipser showed that there are relativized worlds in which R and NP $\cap$ coNP lack many-one complete languages. Hartmanis and Hemachandra showed a relativized world in which UP—unique polynomial time (Section 3.1)—lacks many-one complete languages, and noted that if UP does have complete languages then UP has complete languages with an unusually simple form—the intersection of SAT with a set in P [HH86].

One way of strengthening the above theorems would be to show that these classes lack complete sets even with respect to reducibilities more flexible than many-one reductions, e.g., k-truth-table, positive truth-table, truth-table, and ultimately Turing reductions [LLS75]. Hartmanis and Immerman, exploiting an insightful characterization of Kowalczyk [Kow84], showed that NP $\cap$ coNP has many-one complete languages if and only if it has Turing complete languages [HI85]. An elegant generalization of their result by Ambos-Spies shows that for any class C closed under Turing reductions, C has Turing complete sets if and only if C has many-one complete sets [Amb86]. In particular, it follows from the result of Sipser [Sip82] that there is a relativized world A in which $\mathrm{NP}^A \cap \mathrm{coNP}^A$ lacks Turing complete sets [HI85, Amb86]. Similarly, since $\mathrm{P}^{\mathrm{BPP}} = \mathrm{BPP}$ [Zac86], from [HH86]'s proof that BPP lacks many-one complete sets in some relativized worlds it follows that it also may lack Turing complete sets.

Theorem 2.20 *There is a relativized world A in which BPP^A lacks Turing complete sets.*

However, Ambos-Spies's result does not apply to UP or any other class not known to be closed under Turing reductions. Furthermore, the technique used to show that UP may lack many-one complete languages was an indirect proof via the contradiction of an enumeration condition that characterized the existence of many-one complete languages [HH86]—and does not generalize to the case of Turing completeness.

Section 3 constructs an oracle A for which UP^A contains no Turing complete sets. Our proof exploits the limited combinatorial control of nondeterministic machines to trivialize or corrupt candidates for Turing completeness. This approach extends our theme: the exploitation of the combinatorics of the nondeterministic acceptance mechanism.

It follows immediately from our proof that there is a relativized world A in which UP^A lacks complete languages under all reducibilities more restrictive than Turing reductions.

3 Does UP have Turing Complete Languages?

3.1 Definitions

Definition 3.1 [Val76]
(Unique Polynomial Time) UP = $\{L \mid$ there is a nondeterministic polynomial time Turing machine N such that $L = L(N)$, and for all x, the computation of $N(x)$ has at most one accepting path$\}$. We say that a machine N that for every input has at most one accepting path is *categorical.*

UP captures the power of uniqueness; UP is the class of problems that have (on some NP machine) unique witnesses. That is, if there is an NP machine N accepting L and for every input x the computation $N(x)$ has at most one accepting path (i.e., N is a categorical machine), then we say $L \in$ UP.

Recently, UP has come to play a crucial role in both cryptography and structural complexity theory. In cryptography, Grollmann and Selman have shown that one-way functions[3] exist if and only if P $\neq$ UP, and one-way functions whose range[4] is in P exist if and only if P $\neq$ UP $\cap$ coUP. Thus, we suspect that P $\neq$ UP because we suspect that one-way functions exist. In structural complexity theory, a conjecture that "P $\neq$ UP $\Longleftrightarrow$ there exist non-p-isomorphic NP-complete sets" was recently refuted in a relativized world [HH87].

[3] A function f is *honest* if $(\exists k)(\forall x)[|f(x)|^k + k \geq |x|]$ ([GS84], see also [Wat86]). A *one-way function* is a total, single-valued, one-to-one, honest, polynomial time computable function f such that f^{-1} (which will be a partial function if range$(f) \neq \Sigma^*$) is not computable in polynomial time [GS84].
[4] *Range*$(f) = \bigcup_{i \in \Sigma^*} f(i)$.

For background, we first define Turing reductions and completeness in the real (unrelativised) world.

Definition 3.2

1. $S_1 \leq_T^p S_2$ if $S_1 \subseteq \mathrm{P}^{S_2}$ [GJ79].
2. L is $\leq_T^p$-complete for UP if $L \in$ UP and every set in UP Turing reduces to L (i.e., $(\forall S \in \mathrm{UP})[S \leq_T^p L]$).

If we wish to discuss Turing completeness in relativised worlds, we must address the key question: are the Turing reductions allowed access to the oracle? Definitions 3.3.2 and 3.3.3 answer this question "yes" and "no," respectively.

Definition 3.3

1. $S_1 \leq_T^{p,A} S_2$ if $S_1 \subseteq \mathrm{P}^{S_2 \oplus A}$.
2. L is $\leq_T^{p,A}$-complete for UP^A if $[L \in \mathrm{UP}^A$ and $(\forall S \in \mathrm{UP}^A)[S \leq_T^{p,A} L]]$.
3. L is $\leq_T^p$-complete for UP^A if $[L \in \mathrm{UP}^A$ and $(\forall S \in \mathrm{UP}^A)[S \leq_T^p L]]$.

We suggest that Definition 3.3.2 above is the natural notion of relativized Turing completeness. Adopting it, we prove that there is a relativized world in which UP^A has no $\leq_T^{p,A}$-complete sets. However, for purposes of completeness results, the different notions of relativized Turing reductions stand or fall together.

Lemma 3.4 For any oracle A: [UP^A has $\leq_T^{p,A}$-complete sets if and only if UP^A has $\leq_T^p$-complete sets].

This is true since if B is $\leq_T^{p,A}$-complete for UP^A, then $B \oplus A$ is $\leq_T^p$-complete for UP^A. The analog of Lemma 3.4 for many-one reductions was proven by Sipser [Sip82].

The difference between Definitions 3.3.2 and 3.3.3 is exactly the difference between "full" (3.3.2) and "partial" (3.3.3) relativization discussed in [KMR86] and [Rog67, Section 9.3]. [KMR86] describes how this distinction has had a crucial effect on recent research asking if all NP-complete sets are polynomially isomorphic [Kur83,GJ86,HH87]. However, Lemma 3.4 indicates that in our study of Turing completeness, we need not be concerned with the distinction.

3.2 A Relativized World in Which UP Lacks Turing Complete Sets

This section sketches the construction of an oracle for which UP^A has no Turing complete sets. It follows immediately that UP^A lacks complete sets with respect to reductions more restrictive than $\le_T^{p,A}$, such as truth-table reductions [LLS75], bounded truth-table reductions [LLS75], etc.

Theorem 3.5 There is a recursive oracle A such that UP^A contains no $\le_T^{p,A}$-complete sets.

Corollary 3.6 There is a recursive oracle A such that UP^A contains no:

1. $\le_T^p$-complete languages.
2. truth-table complete languages.
3. bounded truth-table complete languages.
4. [HH86] $\le_m^p$ or $\le_m^{p,A}$-complete languages.

Let $\{N_i\}$ be a standard enumeration of nondeterministic polynomial-time Turing machines and let $\{M_i\}$ be a standard enumeration of deterministic polynomial-time Turing machines. The idea of the proof is as follows. We wish to show that for no $L \in \mathrm{UP}^A$ is $\mathrm{UP}^A \subseteq \mathrm{P}^L$, which suffices by Lemma 3.4. Each L in UP^A is, by definition, accepted by a *categorical* machine ($L = L(N_i^A)$, N_i^A categorical). Our goal is to show that for each i, either

1. N_i^A is not categorical, or
2. $(\exists \hat{L}_i)[\hat{L}_i \in \mathrm{UP}^A$ and $\hat{L}_i \notin \mathrm{P}^{L(N_i^A)}]$.

The second condition says that some UP^A language does not Turing reduce to $L(N_i^A)$. That is, every Turing reduction fails on some value. Thus it certainly suffices to show that for all i, either

1. N_i^A is not categorical or
2. (a) $(\forall j)(\exists x)[x \in \hat{L}_i \not\Leftrightarrow x \in L(M_j^{L(N_i^A)})]$, where $\hat{L}_i = \{1^n \mid (\exists k)[(n = (p_i)^k) \wedge (\exists y)[|y| = n \wedge y \in A]]\}$ and p_i is the ith prime, and
   (b) $\hat{L}_i \in \mathrm{UP}^A$.

Note that we have specified $\hat{L}_i$.

Briefly put, for each $< i, j >$, we seek to find a way of extending the oracle to make N_i^A noncategorical. Failing this, we argue that we can choose our oracle in such a way as to determine the answers to all oracle queries made by M_j, and still have the flexibility to diagonalize against $\hat{L}_i$. The crucial step is a combinatorial argument that categorical machines which don't trivially accept must reject on an overwhelming number of oracle extensions.

Proof Sketch for Theorem 3.5

We wish to show that there is a relativized world A where UP^A has no Turing complete languages, i.e.,

$$(\forall L \in \mathrm{UP}^A)(\exists L' \in \mathrm{UP}^A)[L' \not\le_T^{p,A} L].$$

By Lemma 3.4 it suffices to show that

$$(\forall L \in \mathrm{UP}^A)(\exists L' \in \mathrm{UP}^A)[L' \not\leq^p_T L].$$

Since each language in UP^A is accepted by at least one categorical machine, we can equivalently show that

$$(\forall i)[(N_i^A \text{ is noncategorical}) \vee$$
$$(\exists \hat{L}_i \in \mathrm{UP}^A)[\hat{L}_i \not\leq^p_T L(N_i^A)]].$$

This says that:

$(\forall i)[(N_i^A \text{ is noncategorical}) \vee$
$(\exists \hat{L}_i \in \mathrm{UP}^A)[\hat{L}_i \not\in \mathrm{P}^{L(N_i^A)}]].$ $**$

Let requirement $R_{<i,j>}$ be
$R_{i,j}$: $(\exists x)[x \in \hat{L}_i \not\Leftrightarrow x \in L(M_j^{L(N_i^A)})]$, where $\hat{L}_i = \{1^n \mid (\exists k \ni n = p_i^k) \wedge (\exists y)[|y| = n \wedge y \in A]\}$, and p_i is the ith prime.

Note that we satisfy $(**)$ if we can satisfy for all i the following, which simply (1) specifies the $\hat{L}_i$, (2) uses the fact that $\mathrm{P}^X = \bigcup_i L(M_i^X)$, where $\{M_i\}$ is a standard enumeration of polynomial machines and without loss of generality M_i runs in time $n^i + i$, and (3) notes that differing languages must differ on some specific element.

1. N_i^A is noncategorical, OR $***$

2. $(\forall j)[R_{i,j}$ is satisfied] and $(\hat{L}_i \in \mathrm{UP}^A)$.$****$

Our construction will go by stages. In stage $<i,j>$ we will either satisfy $R_{i,j}$ or know that N_i^A is noncategorical.

Initially set $A_{<i,j>} := \emptyset$. We'll have $A = \bigcup_{<i,j>} A_{<i,j>}$.

Stage $<i,j>$: If N_i^A has already been made noncategorical, skip this stage and set $A_{<i,j>} := A_{<i,j>-1}$. Otherwise, choose a huge integer n that is a power of the ith prime (i.e., so $(\exists k \ni n = p_i^k)$) and is much larger than any previously touched length.

A *legal* extension of $A_{<i,j>}$ will be one that does not touch any string shorter than n, and that adds strings to $A_{<i,j>}$ only at lengths that are powers of p_i.

If there is a legal extension $\hat{A}$ of $A_{<i,j>-1}$ such that for some $y \ni |y| \leq (n^j + j)^i + i$ we have that $N_i^{\hat{A}}(y)$ is noncategorical,[5] then choose two accepting paths of $N_i^{\hat{A}}(y)$ and set $A_{<i,j>}$ to be $A_{<i,j>-1}$ augmented to agree with $\hat{A}$ on all strings queried on those two paths and go to the next stage.

Otherwise, we've failed to make $N_i^{A_{<i,j>}}$ noncategorical, so we must try to satisfy requirement $R_{<i,j>}$. *Our goal is to fix the behavior of $M_j^{L(N_i^A)}(1^n)$ and yet have enough flexibility left to ensure that $1^n \in \hat{L}_i$ or $1^n \not\in \hat{L}_i$ as we wish. Thus we can diagonalize to insure that $R_{<i,j>}$ is satisfied. Furthermore,* unless *we discover a way of making N_i^A noncategorical, we'll put only one string into A at each length p_i^k thus insuring that $\hat{L}_i \in \mathrm{UP}^A$.*

We simulate the behavior of $M_j^{L(N_i^{A_{<i,j>-1}})}(1^n)$, and each time M_j makes an oracle query, we take action to control that query.

We use T_l to indicate the set of strings of length n to which the lth query is oblivious. We'll eventually show that the T_l are all so huge that there must be at some string in $T_1 \cap T_2 \cap \cdots T_{n^j+j}$.

Action for the first query:

Run $M_j^{L(N_i^{A_{<i,j>-1}})}(1^n)$ until M_j asks its first oracle query, q_1.

Case 1: $N_i^{A_{<i,j>-1}}(q_1)$ accepts. Freeze the elements along the accepting path, and proceed to action for the second query. Set $T_1 = \{z \mid |z| = n$

[5] We *could* remove the bound on y's size, but this would make the construction nonrecursive.

and z has not just been frozen}.

Case 2: $N_i^{A_{<i,j>-1}}(q_1)$ rejects. We argue that there are a huge number of strings of length n that can be added to the oracle which will not change this rejection. Let $T_1 = \{z \mid n = |z|$ and the membership of element z in A has not yet been determined and $N_i^{A_{<i,j>-1} \cup \{z\}}(q_1)$ rejects$\}$. Let $t_1 = ||T_1||$. By Lemma 3.7, $t_1 \geq 2^n - 8((n^j + j)^{2i} + i^2)$.

Action for the lth query:

At this point, we've already determined the answers to the first $l-1$ oracle queries. We proceed analogously to the action for the first query (except we respect—and use k of Lemma 3.7 to account for—strings frozen from case 1 of earlier queries, which causes the bounds on T_l to weaken slightly as l grows).

End of query sequence

Our goal was to fix the responses to the queries while leaving ourselves enough freedom to make $1^n \in \hat{L}_i$ *or* $1^n \notin \hat{L}_i$ *as we like. We can now do that.*

Each T_l is easily of size $\geq 2^n - n^{\log n}$. Thus, since each T_l is a subset of the set of length n strings, and since there are at most $n^j + j$ T_l's, there is some length n string $\hat{z}$ in $T_1 \cap T_2 \cap \cdots \cap T_{n^j+j}$. By the definition of the T_l's, adding this string to $A_{<i,j>-1}$ will have no effect on any of the oracle responses. That is, $M_j^{L(N_i^{A_{<i,j>-1}})}(1^n)$ accepts if and only if $M_j^{L(N_i^{A_{<i,j>-1}}) \cup \{\hat{z}\}}(1^n)$ accepts. If these do accept, choose $\hat{z} \notin A$. Thus, $1^n \notin \hat{L}_i$ but $1^n \in L(M_j^{L(N_i^A)})$, so requirement $R_{<i,j>}$ has been satisfied. On the other hand, if $M_j^{L(N_i^{A_{<i,j>-1}})}(1^n)$ rejects, choose $\hat{z} \in A$. Thus, $1^n \in \hat{L}_i$ but $1^n \notin L(M_j^{L(N_i^A)})$, so requirement $R_{<i,j>}$ has been satisfied.

End of Stage $< i, j >$

Note that if we never find a way of making N_i^A noncategorical, then $\hat{L}_i \in \mathrm{UP}^A$ (because the above procedure puts in only one string, $\hat{z}$, at each length important to $\hat{L}_i$) and $(\forall j)$[requirement $R_{<i,j>}$ is satisfied]. Thus $(* * **)$ is satisfied. On the other hand, if we do find a way of making N_i^A noncategorical, then $(* * *)$ is satisfied.[6] Thus we have met requirements that are, by the discussion following Corollary 3.6, sufficient to insure that UP^A has no $\leq_T^{p,A}$-complete languages. ∎

In the proof, we referred to the following lemma. Loosely, what the lemma says is that if a machine tries to be categorical on all possible oracle extensions and it rejects on the empty extension, then it must also reject on an overwhelmingly large proportion of extensions that add exactly one string.

Lemma 3.7 Let N_i^A be a nondeterministic Turing machine that runs in NTIME$[n^i + i]$. Suppose $N_i^A(x)$ rejects, A has no strings of length $|x|$, and k strings of length $|x|$ have been designated as forbidden from being added to the oracle A. Let $Rejectors_x = \{y \mid N_i^{A \cup \{y\}}(x)$ rejects and $|y| = |x|$ and y is not one of the k forbidden strings$\}$. Then either $||Rejectors_x|| \geq 2^n - (k + 8(|x|^{2i} + i^2))$ or there exists a set S such that (1) $||S|| \leq 2$, (2) S contains no forbidden string, (3) $(\forall w \in S)[|w| = |x|]$, and (4) $N_i^{A \cup S}$ is noncategorical (in particular, it has more than one accepting path on input x).

Proof Sketch for Lemma 3.7

[6] In this case $\hat{L}_i$ may not be in UP^A, but since we have met condition $(* * *)$, we don't care if $\hat{L}_i$ is in UP^A.

Let $Acceptors_x = \{y \mid N_i^{A \cup \{y\}}(x)$ accepts and $|y| = |x|$ and y is not one of the k forbidden strings$\}$. Suppose there are $l + k$ strings in $Acceptors_x$, and thus at least l usable acceptors. Each acceptor v triggers exactly one accepting path (otherwise, set $S = \{v\}$ and we're done). For each pair of distinct acceptors v_q and v_r, we must have that v_q is queried on the unique path on which $N_i^{A \cup \{v_r\}}(x)$ accepts or v_r is queried on the unique path on which $N_i^{A \cup \{v_q\}}(x)$ accepts (otherwise $N_i^{A \cup \{v_q, v_r\}}(x)$ has two accepting paths so set $S = \{v_q, v_r\}$ and we're done). There are $\binom{l}{2}$ candidate *pairs* of elements of $Acceptors_x$, which each threaten to make the machine noncategorical when both are simultaneously added. Consider just the set of paths, $Paths$, that accept for some string added from $Acceptors_x$. Clearly $l/(n^i+i) \leq ||Paths|| \leq l$. Each element $v \in Acceptors$ kills the candidacy of at most $(n^i+i)+p_v$ pairs, where p_v is the number of occurrences of v along paths in $Paths$. So the total number of killed candidates is $\leq ((n^i+i)||Paths||)(n^i+i)+||Paths||(n^i+i) \leq l((n^i+i)^2+(n^i+i))$. To avoid the existence of a set S as described in the lemma, we thus must have $\binom{l}{2} \leq l((n^i+i)^2+(n^i+i))$. So easily we have $l \leq 8(n^{2i}+i^2)$. ∎

4 Conclusion

This paper surveyed recent progress in structural complexity theory, and suggested that counting arguments have played a crucial role in these advances. We proved, using counting techniques, that there is a relativized world in which UP has no Turing complete languages.

Many related open problems remain. How much further can counting techniques by pushed in obtaining complexity hierarchy collapses, class containments, and relativised separations? What techniques might be used to separate complexity classes?

This paper has surveyed just a few of the interesting recent uses of counting. A great variety of applications and discussions of counting in complexity can be found in [All85,CH88,Hem86, Hem87b,Hem87a,Sch88,Sim77,Wag86].

References

[AG87] A. Amir and W. Gasarch. Polynomial terse sets. In *Proceedings 2nd Structure in Complexity Theory Conference*, pages 22–27, 1987.

[All85] E. Allender. Invertible functions. 1985. Ph.D. thesis, Georgia Institute of Technology.

[All86] E. Allender. The complexity of sparse sets in P. In *Proceedings 1st Structure in Complexity Theory Conference*, pages 1–11, Springer-Verlag *Lecture Notes in Computer Science #223*, June 1986.

[Amb86] K. Ambos-Spies. A note on complete problems for complexity classes. *Information Processing Letters*, 23:227–230, 1986.

[BCRT88] A. Borodin, S. Cook, W. Ruzzo, and M. Tompa. Two applications of complementation via induction counting. In *Proceedings 3rd Structure in Complexity Theory Conference*, IEEE Computer Society Press, June 1988. To appear.

[Bei87] R. Beigel. *Bounded Queries to SAT and the Boolean Hierarchy*. Technical Report TR-7, Johns Hopkins Department of Computer Science, Baltimore, MD, June 1987.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P=?NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[BH77] L. Berman and J. Hartmanis. On isomorphisms and density of NP and other complete sets. *SIAM Journal on Computing*, 6(2):305–322, 1977.

[BI87] M. Blum and R. Impagliazzo. Generic oracles and oracle classes. In *28th Annual IEEE Symposium on Foundations of Computer Science*, October 1987.

[Cai86] J. Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. In *18th ACM Symposium on Theory of Computing*, pages 21–29, 1986.

[CGH*a] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung. The boolean hierarchy I: Structural properties. To appear in *SIAM Journal on Computing*.

[CGH*b] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung. The boolean hierarchy II: Applications. To appear in *SIAM J. on Computing*.

[CH86] J. Cai and L. Hemachandra. The boolean hierarchy: Hardware over NP. In *Proceedings 1st Structure in Complexity Theory Conference*, pages 105–124, Springer-Verlag *Lecture Notes in Computer Science #223*, June 1986.

[CH87] J. Cai and L. Hemachandra. *On the Power of Parity Polynomial Time.* Technical Report CUCS 274-87, Columbia Computer Science Department, New York, NY, December 1987.

[CH88] J. Cai and L. Hemachandra. Enumerative counting is hard. In *Proceedings 3rd Structure in Complexity Theory Conference*, IEEE Computer Society Press, June 1988. To appear.

[FSS84] M. Furst, J. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

[GJ86] J. Goldsmith and D. Joseph. Three results on the polynomial isomorphism of complete sets. In *Proceedings 27th IEEE Symposium on Foundations of Computer Science*, pages 390–397, 1986.

[GS84] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. In *Proceedings 25th IEEE Symposium on Foundations of Computer Science*, pages 495–503, 1984.

[GW] T. Gundermann and G. Wechsung. Counting classes with finite acceptance types. To appear.

[Hem86] L. Hemachandra. *Can P and NP Manufacture Randomness?* Technical Report TR86-795, Cornell Computer Science Department, Ithaca, NY, December 1986.

[Hem87a] L. Hemachandra. *Counting in Structural Complexity Theory.* PhD thesis, Cornell University, Ithaca, NY, May 1987. Available as Cornell Department of Computer Science Technical Report TR87-840.

[Hem87b] L. Hemachandra. On ranking. In *Proceedings 2nd Structure in Complexity Theory Conference*, pages 103–117, IEEE Computer Society Press, June 1987.

[Hem87c] L. Hemachandra. The strong exponential hierarchy collapses. In *19th ACM Symposium on Theory of Computing*, pages 110–122, May 1987.

[HH86] J. Hartmanis and L. Hemachandra. Complexity classes without machines: On complete languages for UP. In

[Sch83] U. Schöning. A low and a high hierarchy in NP. *Journal of Computer and System Sciences.*, 27:14–28, 1983.

[Sch85] U. Schöning. Robust algorithms: A different approach to oracles. *Theoretical Computer Science*, 40:57–66, 1985.

[Sch86] U. Schöning. *Complexity and Structure*. Springer Verlag *Lecture Notes in Computer Science #211*, 1986.

[Sch88] U. Schöning. The power of counting. In *Proceedings 3rd Structure in Complexity Theory Conference*, IEEE Computer Society Press, June 1988. To appear.

[Sim77] J. Simon. On the difference between one and many. In *Automata, Languages, and Programming (ICALP 1977)*, pages 480–491, Springer-Verlag *Lecture Notes in Computer Science #52*, 1977.

[Sip82] M. Sipser. On relativization and the existence of complete sets. In *Automata, Languages, and Programming (ICALP 1982)*, Springer-Verlag *Lecture Notes in Computer Science #140*, 1982.

[Sto77] L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3:1–22, 1977.

[SW88] U. Schöning and K. Wagner. Collapsing oracle hierarchies, census functions, and logarithmically many queries. In *STACS 1988: 5th Annual Symposium on Theoretical Aspects of Computer Science*, Springer-Verlag *Lecture Notes in Computer Science*, February 1988.

[Sze87] R. Szelepcsényi. The method of forcing for nondeterministic automata. *Bulletin of the EATCS*, (33):96–99, 1987.

[Tar87] G. Tardos. Query complexity, or why is it difficult to separate $NP^A \cap coNP^A$ from P^A by random oracles A. July 1987. Manuscript.

[Tod87] S. Toda. Σ_2SPACE[n] is closed under complement. *Journal of Computer and System Sciences*, 35:145–152, 1987.

[Tor88] Jacobo Torán. *Structural Properties of the Counting Hierarchies*. PhD thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 1988.

[Val76] L. Valiant. The relative complexity of checking and evaluating. *Information Processing Letters*, 5:20–23, 1976.

[Wag86] K. Wagner. *Some Observations on the Connection Between Counting and Recursion.* Technical Report MIP-8611, Universität Passau, Fakultät für Mathematik und Informatik, June 1986.

[Wat86] Osamu Watanabe. On hard one-way functions (Abstract). *Bulletin of the European Association for Theoretical Computer Science*, June 1986.

[Wec85] G. Wechsung. On the boolean closure of NP. In *Proceedings of the 1985 International Conference on Fundamentals of Computation Theory*, pages 485–493, Lecture Notes in Computer Science, Springer-Verlag, 1985.

[Yao85] A. Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings 26th IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.

[Zac86] S. Zachos. Probabilistic quantifiers, adversaries, and complexity classes: An overview. In *Proceedings 1st Structure in Complexity Theory Conference*, pages 383–400, IEEE Computer Society Press, June 1986.

Automata, Languages, and Programming (ICALP 1986), pages 123–135, Springer-Verlag *Lecture Notes in Computer Science #226*, July 1986. To appear in *Theoretical Computer Science*.

[HH87] J. Hartmanis and L. Hemachandra. One-way functions, robustness, and the non-isomorphism of NP-complete sets. In *Proceedings 2nd Structure in Complexity Theory Conference*, pages 160–174, IEEE Computer Society Press, June 1987.

[HI85] J. Hartmanis and N. Immerman. On complete problems for NP$\cap$coNP. In *Automata, Languages, and Programming (ICALP 1985)*, pages 250–259, Springer-Verlag *Lecture Notes in Computer Science #194*, 1985.

[Imm87] N. Immerman. *Nondeterministic Space is Closed under Complement*. Technical Report YALEU/DCS/TR 552, Yale University, Department of Computer Science, New Haven, CT, July 1987. To appear in STRUCTURES 1988.

[Kad87] J. Kadin. $P^{NP[\log n]}$ and sparse Turing-complete sets for NP. In *Proceedings 2nd Structure in Complexity Theory Conference*, pages 33–40, IEEE Computer Society Press, June 1987.

[KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *12th ACM Sym. on Theory of Computing*, pages 302–309, 1980.

[KMR86] S. Kurtz, S. Mahaney, and J. Royer. Collapsing degrees. In *Proceedings 27th IEEE Symposium on Foundations of Computer Science*, pages 380–389, 1986.

[Ko88] K. Ko. Relativized polynomial time hierarchies having exactly k levels. In *20th ACM Symposium on Theory of Computing*, ACM Press, May 1988.

[Kow84] W. Kowalczyk. Some connections between representability of complexity classes and the power of formal reasoning systems. In *Mathematical Foundations of Computer Science*, pages 364–369, Springer-Verlag *Lecture Notes in Computer Science #176*, 1984.

[KSW86] J. Köbler, U. Schöning, and K. Wagner. *The Difference and Truth-Table Hierarchies for NP*. Technical Report, Fachberichte Informatik, EWH Rheinland-Pfalz, Koblenz, West Germany, July 1986.

[Kur83] S. Kurtz. *A Relativized Failure of the Berman-Hartmanis Conjecture*. Technical Report TR83-001, University of Chicago Department of Computer Science, Chicago, IL, 1983.

[LJK87] K. Lange, B. Jenner, and B. Kirsig. The logarithmic alternation hierarchy collapses: $A\Sigma_2^L = A\Pi_2^L$. In *Automata, Languages, and Programming (ICALP 1987)*, Springer-Verlag *Lecture Notes in Computer Science*, 1987.

[LLS75] R. Ladner, N. Lynch, and A. Selman. A comparison of polynomial time reducibilities. *Theoretical Computer Science*, 1(2):103–124, 1975.

[Lon82] T. Long. A note on sparse oracles for NP. *Journal of Computer and System Sciences*, 24:224–232, 1982.

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

[PZ83] C. Papadimitriou and S. Zachos. Two remarks on the power of counting. In *Proceedings 6th GI Conference on Theoretical Computer Science*, pages 269–276, Springer-Verlag Lecture Notes in Computer Science #145, 1983.

[Rog67] H. Rogers, Jr. *The Theory of Recursive Functions and Effective Computability*. McGraw-Hill, 1967.

Sep Eng

Lecture Notes in
Computer Science

Edited by G. Goos and J. Hartmanis

324

M. P. Chytil L. Janiga V. Koubek (Eds.)

MFCS '88

Mathematical Foundations of Computer Science 1988

Proceedings of the 13th Symposium
Carlsbad, Czechoslovakia, August 29 – September 2, 1988

Sep Eng-
709-D6182

Springer-Verlag
Berlin Heidelberg New York London Paris Tokyo