# On Some Promise Classes in Structural Complexity Theory

## Dissertation

**zur Erlangung des akademischen Grades**
**doctor rerum naturalium (Dr. rer. nat.)**

vorgelegt dem Rat der
Fakultät für Mathematik und Informatik
der Friedrich-Schiller-Universität Jena

von Diplom-Mathematiker
Jörg Rothe
geb. am 1. November 1966 in Erfurt

| | |
|---|---|
| Gutachter: | Prof. Dr. Gerd Wechsung |
| | Prof. Lane A. Hemaspaandra |
| | Prof. Dr. Uwe Schöning |

| | |
|---|---|
| Tag des Rigorosums: | 5. Oktober 1995 |
| Tag der öffentlichen Verteidigung: | 12. Oktober 1995 |

*To my parents*

# Acknowledgments

Professor Gerd Wechsung, my advisor, has been the most influential and decisive person regarding my scientific career. He not only nurtured my pleasure in mathematics when I was an undergraduate student but also introduced me to the joy and challenge of doing research. I very much appreciate his inspiring and enthusiastic way of creating and formalizing ideas and his graceful style of holding lectures as well as his humane warmth, grace, and optimism. I am particularly grateful to him for bringing about my collaboration with Professor Lane Hemaspaandra and, in general, for his constant encouragement and support.

I am very grateful to Professor Lane Hemaspaandra for his kind guidance, insightful advice, and seminal collaboration during my one year stay at the University of Rochester and ever since. His scientific enthusiasm and his flood of wonderful ideas, comments, and suggestions that contributed to the research described in this work have been invaluable for this thesis and, in fact, made it possible at all. Particularly, I am much indebted to him for having me take part in, and contribute to, his research projects, which eventually led to the first three sections of Chapter 3 and to Chapter 5 of this thesis. From him I learned that striking out in new directions from the solid foundation of being well versed in one's field, combined with the American virtue, *speed in getting done whatever one aims at*, is the most crucial prerequisite for successful work. In addition, I thank him for his financial support that allowed me to present the results contained in the fourth section of Chapter 3 at the Sixth International Conference on Computing and Information in Peterborough, Ontario, in May, 1994.

I thank Professor Lane Hemaspaandra and Professor Uwe Schöning for kindly agreeing to serve—in addition to my advisor, Professor Wechsung—as referees for this thesis.

I further thank all my coauthors, including Lane Hemaspaandra, Osamu Watanabe, Zhigen Jiang, Rajesh Rao, Gerd Wechsung, and Jörg Vogel, in Rochester, Tokyo, and

Jena for our wonderful collaboration in many projects and for their generous permission to present the results of our joint papers in this thesis. In particular, I'd like to mention that Osamu Watanabe's idea of looking at Buhrman, E. Hemaspaandra, and Longpré's tally encoding of sparse sets ignited the research described in Chapter 4 of this thesis, and I again have to thank Lane Hemaspaandra for proposing the investigation of the upward separation technique regarding promise classes. I also thank Arfst Nickelsen and Johannes Köbler for their permission to state and cite still unpublished results of theirs in this work.

Furthermore, I thank all my current and former colleagues, Gerhard Lischke, Jörg Vogel (who was supervising my diploma thesis in 1991), Dieter Kratsch, Haiko Müller, Arfst Nickelsen, Harald Hempel, Johannes Waldmann, Peter Sauer, Dietmar Schuchardt, Zhigen Jiang, Rajesh Rao, Polly Pook, Marc Light, Mitsunori Ogihara, Yenjo Han, Ioan Macarie, and Marius Zimand, from Jena and Rochester. In particular, I'd like to express my gratitude to Johannes Waldmann, Dieter Kratsch, and Haiko Müller for spending much of their time with explaining to me the subtleties of the local Jena LaTeX installation.

The German Academic Exchange Service (DAAD) generously supported with a fellowship my research at the University of Rochester, and the American National Science Foundation (NSF) also supported in part my research under grants NSF-CCR-8957604 and NSF-CCR-9322513.

For a lifetime of understanding, love, and advice, I'm grateful to my family, especially to my parents, to all my friends in Jena and Rochester, and, chiefly, to my wife, Irene, for her great patience and love during the hectic period in which this thesis was written and for the wonderful time we together enjoyed in Rochester and Jena.

# Contents

# List of Figures

# Chapter 1

# Introduction

Structural complexity theory is the study of the structural properties of, and the relationships between, complexity classes. Each complexity class collects similarly structured problems and is represented by a family of algorithms that decide (or accept) the problems in the class. For example, the class P (which was first perceived in [Cob64, Edm65] as the most sensible formal embodiment of the informal term "feasible" computation) collects all problems that can be decided by deterministic polynomial-time bounded Turing machines (DPMs), and NP [Coo71, Lev73] is the class of all sets that are accepted by nondeterministic polynomial-time bounded Turing machines (NPMs).

In terms of their underlying families of algorithms, complexity classes embody various computational paradigms such as probabilistic computation, alternating computation, counting-based computation, unambiguous computation, etc. In many cases, such computational paradigms can be formalized by appropriate modifications of the nondeterministic acceptance mechanism. That is, given an NPM running on some problem instance as input, the *machine* may decide on each of its computation paths whether the input is accepted or rejected on that path, yet *we* decide, looking at the whole tree of all paths of this computation, whether or not the machine accepts its input. In this way, a certain acceptance behavior of NPMs is fixed. For example, probabilistic polynomial-time Turing machines [Gil77, Sim75] may be viewed as NPMs that accept an input if and only if more than half of its paths accept. Alternating polynomial-time Turing machines [CKS81] characterize (for a fixed number of alternations) the levels of the polynomial hierarchy PH [MS72, Sto77] and (for an unbounded number of alternations) the class of sets decidable in polynomial space.

The levels of the Boolean hierarchy over NP are computationally formalized by machines with an appropriate (so-called "chain-respecting") acceptance type [Wec85, GW86]. Finally, a rich spectrum of complexity classes is based on counting the accepting paths of NPMs [Val79, Hem87, GW86, GW87, Wag86, Tor88, Tod91, FFK94].

Unambiguous polynomial time [Val76], denoted by UP, is defined via NPMs that on no input have more than one accepting computation path. FewP [All86] is the class of sets that are accepted by NPMs that on no input have more than polynomially many accepting computations. Clearly, $\mathrm{P} \subseteq \mathrm{UP} \subseteq \mathrm{FewP} \subseteq \mathrm{NP}$. Classes such as UP and FewP are called *promise classes*, since their machines (having both an acceptance criterion and a rejection criterion that is *more restrictive* than the logical negation of the acceptance criterion) "promise" that on all inputs exactly one of the two criteria holds and all known acceptance/rejection criteria for the class also share the property that the rejection criterion is more restrictive than the logical negation of the acceptance criterion. *Promise classes are the main focus of attention in this thesis. In particular, we study to what extent, if any, results for the thoroughly investigated non-promise class NP carry over to the promise classes UP and FewP.*

The study of UP is crucial in both cryptography and structural complexity theory. There has been a long line of research regarding UP [Val76, Rac82, GS88, HH88, HH91, Wat88, Wat91]. To pinpoint some of the most important results about UP, we mention the following. Grollmann and Selman [GS88] have shown that "one-way functions" exist if and only if $\mathrm{P} \neq \mathrm{UP}$. (Informally speaking, a one-way function is one that is easy to compute but hard to invert.) It is not known whether UP has complete sets. Hartmanis and Hemachandra prove there exists an oracle $A$ such that $\mathrm{UP}^A$ has no complete set, and there exists an oracle $B$ such that $\mathrm{P}^B \neq \mathrm{UP}^B \neq \mathrm{NP}^B$ and yet $\mathrm{UP}^B$ does have complete sets [HH88]. They also provide unrelativized evidence that UP is unlikely to have complete sets: if UP has complete sets, then it has complete sets of the form $\mathrm{SAT} \cap A$, where $A$ is a set in P and SAT is the satisfiability problem (i.e., "Given a Boolean formula $f$, is $f$ satisfiable?") [HH88]. Regarding FewP, Allender and Rubinstein [AR88] prove that $\mathrm{P} \neq \mathrm{FewP}$ if and only if there exist sparse sets in P that are not P-printable [HY84],[1] a notion arising in the study of generalized Kolmogorov complexity and data compression.

[1]A set $S$ is *sparse* if there is a polynomial $p$ such that for each length $n$, there are at most $p(n)$ elements of length at most $n$ in $S$. A set $S$ is P-*printable* if there is a DPM $M$ such that for each length $n$, $M$ on input $1^n$ prints all elements of $S$ having length at most $n$.

Chapter 2 gives the notations to be used in this thesis. The definitions of the complexity classes considered in this work are briefly reviewed and some technical points are discussed.

In Chapter 3 and Chapter 4, we study, for the promise classes UP and FewP, some topics that have been intensely studied for NP: Boolean hierarchies, the consequences of the existence of sparse Turing-complete sets, and upward separation. Unfortunately, as is often the case, the results for NP draw on special properties of NP that do not seem to carry over straightforwardly to UP or FewP. For example, NP is easily seen to be closed both under union and intersection, whereas UP is closed under intersection but is not known to be closed under union. Also, NP has complete sets (SAT being the most prominent example), whereas neither UP nor FewP are known to have complete sets.

For the Boolean hierarchy over NP (and more generally over any class containing $\Sigma^*$ and $\emptyset$ and closed under union and intersection), a large number of definitions are known to be equivalent. For example, for NP, all the following coincide [CGH$^+$88, CGH$^+$89, KSW87]: the Boolean closure of NP, the Boolean (alternating sums) hierarchy, the nested difference hierarchy, the Hausdorff hierarchy, and the symmetric difference hierarchy. In Section 3.2, we prove that for the symmetric difference hierarchy and the Boolean hierarchy, closure under union is not needed for this claim: For any class $\mathcal{K}$ that contains $\Sigma^*$ and $\emptyset$ and is closed under intersection (such as UP), the symmetric difference hierarchy over $\mathcal{K}$, the Boolean hierarchy over $\mathcal{K}$, and the Boolean closure of $\mathcal{K}$ all are equal. On the other hand, we show that in the UP case the remaining two hierarchies—the Hausdorff hierarchy over UP and the nested difference hierarchy over UP—fail to be equal to the Boolean closure of UP in some relativized worlds. In fact, the failure is relatively severe; we provide relativizations for which even low levels of other Boolean hierarchies over UP—the third level of the symmetric difference hierarchy and the fourth level of the Boolean (alternating sums) hierarchy—fail to be captured by either the Hausdorff hierarchy or the nested difference hierarchy.

The question of whether there exist sparse Turing-complete or Turing-hard sets for NP has been carefully investigated in the literature [KL80, Hop81, KS85, BBS86a, Sch86, Kad89] (for reductions less flexible than Turing reductions, this issue has been studied even more intensely; see, e.g., the surveys [You92, HOW92]). The results obtained show that NP has no sparse Turing-complete or Turing-hard sets unless certain complexity-theoretic consequences hold that are considered to be unlikely. For instance, Karp and Lipton prove that if there exist sparse Turing-hard sets for NP, then the polynomial hierarchy collapses to its second level [KL80]. Kadin shows that the assumption of the existence of a sparse Turing-*complete* set in NP implies an even stronger collapse of the polynomial hierarchy [Kad89]. Due to the promise nature of UP (in particular, UP probably lacks complete sets [HH88]), Kadin's proof does not seem to apply to UP. In Section 3.3, we prove that if UP has sparse Turing-complete sets, then the levels of the unambiguous polynomial hierarchy (an unambiguous analog [NR93] of the polynomial hierarchy) are simpler than one would otherwise expect: they "slip down" one level in terms of their location in the promise unambiguous polynomial hierarchy (a promise analog of the unambiguous polynomial hierarchy first defined in [NR93, p. 483]). Using the result of Karp and Lipton, we obtain related results under the weaker assumption that UP has sparse Turing-hard sets. In particular, under this assumption, UP is contained in the second level of the low hierarchy [Sch83].

Chapter 4 studies the application domain of the upward separation technique that has been introduced by Hartmanis to relate certain structural properties of polynomial-time complexity classes to their exponential-time analogs and was first applied to NP [Har83]. Later work revealed the limitations of the technique and identified classes defying upward separation. In particular, it is known that coNP as well as certain promise classes such as BPP, R, and ZPP do not possess upward separation in all relativized worlds [HIS85, HJ93], and it had been suspected [All91] that this was also the case for other promise classes such as UP and FewP. We refute this conjecture for the FewP case by proving that FewP *does* display upward separation, thus providing the first upward separation result for a promise class. In fact, this follows from a more general result the proof of which heavily draws on Buhrman, E. Hemaspaandra, and Longpré's recently discovered tally encoding of sparse sets [BHL]. As consequences of our main result, we obtain upward separations for various counting classes such as $\oplus$P, co$\mathrm{C}_{=}$P, SPP, and LWPP (see Chapter 2 for the precise definitions of these classes). Some applications and open problems are also discussed.

The investigations in Section 3.4 are motivated by the open question (raised by Toda and Ogiwara in [TO92]) of whether any set in PH randomly reduces to a set in the class SPP. This question is reformulated in the different context of promise problems, which were introduced by Even, Selman, and Yacobi [EY80, ESY84] in the theory of public-key cryptosystems. Informally, their framework for promise problems relaxes the strict requirement (which applies to the promise classes UP, FewP, or SPP considered above) that some promise-breaking input for a machine $M$ immediately invalidates $M$'s ability to represent the class:

promise-breaking inputs to an algorithm solving a promise problem are allowed; if the promise is not met for some input, however, the algorithm may return an incorrect answer and is thus not reliable. We introduce an analog of SPP in this setting, denoted by $\mathcal{SPP}$, and prove that $\mathcal{SPP}$ indeed is hard for the polynomial hierarchy w.r.t. random reductions, thus generalizing the corresponding result of Valiant and Vazirani for NP [VV86] to all of PH. The original question of Toda and Ogiwara, however, remains unresolved.

Finally, in Chapter 5, we turn to the concept of selectivity in complexity theory. Selman introduced the P-selective sets [Sel79] as the complexity-theoretic analogs of Jockusch's semi-recursive sets [Joc68]. Informally, a set is P-selective if there is a polynomial-time computable function (called a P-selector) that, given any two inputs, outputs one that is logically no less likely to be in the set than the other. In this way, a P-selector performs a "semi-decision" for its set. There are several generalizations of P-selectivity: Ko's "weak P-selectivity" [Ko83], Amir, Beigel, and Gasarch's "non-p-superterse sets" [ABG90] (called "approximable sets" in [BKS94]), and Ogihara's "polynomial-time membership comparable sets" [Ogi94]. In Chapter 5, we introduce a generalization of P-selectivity that is based on the "promise idea" in the sense that if a certain promise is not satisfied, then the selector may output an arbitrary subset of the inputs. Depending on parameters that quantify the "amount of promise," we obtain a selectivity hierarchy, denoted by SH, which we prove does not collapse. In Section 5.2, we study the internal structure and the properties of SH and completely establish, in terms of incomparability and strict inclusion, the relations between our generalized selectivity classes and Ogihara's classes of polynomial-time membership comparable sets. Although SH is a strict hierarchy, we show that the core results holding for the P-selective sets, and proving them structurally simply, also hold for SH. In particular, all sets in SH have small circuits; the NP sets in SH are in $\mathrm{Low}_2$, the second level of the low hierarchy within NP [Sch83]; and SAT cannot be in SH unless $\mathrm{P} = \mathrm{NP}$.

Though the P-selective sets are in $\mathrm{EL}_2$, the second level of the extended low hierarchy [BBS86b], we prove in Section 5.3 that not all sparse sets in SH are in $\mathrm{EL}_2$. This is the strongest known $\mathrm{EL}_2$ lower bound, strengthening the result that P/poly, and indeed SPARSE, is not contained in $\mathrm{EL}_2$ [AH92]. Relatedly, we prove that the join of sets may actually be simpler than the sets themselves: there exist sets that are not in $\mathrm{EL}_2$, yet their join *is* in $\mathrm{EL}_2$. That is, *in terms of extended lowness, the join operator can lower complexity*. We also prove that $\mathrm{EL}_2$ is not closed under union or intersection.

Finally, it is known that the P-selective sets are not closed under union or intersection [HJ]. However, in Section 5.4, we provide an extended selectivity hierarchy that is based on SH and is large enough to capture those closures of the P-selective sets, and yet, in contrast with the P-mc classes, is refined enough to distinguish them.

The results of the fourth section of Chapter 3 have been presented at the *Sixth International Conference on Computing and Information* (*ICCI'94*) [Rot95] in Peterborough, Ontario, and the results of the second section of Chapter 3 have been presented at the *First Annual International Computing and Combinatorics Conference* (*COCOON'95*) [HR95] in Xi'an, China. The first three sections of Chapter 3 will appear in *SIAM Journal on Computing* [HR]. Chapter 4 has been published in *Information Processing Letters* [RRW94], and the results of Chapter 5 have been submitted for publication (a technical report is available as [HJRW95]).

# Chapter 2

# Notations

In this chapter, we fix notations and introduce basic concepts and definitions. In general, we adopt the standard notations of Hopcroft and Ullman [HU79]. We assume that the reader is familiar with the basic concepts of structural complexity theory.

## 2.1 Strings, Sets, Functions, and Boolean Operations

Fix the alphabet $\Sigma = \{0, 1\}$. We consider sets (sometimes called languages) of strings over $\Sigma$. $\Sigma^*$ is the set of all strings over $\Sigma$. For each string $x \in \Sigma^*$, $|x|$ denotes the length of $x$. For $k \geq 1$ and any string $x$, let $x^k \stackrel{\text{df}}{=} x \cdot x^{k-1}$, where $x^0 \stackrel{\text{df}}{=} \epsilon$ is the empty string and $\cdot$ denotes the concatenation of strings. $\mathfrak{P}(\Sigma^*)$ is the class of all sets of strings over $\Sigma$. For any set $L \subseteq \Sigma^*$, $\|L\|$ represents the cardinality of $L$, and $\overline{L} \stackrel{\text{df}}{=} \Sigma^* - L$ denotes the complement of $L$ in $\Sigma^*$. $L^{=n}$ ($L^{\leq n}$) is the set of all strings in $L$ having length $n$ (less than or equal to $n$). Let $\Sigma^n$ and $\Sigma^{\leq n}$ be shorthands for $(\Sigma^*)^{=n}$ and $(\Sigma^*)^{\leq n}$, respectively. Let $\mathbb{Z}$ ($\mathbb{N}$ and $\mathbb{N}^+$, respectively) denote the set of integers (non-negative integers and positive integers). $\mathbb{P}\mathrm{ol}$ is the set of all polynomials over $\mathbb{N}$ in one variable. For any function $f$ from $\mathbb{N}$ into $\mathbb{N}$, define $\mathcal{O}(f)$ as the set of all functions $g$ from $\mathbb{N}$ into $\mathbb{N}$ such that for some real constant $r > 0$ and for all but finitely many $n$, $g(n) < r \cdot f(n)$. For any real number $r$, let $\lceil r \rceil$ ($\lfloor r \rfloor$) denote the least (largest) integer $\geq r$ ($\leq r$).

For sets $A$ and $B$, their *join*, $A \oplus B$, is $\{0x \mid x \in A\} \cup \{1x \mid x \in B\}$, and the Boolean operations *symmetric difference* (also called exclusive-or) and *nxor* (also called equivalence) are defined as $A \Delta B \stackrel{\text{df}}{=} (A \cap \overline{B}) \cup (\overline{A} \cap B)$ and $A \overline{\Delta} B \stackrel{\text{df}}{=} (A \cap B) \cup (\overline{A} \cap \overline{B})$. For any class $\mathcal{K}$, define $\mathrm{co}\mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid \overline{L} \in \mathcal{K}\}$ (which occasionally is denoted "co · $\mathcal{K}$"), and let $\mathrm{BC}(\mathcal{K})$ denote the *Boolean algebra generated by* $\mathcal{K}$, i.e., the smallest class containing $\mathcal{K}$ and closed under all Boolean operations. For classes $\mathcal{C}$ and $\mathcal{D}$ of sets, define

$$\begin{array}{lcllcl}
\mathcal{C} \bigwedge \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A \cap B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}, & \mathcal{C}\, \boldsymbol{\Delta}\, \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A \,\Delta\, B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}, \\
\mathcal{C} \bigvee \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A \cup B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}, & \mathcal{C}\, \overline{\boldsymbol{\Delta}}\, \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A \,\overline{\Delta}\, B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}, \\
\mathcal{C} \bigoplus \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A \oplus B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}, & \mathcal{C} - \mathcal{D} & \stackrel{\mathrm{df}}{=} & \{A - B \mid A \in \mathcal{C} \wedge B \in \mathcal{D}\}.
\end{array}$$

For $k$ sets $A_1, \ldots, A_k$, the join extends to $A_1 \oplus \cdots \oplus A_k \stackrel{\mathrm{df}}{=} \bigcup_{1 \leq i \leq k} \{\underline{i}x \mid x \in A_i\}$, where $\underline{i}$ is the bit pattern of $\lceil \log k \rceil$ bits representing $i$ in binary (and the logarithm is base 2). We write $\bigoplus_k(\mathcal{C}) \stackrel{\mathrm{df}}{=} \{A_1 \oplus \cdots \oplus A_k \mid (\forall i : 1 \leq i \leq k)\, [A_i \in \mathcal{C}]\}$. Similarly, we use the shorthands $\bigwedge_k(\mathcal{C})$ and $\bigvee_k(\mathcal{C})$ in an analogous way.

For any set $L$, let $\chi_L$ denote the *characteristic function of* $L$, i.e., $\chi_L(w) = 1$ if $w \in L$, and $\chi_L(w) = 0$ if $w \notin L$. The *census function of* $L$ is defined by $\mathrm{census}_L(0^n) \stackrel{\mathrm{df}}{=} \|L^{\leq n}\|$. A set $L$ is said to be $d$-sparse (or of density $d$) if $d$ is a function such that for any $n$, $\mathrm{census}_L(0^n) \leq d(n)$; call $L$ *sparse* if $L$ is $d$-sparse for some $d \in \mathbb{P}\mathrm{ol}$. Let SPARSE denote the class of all sparse sets. A set $T$ is said to be *tally* if $T \subseteq 0^*$. To encode a pair of strings, we use a polynomial-time computable, one-one, onto pairing function, $\langle \cdot, \cdot \rangle : \Sigma^* \times \Sigma^* \to \Sigma^*$, that has polynomial-time computable inverses; this notion is extended to encode every $m$-tuple of strings, in the standard way. We simply write $f(x_1, \ldots, x_m)$ instead of $f(\langle x_1, \ldots, x_m \rangle)$—we won't consider any functions on $(\Sigma^*)^m$, $m > 1$, so this causes no problems. Using the standard correspondence between $\Sigma^*$ and $\mathbb{N}$, we will view $\langle \cdot, \cdot \rangle$ also as a pairing function mapping $\mathbb{N} \times \mathbb{N}$ onto $\mathbb{N}$. Let $\leq_{\mathrm{lex}}$ denote the standard *quasi-lexicographical ordering* on $\Sigma^*$, i.e., for strings $x$ and $y$, $x \leq_{\mathrm{lex}} y$ if either $x = y$, or $|x| < |y|$, or $(|x| = |y|$ and there exists some $z \in \Sigma^*$ such that $x = z0u$ and $y = z1v)$. If $x \leq_{\mathrm{lex}} y$ but $x \neq y$, we write $x <_{\mathrm{lex}} y$.

## 2.2 Machines and Reducibilities

Our model of computation is the (multi-tape) Turing machine (see [HU79, Chapter 7]). A Turing machine (TM, for short) can work deterministically (DTM) or nondeterministically (NTM). Although all NTMs considered in this thesis are acceptors, a DTM may be either an acceptor or a transducer. A transducer is a DTM that computes functions from $\Sigma^*$ into $\Sigma^*$ (rather than accepting sets of strings), where the function value computed is written on an output tape. We also consider (deterministic and nondeterministic) oracle TMs—as this notion is standard, we refer for details to the literature [BGS75] [HU79, Chapter 8].

In complexity theory, one is interested in the computational power of TMs having bounds imposed on their computational resources (such as time, space, etc.). This thesis focuses on the *time* complexity of TMs only, and we denote by DTIME$[t(n)]$ (respectively, NTIME$[t(n)]$) the class of all sets accepted by some $t(n)$-time bounded DTM (NTM). As is standard, E will denote $\bigcup_{c\geq 0}$DTIME$[2^{cn}]$, and NE will denote $\bigcup_{c\geq 0}$NTIME$[2^{cn}]$.

We will abbreviate "polynomial-time deterministic (nondeterministic) Turing machine" by DPM (NPM ). An *unambiguous* (sometimes called categorical) polynomial-time Turing machine (UPM) is an NPM that on no input has more than one accepting computation path [Val76]. For the respective oracle machines we use the shorthands DPOM, NPOM, and UPOM. Note, crucially, that whether a machine is categorical or not depends on its oracle. In fact, it is well known that machines that are categorical with respect to all oracles accept only easy languages [HH90] and create a polynomial hierarchy[1] analog that is completely contained in a low level of the polynomial hierarchy (Allender and Hemaspaandra as cited in [HR92]). Thus, when we speak of a UPOM, we will simply mean an NPOM that, with the oracle the machine has in the context being discussed, happens to be categorical.

For any TM $M$, $L(M)$ denotes the set of strings accepted by $M$, and the notation $M(x)$ means "$M$ on input $x$." For any oracle TM $M$ and any oracle set $A$, $L(M^A)$ denotes the set of strings accepted by $M$ relative to $A$, and the notation $M^A(x)$ means "$M^A$ on input $x$." Without loss of generality, we assume each NPM and NPOM (in our standard enumeration of such machines) $M$ has the property that for every $n$, there is an integer $\ell_n$ such that, for every $x$ of length $n$, every path of $M(x)$ is of length $\ell_n$ and all paths of length $\ell_n$ exist in the computation of $M(x)$, and furthermore, in the case of oracle TMs, that $\ell_n$ is independent of the oracle. NPMs meeting these requirements are said to be *normalized*. Unless otherwise stated, all NPMs considered in this thesis are required to be normalized.

FP denotes the class of functions computed by polynomial-time transducers. Let $A$ and $B$ be sets. $A$ is *many-one reducible* to $B$ (denoted by $A \leq_m^p B$) if and only if there is an FP function $f$ such that $A = \{x \mid f(x) \in B\}$. $A$ is *Turing reducible* to $B$ (denoted by $A \leq_T^p B$ or $A \in P^B$) if and only if there is a DPOM $M$ such that $A = L(M^B)$. $A$ is *truth-table reducible* to $B$ (denoted by $A \leq_{tt}^p B$) if $A \leq_T^p B$ via a DPOM $M$ satisfying that for each input $x$, all oracle queries are asked in a "nonadaptive" manner, i.e., $M(x)$ first computes

[1]The polynomial hierarchy is defined in Definition 2.3.4 on page 12.

a list of all queries $q_1, \ldots, q_k$, where $k \in \mathrm{FP}$ depends on $x$, and a $k$-ary truth-table $\alpha$, and accepts $x$ if and only if $\alpha(\chi_B(q_1), \ldots, \chi_B(q_k))$ evaluates to true. For the definition of special truth-table reductions such as bounded truth-table reductions, conjunctive or disjunctive truth-table reductions, we refer to [LLS75]. Other reducibilities will be defined later in this thesis. Define $\mathfrak{R}_r^t(\mathcal{C}) \stackrel{\text{df}}{=} \{L \mid (\exists C \in \mathcal{C})\,[L \leq_r^t C]\}$ for any class $\mathcal{C}$ and for any $r$ and $t$ for which the reducibility $\leq_r^t$ is defined. $\mathcal{C}$ is said to be *closed* under $\leq_r^t$ if $\mathfrak{R}_r^t(\mathcal{C}) \subseteq \mathcal{C}$. A set $B$ is *Turing-hard* for a complexity class $\mathcal{C}$ if for all $A \in \mathcal{C}$, $A \leq_T^p B$. A set $B$ is *Turing-complete* for $\mathcal{C}$ if $B$ is Turing-hard for $\mathcal{C}$ and $B \in \mathcal{C}$.

## 2.3 Complexity Classes and Operators

P (respectively, NP) is the class of all sets that are accepted by some DPM (NPM). Many interesting polynomial-time complexity classes reflecting various computational paradigms such as unambiguous computation, probabilistic computation, etc. can be defined in terms of NPMs whose particular acceptance mode corresponds to the respective paradigm. For instance, UP [Val76] (unambiguous polynomial time) is defined to be the class of all sets that are accepted by some UPM. More generally, in order to refer to some NPM (or NPOM) whose specific mode of acceptance defines a class $\mathcal{C}$ (or the relativized version of $\mathcal{C}$), we shall use the term "$\mathcal{C}$ machine" ("$\mathcal{C}$ oracle machine"). $\mathcal{C}^{\mathcal{B}}$ denotes the class of all sets that are accepted by some $\mathcal{C}$ oracle machine accessing an oracle set from $\mathcal{B}$. As such modifications of the acceptance behavior of NPMs are usually related to the number of accepting (or to the number of accepting and rejecting) computation paths, we will below define some of the complexity classes of interest to us in this work via #P functions [Val79] and GapP functions [FFK91, Gup91]. Moreover, we seize this opportunity to introduce the common operator notation, which will sometimes be used as an alternative to machine-based notations.

**Definition 2.3.1** Let $\mathcal{K}$ be any class of sets, and let $f$ be a function from $\Sigma^*$ into $\mathbb{Z}$.

1. $f \in \mathrm{NUM} \cdot \mathcal{K}$ if and only if
   $(\exists A \in \mathcal{K})\,(\exists p \in \mathbb{P}\mathrm{ol})\,(\forall x)\,[f(x) = \|\{y \mid \langle x, y\rangle \in A \wedge |y| = p(|x|)\}\|]$.

2. $f \in \mathrm{GAP} \cdot \mathcal{K}$ if and only if $(\exists A \in \mathcal{K})\, (\exists p \in \mathbb{P}\mathrm{ol})\, (\forall x)$
$[f(x) = \frac{1}{2}(\|\{y \mid \langle x, y\rangle \in A \wedge |y| = p(|x|)\}\| - \|\{y \mid \langle x, y\rangle \notin A \wedge |y| = p(|x|)\}\|)]$.[2]

3. $\exists \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{NUM} \cdot \mathcal{K})\, [L = \{x \mid f(x) > 0\}]\}$.

4. $\forall \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \mathrm{co} \cdot \exists \cdot \mathrm{co} \cdot \mathcal{K}$.

5. $\mathrm{C} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{GAP} \cdot \mathcal{K})\, [L = \{x \mid f(x) > 0\}]\}$.

6. $\mathrm{C}_{=} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{GAP} \cdot \mathcal{K})\, [L = \{x \mid f(x) = 0\}]\}$.

7. $\oplus \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{GAP} \cdot \mathcal{K})\, [L = \{x \mid f(x) \equiv 0 (\mathrm{mod}\ 2)\}]\}$.

8. $\mathrm{SP} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{GAP} \cdot \mathcal{K})\, (\forall w)\, [\chi_L(w) = f(w)]\}$.

9. $\mathrm{BP} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \left\{L \,\middle|\, (\exists A \in \mathcal{K})\, (\exists p \in \mathbb{P}\mathrm{ol})\, (\forall w)\, \left[\Pr_{p(|w|)}[x \mid \chi_L(w) = \chi_A(w, x)] \geq \frac{3}{4}\right]\right\}$.[3]

10. $\mathrm{RP} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=}$
$$\left\{L \,\middle|\, (\exists A \in \mathcal{K})\, (\exists p \in \mathbb{P}\mathrm{ol})\, (\forall x) \left[\begin{array}{lll} x \in L & \Longrightarrow & \Pr_{p(|x|)}[y \mid \langle x, y\rangle \in A] \geq \frac{3}{4} \\ x \notin L & \Longrightarrow & \Pr_{p(|x|)}[y \mid \langle x, y\rangle \in A] = 0 \end{array}\right] \right\}.$$

11. $\mathrm{ZP} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \mathrm{RP} \cdot \mathcal{K} \cap \mathrm{co}(\mathrm{RP} \cdot \mathcal{K})$.

**Remark 2.3.2** 1. Clearly, $\mathrm{NUM} \cdot \mathrm{P} = \#\mathrm{P}$ ([Val79]; the NUM operator was first defined in [Tod91]) and $\mathrm{GAP} \cdot \mathrm{P} = \mathrm{GapP}$ [FFK91]. GapP is the closure of #P under subtraction.

2. For $\mathcal{K} = \mathrm{P}$, we obtain in Parts 3 to 11 of Definition 2.3.1 above the classes NP, coNP, PP [Sim75, Gil77], $\mathrm{C}_{=}\mathrm{P}$ [Sim75, Wag86], $\oplus$P [PZ83, GP86], SPP ([FFK91], independently defined in [OH90], where it was called XP), BPP, R, and ZPP ([Gil77]; the class R was called VPP in Gill's work). Note that SPP is the "gap analog" of UP, and PP can similarly be viewed as the "gap analog" of NP.

[2] The factor $\frac{1}{2}$ is required to keep $f$ from having even values only, since this would be rather an unnatural property. This requirement is just a technical one and doesn't cause any loss of generality.

[3] For a predicate $Q$ over strings, let $\Pr_m[w \mid Q(w)] \stackrel{\mathrm{df}}{=} \|\{w \mid Q(w)\}\| \cdot 2^{-m}$ denote the probability that $Q(w)$ is true, where $w \in \Sigma^m$ is chosen at random under the uniform distribution.

3. As noted in [Gup93], the classes $\mathrm{co}(\mathrm{RP}\cdot\mathcal{K})$ and $(\mathrm{coRP})\cdot\mathcal{K}$, where the latter is defined as

$$\left\{ L \;\middle|\; (\exists A \in \mathcal{K})\,(\exists p \in \mathbb{P}\mathrm{ol})\,(\forall x) \left[ \begin{array}{l} x \notin L \implies \Pr_{p(|x|)}[y \mid \langle x, y\rangle \notin A] \geq \frac{3}{4} \\ x \in L \implies \Pr_{p(|x|)}[y \mid \langle x, y\rangle \in A] = 1 \end{array} \right] \right\},$$

probably differ if the class $\mathcal{K}$ is not closed under complementation. In particular, $(\mathrm{coRP}) \cdot \mathrm{C_{=}P} = \mathrm{C_{=}P}$, whereas it is not known whether $\mathrm{co} \cdot (\mathrm{RP} \cdot \mathrm{C_{=}P})$ is equal to $\mathrm{C_{=}P}$.

4. Polynomial-time bounded operators such as those defined above, which yield some class $\mathcal{C}$ when applied to P, formalize a generalized type of many-one reducibility $\leq_m^{\mathcal{C}}$. For example, the "polynomial-time bounded exist quantifier" [MS72, Sto77] expresses the polynomial-time nondeterministic many-one reducibility, $\leq_m^{\mathrm{NP}}$, in the sense that $\mathfrak{R}_m^{\mathrm{NP}}(\mathcal{K}) = \exists \cdot \mathcal{K}$; the polynomial-time randomized many-one reducibility with bounded error, $\leq_m^{\mathrm{BPP}}$, is formalized by the BP operator [Sch89, Tod91], i.e., $\mathfrak{R}_m^{\mathrm{BPP}}(\mathcal{K}) = \mathrm{BP} \cdot \mathcal{K}$; etc.

**Definition 2.3.3** [KL80] P/poly denotes the class of sets $L$ for which there exist a set $A \in \mathrm{P}$ and a polynomially length-bounded function $h : \Sigma^* \to \Sigma^*$ such that for every $x$, it holds that $x \in L$ if and only if $\langle x, h(0^{|x|})\rangle \in A$.

**Definition 2.3.4** The *polynomial hierarchy* [MS72, Sto77] is defined as follows: $\Sigma_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\Delta_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\Sigma_k^p \stackrel{\text{df}}{=} \mathrm{NP}^{\Sigma_{k-1}^p}$, $\Pi_k^p \stackrel{\text{df}}{=} \mathrm{co}\Sigma_k^p$, $\Delta_k^p \stackrel{\text{df}}{=} \mathrm{P}^{\Sigma_{k-1}^p}$, $k \geq 1$, and $\mathrm{PH} \stackrel{\text{df}}{=} \bigcup_{k\geq 0} \Sigma_k^p$.

**Definition 2.3.5**

1. [Sch83] For each $k \geq 1$, define $\mathrm{Low}_k \stackrel{\text{df}}{=} \{L \in \mathrm{NP} \mid \Sigma_k^{p,L} = \Sigma_k^p\}$.

2. [BBS86b, LS94] For each $k \geq 2$, define $\mathrm{EL}_k \stackrel{\text{df}}{=} \{L \mid \Sigma_k^{p,L} = \Sigma_{k-1}^{p,\mathrm{SAT}\oplus L}\}$, and for each $k \geq 3$, define $\mathrm{EL\Theta}_k \stackrel{\text{df}}{=} \{L \mid \mathrm{P}^{(\Sigma_{k-1}^{p,L})[\log n]} \subseteq \mathrm{P}^{(\Sigma_{k-2}^{p,\mathrm{SAT}\oplus L})[\log n]}\}$. The $[\log n]$ indicates that at most $\mathcal{O}(\log n)$ queries are made to the oracle.

More generally, a set $L$ is said to be *low* for a (relativized) complexity class $\mathcal{C}$ if $\mathcal{C}^L = \mathcal{C}$, i.e., $L$ does not provide $\mathcal{C}$ with any additional computational power when used as oracle by $\mathcal{C}$ oracle machines. Call a class $\mathcal{L}$ of sets low for $\mathcal{C}$ if $\mathcal{C}^L = \mathcal{C}$ holds for each set $L \in \mathcal{L}$. A class $\mathcal{C}$ is said to be *self-low* if $\mathcal{C}^{\mathcal{C}} = \mathcal{C}$.

## 2.4 Promise Classes

Some of the above-defined complexity classes (UP, SPP, BPP, R, ZPP, and NP $\cap$ coNP) are defined by machines with both an acceptance criterion and a rejection criterion (that is *more restrictive* than the logical negation of the acceptance criterion), along with a "promise" that on all inputs exactly one of the two criteria holds (and all known acceptance/rejection criteria for the class also share the property that the rejection criterion is more restrictive than the logical negation of the acceptance criterion). As is standard (at least since [HR92]), we will refer to those classes as "promise classes" in this thesis.[4] Another example of a promise class is the class FewP, which was first defined in [All86]:

$$\mathrm{FewP} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \#\mathrm{P})\ (\exists q \in \mathbb{P}\mathrm{ol})\ [(\forall x)\ [f(x) \leq q(|x|)]\ \wedge\ L = \{x \mid f(x) > 0\}]\}.$$

This definition straightforwardly extends to the definition of the FEW operator applied to any class of sets $\mathcal{K}$:

$$\mathrm{FEW} \cdot \mathcal{K} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{NUM} \cdot \mathcal{K})\ (\exists q \in \mathbb{P}\mathrm{ol})\ [(\forall x)\ [f(x) \leq q(|x|)]\ \wedge\ L = \{x \mid f(x) > 0\}]\}.$$

Fenner, Fortnow, and Kurtz [FFK91] introduced the promise class LWPP as a generalization of SPP:

$$\mathrm{LWPP} \stackrel{\mathrm{df}}{=} \{L \mid (\exists f \in \mathrm{GapP})\ (\exists g \in \mathrm{FP};\ g : \mathbb{N} \to \mathbb{N}^{+})\ (\forall w)\ [g(|w|) \cdot \chi_L(w) = f(w)]\}.$$

A different concept of promise problems was introduced by Even, Selman, and Yacobi in the theory of public-key cryptosystems. To distinguish between the notions, we will refer to collections of promise problems defined in the sense of Even, Selman, and Yacobi as "classes of promise problems" in this thesis, reserving the term "promise class" for collections of decision problems in the above sense. The term "promise problem" will be used exclusively for members of classes of promise problems, while elements of promise classes are called sets. Even, Selman, and Yacobi [EY80, ESY84] define a promise problem

[4] It has been shown in [HHT93] that the "promise" in the definition of $\mathrm{R}_{\mathrm{path}}$, the analog of R in the model of threshold computation [Sim75], is a *trivial* one, i.e., $\mathrm{R}_{\mathrm{path}}$ equals NP and is thus not a promise class in our sense. The proof that $\mathrm{R}_{\mathrm{path}} = \mathrm{NP}$ essentially rests on the fact that threshold machines need not be normalized in general. Since we exclusively consider *normalized* NTMs in this thesis, the informal explanation of promise classes given above suffices.

to be a partial decision problem having the structure

**input** $x$
**promise** $Q(x)$
**property** $R(x)$

where $Q$ and $R$ are (recursive) predicates.[5] That is, on input $x$, an algorithm solving a promise problem $(Q, R)$ has to correctly decide property $R(x)$ if the promise $Q(x)$ is met; otherwise, it can give an incorrect answer. More formally, a set $S$ is said to be a *solution* to $(Q, R)$ if $(\forall x \in \Sigma^*)\,[x \in Q \implies (x \in R \iff x \in S)]$. Let $\mathrm{solns}(Q, R)$ denote the set of all solutions to the promise problem $(Q, R)$. Note that every set of the form $(Q \cap R) \cup X$, where $X \subseteq \overline{Q}$, is a solution of $(Q, R)$. In particular, $R$ is the unique solution to $(\Sigma^*, R)$; thus, the promise problem $(\Sigma^*, R)$ may be identified with the decision problem $R$. For notational convenience, we will write $\mathcal{D} \subseteq \mathcal{P}$ for any class $\mathcal{D}$ of decision problems and any class $\mathcal{P}$ of promise problems if for each set $L \in \mathcal{D}$ the corresponding promise problem $(\Sigma^*, L)$ is in $\mathcal{P}$.

For example, $(\mathrm{1SAT}, \mathrm{SAT})$ is a well-known and intensely studied promise problem (see, e.g., [CHV93, KST92, VV86, Wat92] and the references given therein), where

$$\begin{aligned} \mathrm{SAT} &\stackrel{\mathrm{df}}{=} \{f \mid \text{Boolean formula } f \text{ is satisfiable}\}, \\ \mathrm{1SAT} &\stackrel{\mathrm{df}}{=} \{f \mid \text{Boolean formula } f \text{ has at most one satisfying assignment}\}. \end{aligned}$$

$(\mathrm{1SAT}, \mathrm{SAT})$ is closely related to the class of promise problems $\mathcal{UP}$ ("promise UP"), which is defined as:

$$\mathcal{UP} \stackrel{\mathrm{df}}{=} \{(Q, R) \mid (\exists f \in \#\mathrm{P})\, [Q = \{x \mid f(x) \in \{0, 1\}\} \wedge R = \{x \mid f(x) = 1\}]\}.$$

This definition of $\mathcal{UP}$ is equivalent to the one given in [CHV93]. Watanabe [Wat92] defines a similar notion: A promise problem $(Q, R)$ is *unambiguous* if there exist a solution $X$ in NP and an NPM $M$ accepting $X$ that is unambiguous on $Q$. As noted by Hemaspaandra [Hem94], these two notions are subtly different, since $(\mathrm{HALTINGPROBLEM}, \Sigma^*)$ is an unambiguous promise problem in Watanabe's setting (as it has the solution $\Sigma^*$ and the (deterministic) polynomial-time Turing machine accepting $\Sigma^*$ never has more than one accepting path), yet $(\mathrm{HALTINGPROBLEM}, \Sigma^*) \notin \mathcal{UP}$ (as there is no NPM that has at most one accepting path *exactly* on the HALTINGPROBLEM).

[5]We will identify predicates and sets, i.e., for a predicate $A$ over strings, we will use $A$ also to denote the set $\{x \mid A(x) \text{ is true}\}$, and conversely, set $A$ is identified with the predicate $\chi_A$.

Below we define the "gap analog" of $\mathcal{UP}$, denoted $\mathcal{SPP}$ ("promise SPP"), by introducing the promise operator $\mathcal{SP}$, which yields a class of promise problems when applied to a class of decision problems. In particular, $\mathcal{SPP} = \mathcal{SP} \cdot \mathrm{P}$.

**Definition 2.4.1** Let $\mathcal{K}$ be any class of sets.

$$\mathcal{SP} \cdot \mathcal{K} \stackrel{\text{df}}{=} \{(Q, R) \mid (\exists f \in \mathrm{GAP} \cdot \mathcal{K})\, [Q = \{x \mid f(x) \in \{0, 1\}\} \wedge R = \{x \mid f(x) = 1\}]\}.$$

# Chapter 3

# Unambiguous Computation: Boolean Hierarchies and Sparse Turing-Complete Sets

## 3.1 Introduction

NP and NP-based hierarchies—such as the polynomial hierarchy [MS72, Sto77] and the Boolean hierarchy over NP [CGH$^+$88, CGH$^+$89]—have played such a central role in complexity theory, and have been so thoroughly investigated, that it would be natural to take them as predictors of the behavior of other classes or hierarchies. However, over and over during the past decade it has been shown that NP is a singularly poor predictor of the behavior of other classes (and, to a lesser extent, that hierarchies built on NP are poor predictors of the behavior of other hierarchies).

As examples regarding hierarchies: though the polynomial hierarchy possesses downward separation (that is, if its low levels collapse, then all its levels collapse) [MS72, Sto77], downward separation does not hold "robustly" (i.e., in every relativized world) for the exponential time hierarchy [HIS85, IT89] or for limited-nondeterminism hierarchies ([HJ93], see also [BG94]). As examples regarding UP: NP has $\leq_m^p$-complete sets, but UP does not robustly possess $\leq_m^p$-complete sets [HH88] or even $\leq_T^p$-complete sets [HJV93]; NP positively relativizes, in the sense that it collapses to P if and only if it does so with respect to every tally oracle ([LS86], see also [BBS86a]), but UP does not robustly positively relativize [HR92]; NP has "constructive programming systems," but UP does not robustly have such systems [Reg89]; NP (actually, nondeterministic computation) admits time hierarchy theorems [HS65], but it is an open question whether unambiguous computation has nontrivial time hierarchy theorems; NP displays upward separation (that is, $\mathrm{NP}-\mathrm{P}$ contains sparse sets if and only if $\mathrm{NE} \neq \mathrm{E}$) [HIS85], but it is not known whether UP does (see [HJ93], which shows that R and BPP do not robustly display upward separation, and Chapter 4, which shows that FewP and several related classes do possess upward separation).

In light of the above list of the many ways in which NP parts company with UP, it is clear that we should not merely assume that results for NP hold for UP, but, rather, we must carefully check to see to what extent, if any, results for NP suggest results for UP. In the first two sections of this chapter, we study, for UP, two topics that have been intensely studied for the NP case: the structure of Boolean hierarchies, and the effects of the existence of sparse Turing-complete/Turing-hard sets.

For the Boolean hierarchy over NP [CGH$^+$88, CGH$^+$89], a large number of definitions are known to be equivalent. For example, for NP, all the following coincide [CGH$^+$88]: the Boolean closure of NP, the Boolean (alternating sums) hierarchy, the nested difference hierarchy, and the Hausdorff hierarchy. The symmetric difference hierarchy also characterizes the Boolean closure of NP [KSW87]. In fact, these equalities are known to hold for all classes that contain $\Sigma^*$ and $\emptyset$ and are closed under union and intersection [Hau14, CGH$^+$88, KSW87, BBJ$^+$89]. In Section 3.2, we prove that both the symmetric difference hierarchy (SDH) and the Boolean hierarchy (CH) remain equal to the Boolean closure (BC) *even in the absence of the assumption of closure under union*. That is, for any class $\mathcal{K}$ containing $\Sigma^*$ and $\emptyset$ and closed under intersection (e.g., UP, US [BG82], and DP [PY84]): $\mathrm{SDH}(\mathcal{K}) = \mathrm{CH}(\mathcal{K}) = \mathrm{BC}(\mathcal{K})$. However, for the remaining two hierarchies, we show that not all classes containing $\Sigma^*$ and $\emptyset$ and closed under intersection robustly display equality. In particular, the Hausdorff hierarchy over UP and the nested difference hierarchy over UP both fail to capture the Boolean closure of UP in some relativized worlds. In fact, the failure is relatively severe; we show that even low levels of other Boolean hierarchies over UP—the third level of the symmetric difference hierarchy and the fourth level of the Boolean (alternating sums) hierarchy—fail to be robustly captured by either the Hausdorff hierarchy or the nested difference hierarchy.

The investigations in Sections 3.3 and 3.4 are motivated by certain open problems regarding the classes UP and SPP, where, informally speaking, the promise-like definition of UP and SPP seems to be responsible for the difficulty of the original problem in either case. If some problem appears hard to solve in the context in which it naturally arose, one often tries to reformulate it in another context to tackle it under new conditions. If this happens to succeed, one might then, in light of these new insights, return to deal with the original issue. For example, after each attempt to solve the famous $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$ problem (one way or the other) had failed, Baker, Gill, and Solovay settled it *relative to an oracle* (surprisingly, in both ways) [BGS75], thereby creating an extremely fruitful branch of complexity theory. As another example, though it is still unknown whether or not SAT is Turing-reducible to some set in $\oplus\mathrm{P}$ (which, due to $\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$ [PZ83], is equivalent to the containment question "$\mathrm{NP} \subseteq \oplus\mathrm{P}$?"), Valiant and Vazirani raised and settled the reduction question in the context of *randomized reductions* by showing that each NP set is polynomial-time randomized many-one reducible to a set in $\oplus\mathrm{P}$ [VV86] (in fact, they even prove a technically stronger result that will be discussed in Part 2 of Remark 3.4.2 on page 42). It is worth noting that, in a certain contrast to their result, Torán constructed an oracle relative to which the containment $\mathrm{NP} \subseteq \oplus\mathrm{P}$ does not hold [Tor88].

It is well-known, thanks to the work of Karp and Lipton ([KL80], see also the related references given in Section 3.3), that if NP has sparse Turing-hard (or Turing-complete) sets, then the polynomial hierarchy (PH) collapses. Section 3.3 studies the issue of whether the existence of sparse Turing-hard or Turing-complete sets for UP has similarly unlikely consequences. Unfortunately, the promise-like definition of UP—its unambiguity, the very core of its nature—seems to block any similarly strong claim for UP and the unambiguous polynomial hierarchy, denoted by UPH, which was introduced recently by Niedermeier and Rossmanith [NR93]. Lange, Niedermeier, and Rossmanith [LR94][NR93, p. 483] also define a promise analog of UPH, the *promise unambiguous polynomial hierarchy*, that requires only that oracle computations *actually executed* be unambiguous. This model of access to an oracle from a promise class is known from the literature as "guarded" access [GS88, CHV93].[1] Even though we cannot prove the "clean" UPH analog of the Karp-Lipton result, we establish (in the context of *guardedly* unambiguous oracle access) some results showing that UP is unlikely to have sparse Turing-complete or Turing-hard sets. In particular, if UP has sparse Turing-complete sets, then the levels of the unambiguous polynomial hierarchy are simpler than one would otherwise expect: they "slip down" slightly in terms of their location within the promise unambiguous polynomial hierarchy,

[1] Grollmann and Selman used the term "smart" [GS88] rather than "guarded" [CHV93].

i.e., the $k$th level of UPH is contained in the $(k-1)$st level of the promise unambiguous polynomial hierarchy. If UP has Turing-hard sparse sets, then UP is low for $\mathrm{NP}^{\mathrm{NP}}$; we also provide a generalization of this result related to the promise unambiguous polynomial hierarchy. Furthermore, we show that the same assumption implies that the $k$th level of UPH, where $k \geq 3$, can be accepted via a DPOM given access to both an $\mathrm{NP}^{\mathrm{NP}}$ set and the $(k-1)$st level of the promise unambiguous polynomial hierarchy.

Finally, Section 3.4 studies an issue that is related to the open question of whether the polynomial hierarchy is contained in the class SPP. (Note that the promise unambiguous polynomial hierarchy is contained both in the polynomial hierarchy and in SPP.) Though Toda and Ogiwara have shown that for many counting classes such as PP, $\mathrm{C_{=}P}$, and $\oplus$P, each set in the polynomial hierarchy randomly reduces to some set in the counting class [TO92] (thus generalizing the above-mentioned result of Valiant and Vazirani to all levels of PH), they conjectured that this result is unlikely to hold for SPP also. This conjecture again rests on the promise nature of SPP. However, we will show in Section 3.4 that, in the context of *promise problems* defined in the sense of Even, Selman, and Yacobi [EY80, ESY84], the reduction question can be resolved: Each set in the polynomial hierarchy "randomly reduces" to $\mathcal{SPP}$, where we use Selman's approach [Sel88] to "reductions between promise problems." This supports the conjecture that $\mathcal{SPP}$ indeed is more powerful than SPP.

## 3.2 Boolean Hierarchies over Classes Closed Under Intersection

The Boolean hierarchy is a natural extension of the classes NP [Coo71, Lev73] and $\mathrm{DP} \stackrel{\mathrm{df}}{=} \mathrm{NP} \wedge \mathrm{coNP}$ [PY84]. Both NP and DP contain natural problems, as do the levels of the Boolean hierarchy. For example, graph minimal uncolorability is known to be complete for DP [CM87]. Note that DP clearly is closed under intersection, but is not closed under union unless the polynomial hierarchy collapses (due to [Kad88], see also [CK90b, Cha91]).

**Definition 3.2.1** [CGH+88, KSW87, Hau14] Let $\mathcal{K}$ be any class of sets.

1. The *Boolean ("alternating sums") hierarchy over* $\mathcal{K}$:

$$C_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K}, \quad C_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \begin{cases} C_{k-1}(\mathcal{K}) \vee \mathcal{K} & \text{if } k \text{ odd} \\ C_{k-1}(\mathcal{K}) \wedge \mathrm{co}\mathcal{K} & \text{if } k \text{ even} \end{cases}, \quad k \geq 2,$$

$$\mathrm{CH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 1} \mathrm{C}_k(\mathcal{K}).$$

2. The *nested difference hierarchy over* $\mathcal{K}$:

$$\mathrm{D}_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K},\ \ \mathrm{D}_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K} - \mathrm{D}_{k-1}(\mathcal{K}),\ k \geq 2,\ \ \mathrm{DH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 1} \mathrm{D}_k(\mathcal{K}).$$

3. The *Hausdorff ("union of differences") hierarchy over* $\mathcal{K}$:[2]

$$\mathrm{E}_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K},\ \ \mathrm{E}_2(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K} - \mathcal{K},\ \ \mathrm{E}_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathrm{E}_2(\mathcal{K}) \vee \mathrm{E}_{k-2}(\mathcal{K}),\ \ k > 2,$$

$$\mathrm{EH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 1} \mathrm{E}_k(\mathcal{K}).$$

4. The *symmetric difference hierarchy over* $\mathcal{K}$:

$$\mathrm{SD}_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K},\ \ \mathrm{SD}_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathrm{SD}_{k-1}(\mathcal{K})\, \boldsymbol{\Delta}\, \mathcal{K},\ k \geq 2,\ \ \mathrm{SDH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 1} \mathrm{SD}_k(\mathcal{K}).$$

It is easily seen that for any X chosen from {C, D, E, SD}, if $\mathcal{K}$ contains $\emptyset$ and $\Sigma^*$, then for any $k \geq 1$,

$$\mathrm{X}_k(\mathcal{K}) \cup \mathrm{coX}_k(\mathcal{K}) \subseteq \mathrm{X}_{k+1}(\mathcal{K}) \cap \mathrm{coX}_{k+1}(\mathcal{K}).$$

The following fact is shown by an easy induction on $n$.

**Fact 3.2.2** For every class $\mathcal{K}$ of sets and every $n \geq 1$,

1. $\mathrm{D}_{2n-1}(\mathcal{K}) = \mathrm{coC}_{2n-1}(\mathrm{co}\mathcal{K})$, and

2. $\mathrm{D}_{2n}(\mathcal{K}) = \mathrm{C}_{2n}(\mathrm{co}\mathcal{K})$.

---

[2] Hausdorff hierarchies ([Hau14], see [CGH+88, BBJ+89, GNW90], respectively, for applications to NP, R, and C=P) are interesting both in the case where, as in the definition here, the sets are arbitrary sets from $\mathcal{K}$, and, as is sometimes used in definitions, the sets from $\mathcal{K}$ are required to satisfy additional containment conditions. For classes closed under union and intersection, such as NP, the two definitions are identical, level by level ([Hau14], see also [CGH+88]). In this paper, as, e.g., UP, is not known to be closed under union, the distinction is nontrivial.

**Proof.** The base case holds by definition. Suppose both statements of this fact to be true for $n \geq 1$. Then,

$$\begin{aligned} D_{2n+1}(\mathcal{K}) &= \mathcal{K} \wedge (\text{co}\mathcal{K} \vee D_{2n-1}(\mathcal{K})) &\overset{\text{hyp.}}{=}& \mathcal{K} \wedge (\text{co}\mathcal{K} \vee \text{co}C_{2n-1}(\text{co}\mathcal{K})) \\ &= \mathcal{K} \wedge \text{co}(\mathcal{K} \wedge C_{2n-1}(\text{co}\mathcal{K})) &=& \mathcal{K} \wedge \text{co}C_{2n}(\text{co}\mathcal{K}) \\ &= \text{co}(\text{co}\mathcal{K} \vee C_{2n}(\text{co}\mathcal{K})) &=& \text{co}C_{2n+1}(\text{co}\mathcal{K}) \end{aligned}$$

shows Part 1 of this fact for $n+1$, and

$$D_{2n+2}(\mathcal{K}) = \mathcal{K} - (\mathcal{K} - D_{2n}(\mathcal{K})) \overset{\text{hyp.}}{=} \mathcal{K} \wedge (\text{co}\mathcal{K} \vee C_{2n}(\text{co}\mathcal{K})) = C_{2n+2}(\text{co}\mathcal{K})$$

shows Part 2 of this fact for $n+1$. □

**Corollary 3.2.3** 1. $\text{CH}(\text{UP}) = \text{coCH}(\text{UP}) = \text{DH}(\text{coUP})$, and

2. $\text{CH}(\text{coUP}) = \text{coCH}(\text{coUP}) = \text{DH}(\text{UP})$.

We are interested in the Boolean hierarchies over classes closed under intersection (but perhaps not under union or complementation), such as UP, US, and DP. We state our theorems in terms of the class of primary interest to us, UP. However, many apply to any nontrivial class (i.e., any class containing $\Sigma^*$ and $\emptyset$) closed under intersection (see Theorem 3.2.10). Although it has been proven in [CGH$^+$88] and [KSW87] that all the standard normal forms of Definition 3.2.1 coincide for NP,[3] the situation for UP seems to be different, as UP is probably not closed under union. (The closure of UP under intersection is straightforward.) Thus, all the relations among those normal forms have to be reconsidered for UP.

We first prove that the symmetric difference hierarchy over UP (or any class closed under intersection) equals the Boolean closure. Though Köbler, Schöning, and Wagner [KSW87] proved this for NP, their proof gateways through a class whose proof of equivalence to the Boolean closure uses closure under union, and thus the following result is not implicit in their paper.

**Theorem 3.2.4** $\text{SDH}(\text{UP}) = \text{BC}(\text{UP})$.

[3] Due essentially to its closure under union and intersection, and this reflects a more general behavior of classes closed under union and intersection, as studied by Bertoni et al. ([BBJ$^+$89], see also [Hau14, CGH$^+$88, KSW87, CK90b, Cha91]).

**Proof.** The inclusion from left to right is clear. For the converse inclusion, it is sufficient to show that $\mathrm{SDH}(\mathrm{UP})$ is closed under all Boolean operations, as $\mathrm{BC}(\mathrm{UP})$, by definition, is the smallest class of sets that contains UP and is closed under all Boolean operations. Let $L$ and $L'$ be arbitrary sets in $\mathrm{SDH}(\mathrm{UP})$. Then, for some $k, l \geq 1$, there are sets $A_1, \ldots, A_k, B_1, \ldots, B_l$ in UP representing $L$ and $L'$:

$$L = A_1 \Delta \cdots \Delta A_k \text{ and } L' = B_1 \Delta \cdots \Delta B_l.$$

So

$$L \cap L' = \left(\Delta_{i=1}^{k} A_i\right) \cap \left(\Delta_{j=1}^{l} B_j\right) = \Delta_{i \in \{1,\ldots,k\},\, j \in \{1,\ldots,l\}} (A_i \cap B_j),$$

and since UP is closed under intersection and $\mathrm{SDH}(\mathrm{UP})$ is (trivially) closed under symmetric difference, we clearly have that $L \cap L' \in \mathrm{SDH}(\mathrm{UP})$. Furthermore, since $\overline{L} = \Sigma^* \Delta L$ implies that $\overline{L} \in \mathrm{SDH}(\mathrm{UP})$, $\mathrm{SDH}(\mathrm{UP})$ is closed under complementation. Since all Boolean operations can be represented in terms of complementation and intersection, our proof is complete. □

Next, we show that for any class closed under intersection, instantiated below to the case of UP, the Boolean (alternating sums) hierarchy over the class equals the Boolean closure of the class. Our proof is inspired by the techniques used to prove equality in the case where closure under union may be assumed.

**Theorem 3.2.5** $\mathrm{CH}(\mathrm{UP}) = \mathrm{BC}(\mathrm{UP})$.

**Proof.** We will prove that $\mathrm{SDH}(\mathrm{UP}) \subseteq \mathrm{CH}(\mathrm{UP})$. By Theorem 3.2.4, this will suffice.

Let $L$ be any set in $\mathrm{SDH}(\mathrm{UP})$. Then there is a $k > 1$ (the case $k = 1$ is trivial) such that $L \in \mathrm{SD}_k(\mathrm{UP})$. Let $U_1, \ldots, U_k$ be the witnessing UP sets; that is, $L = U_1 \Delta U_2 \Delta \cdots \Delta U_k$. By the inclusion-exclusion rule, $L$ satisfies the equalities below. For odd $k$,

$$\begin{aligned} L \;=\; & \left(\cdots\left(\left((U_1 \cup U_2 \cup \cdots \cup U_k) \cap \left(\overline{\bigcup_{j_1<j_2} (U_{j_1} \cap U_{j_2})}\right)\right) \cup \right.\right. \\ & \left.\left.\left(\bigcup_{j_1<j_2<j_3} (U_{j_1} \cap U_{j_2} \cap U_{j_3})\right)\right) \cap \cdots \cup \left(\bigcup_{j_1<\cdots<j_k} (U_{j_1} \cap \cdots \cap U_{j_k})\right)\right), \end{aligned}$$

where each subscripted $j$ term must belong to $\{1, \ldots, k\}$. For even $k$, we similarly have:

$$\begin{aligned} L \;=\; & \left(\cdots\left(\left((U_1 \cup U_2 \cup \cdots \cup U_k) \cap \left(\overline{\bigcup_{j_1<j_2} (U_{j_1} \cap U_{j_2})}\right)\right) \cup \right.\right. \\ & \left.\left.\left(\bigcup_{j_1<j_2<j_3} (U_{j_1} \cap U_{j_2} \cap U_{j_3})\right)\right) \cap \cdots \cap \left(\overline{\bigcup_{j_1<\cdots<j_k} (U_{j_1} \cap \cdots \cap U_{j_k})}\right)\right). \end{aligned}$$

For notational convenience, let us use $A_1, \ldots, A_k$ to represent the respective terms in the above expressions (ignoring the complementations). By the closure of UP under intersection, each $A_i$, $1 \le i \le k$, is the union of $\binom{k}{i}$ UP sets $B_{i,1}, \ldots, B_{i,\binom{k}{i}}$. Using the fact that $\emptyset$ is clearly in UP, we can easily turn the union of $n$ arbitrary UP sets (or the intersection of $n$ arbitrary coUP sets) into an alternating sum of $2n-1$ UP sets. So for instance, $A_1 = U_1 \cup U_2 \cup \cdots \cup U_k$ can be written

$$\left(\cdots\left(\left(\left(U_1 \cap \overline{\emptyset}\right) \cup U_2\right) \cap \overline{\emptyset}\right) \cup \cdots \cup U_k\right),$$

call this $C_1$. Clearly, $C_1 \in C_{2k-1}(\mathrm{UP})$. To transform the above representation of $L$ into an alternating sum of UP sets, we need two (trivial) transformations holding for any $m \ge 1$ and for arbitrary sets $S$ and $T_1, \ldots, T_m$:

$$S \cap \left(\overline{T_1 \cup T_2 \cup \cdots \cup T_m}\right) \;=\; \left(\cdots\left(\left(S \cap \overline{T_1}\right) \cap \overline{T_2}\right) \cap \cdots\right) \cap \overline{T_m} \tag{3.1}$$

$$S \cup (T_1 \cup T_2 \cup \cdots \cup T_m) \;=\; (\cdots((S \cup T_1) \cup T_2) \cup \cdots) \cup T_m. \tag{3.2}$$

Using (3.1) with $S = C_1$ and $T_1 = B_{2,1}, \ldots, T_m = B_{2,\binom{k}{2}}$ and the fact that $\emptyset$ is in UP, $A_1 \cap \overline{A_2}$ can be transformed into an alternating sum of UP sets, call this $C_2$. Now apply (3.2) with $S = C_2$ and $T_1 = B_{3,1}, \ldots, T_m = B_{3,\binom{k}{3}}$ to obtain, again using that $\emptyset$ is in UP, an alternating sum $C_3 = \left(A_1 \cap \overline{A_2}\right) \cup A_3$ of UP sets, and so on. Eventually, this procedure of alternately applying (3.1) and (3.2) will yield an alternating sum $C_k$ of sets in UP that equals $L$. Thus, $L \in \mathrm{CH}(\mathrm{UP})$. □

**Corollary 3.2.6** SDH(UP) and CH(UP) are both closed under all Boolean operations.

Note that the proofs of Theorems 3.2.5 and 3.2.4 implicitly give a recurrence yielding an upper bound on the level-wise containments. We find the issue of equality to BC(UP), or lack thereof, to be the central issue, and thus we focus on that. Nonetheless, we point out in the corollary below that losing the assumption of closure under union seems to have exacted a price: though the hierarchies $\mathrm{SDH}(\mathrm{UP})$ and $\mathrm{CH}(\mathrm{UP})$ are indeed equal, the above proof embeds $\mathrm{SD}_k(\mathrm{UP})$ in an exponentially higher level of the C hierarchy. Similarly, the proof of Theorem 3.2.4 embeds $\mathrm{C}_k(\mathrm{UP})$ in an exponentially higher level of $\mathrm{SDH}(\mathrm{UP})$.

**Corollary 3.2.7 (to the proofs of Theorems 3.2.5 and 3.2.4)**

1. For each $k \geq 1$, $\mathrm{SD}_k(\mathrm{UP}) \subseteq \mathrm{C}_{2^{k+1}-k-2}(\mathrm{UP})$.
2. For each $k \geq 1$, $\mathrm{C}_k(\mathrm{UP}) \subseteq \mathrm{SD}_{T(k)}(\mathrm{UP})$, where $T(k) = \begin{cases} 2^k - 1 & \text{if } k \text{ is odd} \\ 2^k - 2 & \text{if } k \text{ is even.} \end{cases}$

**Proof.** For an $\mathrm{SD}_k(\mathrm{UP})$ set $L$ to be placed into the $R(k)$th level of $\mathrm{CH}(\mathrm{UP})$, $L$ is represented (in the proof of Theorem 3.2.5) as an alternating sum of $k$ terms $A_1, \ldots, A_k$, each $A_i$ consisting of $\binom{k}{i}$ UP sets $B_{i,j}$. In the subsequent transformation of $L$ according to the equations (3.1) and (3.2), each $A_i$ requires as many as $\binom{k}{i} - 1$ additional terms $\emptyset$ or $\overline{\emptyset}$, respectively, to be inserted, and each such insertion brings us one level higher in the C hierarchy. Thus,

$$R(k) = \sum_{i=1}^{k} \binom{k}{i} + \left(\binom{k}{i} - 1\right) = -k + 2\sum_{i=1}^{k} \binom{k}{i} = 2^{k+1} - k - 2.$$

A close inspection of the proof of $\mathrm{C}_k(\mathrm{UP}) \subseteq \mathrm{SD}_{T(k)}(\mathrm{UP})$ according to Theorem 3.2.4 leads to the recurrence:

$$T(1) = 1 \quad \text{and} \quad T(k) = \begin{cases} 2T(k-1) + 3 & \text{if } k > 1 \text{ is odd} \\ 2T(k-1) & \text{if } k > 1 \text{ is even,} \end{cases}$$

since any set $L \in \mathrm{C}_k(\mathrm{UP})$ can be represented by sets $A \in \mathrm{C}_{k-1}(\mathrm{UP})$ and $B \in \mathrm{UP}$ as follows:

$$\begin{array}{lllllll} L & = & A \cup B & = & \overline{\overline{A} \cap \overline{B}} & = & \Sigma^*\Delta\left((\Sigma^*\Delta A) \cap (\Sigma^*\Delta B)\right) \quad \text{if } k \text{ is odd,} \\ L & = & A \cap \overline{B} & = & A \cap (\Sigma^*\Delta B) & & \quad \text{if } k \text{ is even.} \end{array}$$

The above recurrence is in (almost) closed form:

$$T(k) = \begin{cases} 2^k - 1 & \text{if } k \geq 1 \text{ is odd} \\ 2^k - 2 & \text{if } k \geq 1 \text{ is even,} \end{cases}$$

as can be proven by induction on $k$ (we omit the trivial induction base): For odd $k$ (i.e., $k = 2n-1$ for $n \geq 1$), assume $T(2n-1) = 2^{2n-1} - 1$ to be true. Then,

$$T(2n+1) = 2T(2n) + 3 = 4T(2n-1) + 3 \overset{\text{hyp.}}{=} 4\left(2^{2n-1} - 1\right) + 3 = 2^{2n+1} - 1.$$

For even $k$ (i.e., $k = 2n$ for $n \geq 1$), assume $T(2n) = 2^{2n} - 2$ to be true. Then,

$$T(2n+2) = 2T(2n+1) = 2(2T(2n) + 3) \overset{\text{hyp.}}{=} 4\left(2^{2n} - 2\right) + 6 = 2^{2n+2} - 2. \quad \square$$

**Remark 3.2.8** The upper bound in the second part of the above proof can be slightly improved using the fact that $\Sigma^*\Delta\Sigma^*\Delta A = \emptyset\Delta A = A$ for any set $A$. This gives the recurrence:

$$T(1) = 1 \quad \text{and} \quad T(k) = \begin{cases} 2T(k-1) + 1 & \text{if } k > 1 \text{ is odd} \\ 2T(k-1) & \text{if } k > 1 \text{ is even,} \end{cases}$$

or, equivalently, $T(1) = 1$, $T(2) = 2$, and $T(k) = 2^{k-1} + T(k-2)$ for $k \geq 3$. Though this shows that the upper bound given in the above proof is not optimal, the new bound is not a strong improvement, as it still embeds $C_k(\mathrm{UP})$ in an exponentially higher level of $\mathrm{SDH}(\mathrm{UP})$. We propose as an interesting task the establishment of *tight* level-wise containments between the two hierarchies $\mathrm{SDH}(\mathrm{UP})$ and $\mathrm{CH}(\mathrm{UP})$ that capture the Boolean closure of UP, at least up to the limits of relativizing techniques. We conjecture that there is some relativized world in which an exponential increase (though less dramatic than the particular exponential increase of Corollary 3.2.7) indeed is necessary.

Theorem 3.2.9 below shows that each level of the nested difference hierarchy is contained in the same level of both the C and the E hierarchy. Surprisingly, it turns out (see Theorem 3.2.13 below) that, relative to a recursive oracle, even the fourth level of $\mathrm{CH}(\mathrm{UP})$ and the third level of $\mathrm{SDH}(\mathrm{UP})$ are not subsumed by any level of the $\mathrm{EH}(\mathrm{UP})$ hierarchy. Consequently, neither the D nor the E normal forms of Definition 3.2.1 capture the Boolean closure of UP.

**Theorem 3.2.9** For every $k \geq 1$, $D_k(\mathrm{UP}) \subseteq C_k(\mathrm{UP}) \cap E_k(\mathrm{UP})$.

**Proof.** For the first inclusion, by [CH85, Proposition 2.1.2], each set $L \in D_k(\mathrm{UP})$ can be represented as

$$L = A_1 - (A_2 - (\cdots (A_{k-1} - A_k) \cdots)),$$

where $A_i = \bigcap_{1 \leq j \leq i} L_j$, $1 \leq i \leq k$, and the $L_j$'s are the original UP sets representing L. Note that since the proof of [CH85, Proposition 2.1.2] only uses intersection, the sets $A_i$ are in UP. A special case of [CH85, Proposition 2.1.3] says that sets in $D_k(\mathrm{UP})$ via decreasing chains such as the $A_i$ are in $C_k(\mathrm{UP})$, and so $L \in C_k(\mathrm{UP})$.

The proof of the second inclusion is done by induction on the odd and even levels separately. The induction base follows by definition in either case. For odd levels, assume $D_{2n-1}(\mathrm{UP}) \subseteq E_{2n-1}(\mathrm{UP})$ to be valid, and let $L$ be any set in $D_{2n+1}(\mathrm{UP})$, i.e., $L \in \mathrm{UP} - (\mathrm{UP} - D_{2n-1}(\mathrm{UP}))$. By our inductive hypothesis, L can be represented as

$$L = A - \left(B - \left(\bigcup_{i=1}^{n-1} (C_i \cap \overline{D_i}) \cup E\right)\right),$$

where $A, B, C_i, D_i$, and $E$ are sets in UP. Thus,

$$\begin{aligned} L &= A \cap \left(\overline{B \cap \left(\overline{\bigcup_{i=1}^{n-1} (C_i \cap \overline{D_i}) \cup E}\right)}\right) \\ &= A \cap \left(\overline{B} \cup \left(\bigcup_{i=1}^{n-1} (C_i \cap \overline{D_i}) \cup E\right)\right) \\ &= (A \cap \overline{B}) \cup \left(\bigcup_{i=1}^{n-1} A \cap C_i \cap \overline{D_i}\right) \cup (A \cap E) \\ &= \left(\bigcup_{i=1}^{n} F_i \cap \overline{D_i}\right) \cup G, \end{aligned}$$

where $F_i = A \cap C_i$, for $1 \leq i \leq n-1$, $F_n = A$, $D_n = B$, and $G = A \cap E$. Since UP is closed under intersection, each of these sets is in UP. Thus, $L \in E_{2n+1}(\mathrm{UP})$. The proof for the even levels is analogous except that the set E is dropped. □

Note that most of the above proofs used only the facts that the class is closed under intersection and contains $\Sigma^*$ and $\emptyset$:

**Theorem 3.2.10** Theorems 3.2.4, 3.2.5, and 3.2.9 and Corollaries 3.2.6 and 3.2.7 apply to all classes that contain $\Sigma^*$ and $\emptyset$ and are closed under intersection.

**Remark 3.2.11** Although DP is closed under intersection but seems to lack closure under union (unless the polynomial hierarchy collapses to DP [Kad88, CK90b, Cha91]) and thus Theorem 3.2.10 in particular applies to DP, we note that the known results about the Boolean hierarchy over NP [CGH⁺88, KSW87] in fact even for the DP case imply stronger results than those given by our Theorem 3.2.10, due to the very special structure of DP. Indeed, since, e.g., $\mathrm{E}_k(\mathrm{DP}) = \mathrm{E}_{2k}(\mathrm{NP})$ for any $k \geq 1$ (and the same holds for the other hierarchies), it follows immediately that all the level-wise equivalences among the Boolean hierarchies (and also their ability to capture the Boolean closure) that are known to hold for NP also hold for DP even in the absence of the assumption of closure under union. This appears to contrast with the UP case (see Remark 3.2.8).

The following combinatorial lemma will be useful in proving Theorem 3.2.13.

**Lemma 3.2.12** [CHV93] Let $G = (S, T, E)$ be any directed bipartite graph with out-degree bounded by $d$ for all vertices. Let $S' \subseteq S$ and $T' \subseteq T$ be subsets such that $S' \supseteq \{s \in S \mid (\exists t \in T)\,[(s,t) \in E]\}$, and $T' \supseteq \{t \in T \mid (\exists s \in S)\,[(t,s) \in E]\}$. Then either:

1. $\|S'\| \leq 2d$, or
2. $\|T'\| \leq 2d$, or
3. $(\exists s \in S')\,(\exists t \in T')\,[(s,t) \notin E \land (t,s) \notin E]$.

For papers concerned with oracles separating internal levels of Boolean hierarchies over classes other than those of this paper, we refer the reader to ([CGH⁺88, Cai87, GNW90, BJY90, Cro94], see also [GW87]). Theorem 3.2.13 is optimal, as clearly $\mathrm{C}_3(\mathrm{UP}) \subseteq \mathrm{EH}(\mathrm{UP})$ and $\mathrm{SD}_2(\mathrm{UP}) \subseteq \mathrm{EH}(\mathrm{UP})$, and both these containments relativize.

**Theorem 3.2.13** There are recursive oracles $A$ and $D$ (though we may take $A = D$) such that

1. $\mathrm{C}_4(\mathrm{UP}^A) \not\subseteq \mathrm{EH}(\mathrm{UP}^A)$, and
2. $\mathrm{SD}_3(\mathrm{UP}^D) \not\subseteq \mathrm{EH}(\mathrm{UP}^D)$.

**Corollary 3.2.14** There is a recursive oracle $A$ such that

1. $\mathrm{EH}(\mathrm{UP}^A) \neq \mathrm{BC}(\mathrm{UP}^A)$ and $\mathrm{DH}(\mathrm{UP}^A) \neq \mathrm{BC}(\mathrm{UP}^A)$,[4] and

[4] As Fact 3.2.2 shows that $\mathrm{DH}(\mathrm{UP}) = \mathrm{CH}(\mathrm{coUP})$, this oracle $A$ also separates the Boolean (alternating sums) hierarchy over coUP from the fourth level of the same hierarchy over UP and, thus, from $\mathrm{BC}(\mathrm{UP})$.

2. $\mathrm{EH}(\mathrm{UP}^A)$ and $\mathrm{DH}(\mathrm{UP}^A)$ are not closed under all Boolean operations.

**Proof of Theorem 3.2.13.** Although the theorem claims there is an oracle keeping $\mathrm{C}_4(\mathrm{UP})$ from being contained in any level of $\mathrm{EH}(\mathrm{UP})$, we will only prove that for any fixed $k$ we can ensure that $\mathrm{C}_4(\mathrm{UP})$ is not contained in $\mathrm{E}_k(\mathrm{UP})$, relative to some oracle $A^{(k)}$. In the standard way, by interleaving diagonalizations, the sequence of oracles, $A^{(k)}$, can be combined into a single oracle, $A$, that fulfills the claim of the theorem. An analogous comment holds for the second claim of the theorem, with a sequence of oracles $D^{(k)}$ yielding a single oracle $D$. Similarly, both statements of the theorem can be satisfied simultaneously via just one oracle, via interleaving with each other the constructions of $A$ and $D$. Though below we construct just $A^{(k)}$ and $D^{(k)}$ for some fixed $k$, as a notational shorthand we'll use $A$ and $D$ below to represent $A^{(k)}$ and $D^{(k)}$.

Before the actual construction of the oracles, we state some preliminaries that apply to the proofs of both statements in the theorem.

For any $n \geq 0$ and any string $v \in \Sigma^{\leq n}$, define $S_v^n \stackrel{\mathrm{df}}{=} \{vw \mid vw \in \Sigma^n\}$. The sets $S_v^n$ are used to distinguish between different segments of $\Sigma^n$ in the definition of the test languages, $L_A$ and $L_D$.

Fix any standard enumeration of all NPOMs. Fix any $k > 0$. We need only consider even levels of $\mathrm{EH}(\mathrm{UP})$, as each odd level is contained in some even level. Call any collection of $2k$ NPOMs, $H = \langle N_{1,1}, \ldots, N_{k,1}, N_{1,2}, \ldots, N_{k,2} \rangle$, a potential (relativized) $\mathrm{E}_{2k}(\mathrm{UP})$ machine, and for any oracle $X$, define its language to be:

$$L(H^X) \stackrel{\mathrm{df}}{=} \bigcup_{i=1}^{k} \left( L(N_{i,1}^X) - L(N_{i,2}^X) \right).$$

If for some fixed oracle $Y$, a potential (relativized) $\mathrm{E}_{2k}(\mathrm{UP})$ machine $H^Y$ has the property that each of its underlying NPOMs with oracle $Y$ is unambiguous, then $L(H^Y)$ indeed is in $\mathrm{E}_{2k}(\mathrm{UP}^Y)$. Clearly, our enumeration of all NPOMs induces an enumeration of all potential $\mathrm{E}_{2k}(\mathrm{UP})$ oracle machines. For $j \geq 1$, let $H_j$ be the $j$th machine in this enumeration. Let $p_j$ be a polynomial bounding the length of the computation paths of each of $H_j$'s underlying machines (and thus bounding the number of and length of the strings they each query). As a notational convenience, we henceforward will use $H$ and $p$ as shorthands for $H_j$ and $p_j$, and we will denote the underlying NPOMs by $N_{1,1}, \ldots, N_{k,1}, N_{1,2}, \ldots, N_{k,2}$.

The oracle $X$, where $X$ stands for $A$ or $D$, is constructed in stages, $X = \bigcup_{j \geq 1} X_j$. In stage $j$, we diagonalize against $H$ by satisfying the following requirement $R_j$ for every $j \geq 1$:

$R_j$ : Either there is an $n > 2$ and an $i$, $1 \leq i \leq k$, such that one of $N_{i,1}^{X_j}$ or $N_{i,2}^{X_j}$ on input $0^n$ is ambiguous (thus, $H$ is in fact not an $E_{2k}(\mathrm{UP})$ machine relative to $X$), or $L(H^X) \neq L_X$.

Let $X_j$ be the set of strings contained in $X$ by the end of stage $j$, and let $X_j'$ be the set of strings forbidden membership in $X$ during stage $j$. The restraint function $r(j)$ will satisfy the condition that at no later stage will strings of length smaller than $r(j)$ be added to $X$. Also, our construction will ensure that $r(j)$ is so large that $X_{j-1}$ contains no strings of length greater than $r(j)$. Initially, both $X_0$ and $X_0'$ are empty, and $r(1)$ is set to be 2.

We now start the proof of Part 1 of the theorem. Define the test language:

$$L_A \stackrel{\text{df}}{=} \{0^n \mid (\exists x)\,[x \in S_0^n \cap A] \,\wedge\, (\forall y)\,[y \notin S_{10}^n \cap A] \,\wedge\, (\forall z)\,[z \notin S_{11}^n \cap A]\}.$$

Clearly, $L_A$ is in $\mathrm{NP}^A \wedge \mathrm{coNP}^A \wedge \mathrm{coNP}^A$. However, if we ensure in the construction that the invariant $\|S_v^n \cap A\| \leq 1$ is maintained for $v \in \{0, 10, 11\}$ and for every $n \geq 2$, then $L_A$ is even in $\mathrm{UP}^A \wedge \mathrm{coUP}^A \wedge \mathrm{coUP}^A$, and thus in $C_4(\mathrm{UP}^A)$.

We now describe stage $j > 0$ of the oracle construction.

**Stage $j$:** Choose $n > r(j)$ so large that $2^{n-2} > 3p(n)$.

**Case 1:** $0^n \in L(H^{A_{j-1}})$. Since $0^n \notin L_A$, we have $L(H^A) \neq L_A$.

**Case 2:** $0^n \notin L(H^{A_{j-1}})$. Choose some $x \in S_0^n$ and set $B_j := A_{j-1} \cup \{x\}$.

**Case 2.1:** $0^n \notin L(H^{B_j})$. Letting $A_j := B_j$ implies $0^n \in L_A$, so $L(H^A) \neq L_A$.

**Case 2.2:** $0^n \in L(H^{B_j})$. Then there is an $i$, $1 \leq i \leq k$, such that $0^n \in L(N_{i,1}^{B_j})$ and $0^n \notin L(N_{i,2}^{B_j})$. "Freeze" an accepting path of $N_{i,1}^{B_j}(0^n)$ into $A_j'$; that is, add those strings queried negatively on that path to $A_j'$, thus forbidding them from $A$ for all later stages. Clearly, at most $p(n)$ strings are "frozen."

**Case 2.2.1:** $\left(\exists z \in (S_{10}^n \cup S_{11}^n) - A_j'\right)\ \left[0^n \notin L(N_{i,2}^{B_j \cup \{z\}})\right]$.

Choose any such $z$. Set $A_j := B_j \cup \{z\}$. We have $0^n \in L(H^A) - L_A$.

**Case 2.2.2:** $\left(\forall z \in (S_{10}^n \cup S_{11}^n) - A_j'\right)\ \left[0^n \in L(N_{i,2}^{B_j \cup \{z\}})\right]$.

To apply Lemma 3.2.12, define a directed bipartite graph $G = (S, T, E)$ by $S \stackrel{\text{df}}{=} S_{10}^n - A_j'$, $T \stackrel{\text{df}}{=} S_{11}^n - A_j'$, and for each $s \in S$ and $t \in T$, $(s,t) \in E$ if and only if $N_{i,2}^{B_j \cup \{s\}}$ queries $t$ along its lexicographically first accepting path, and $(t,s) \in E$ is defined analogously. The out-degree of all vertices of $G$ is bounded by $p(n)$. By our choice of $n$,

> $\min\{\|S\|, \|T\|\} \geq 2^{n-2} - p(n) > 2p(n)$, and thus alternative 3 of Lemma 3.2.12 applies. Hence, there exist strings $s \in S$ and $t \in T$ such that $N_{i,2}^{B_j \cup \{s\}}(0^n)$ accepts on some path $p_s$ on which $t$ is not queried, and $N_{i,2}^{B_j \cup \{t\}}(0^n)$ accepts on some path $p_t$ on which $s$ is not queried. Since $p_s$ ($p_t$) changes from reject to accept exactly by adding $s$ ($t$) to the oracle, $s$ ($t$) must have been queried on $p_s$ ($p_t$). We conclude that $p_s \neq p_t$, and thus $N_{i,2}^{B_j \cup \{s,t\}}(0^n)$ has at least two accepting paths. Set $A_j := B_j \cup \{s, t\}$.

In each case, requirement $R_j$ is fulfilled. Let $r(j+1)$ be $\max\{n, w_j\}$, where $w_j$ is the length of the largest string queried through stage $j$.

**End of stage j.**

We now turn to the proof of Part 2 of the theorem. The test language here, $L_D$, is defined by:

$$L_D \stackrel{\text{df}}{=} \left\{ 0^n \;\middle|\; \begin{array}{l} ((\exists x)\,[x \in S_0^n \cap D] \ \wedge\ (\exists y)\,[y \in S_{10}^n \cap D] \ \wedge\ (\exists z)\,[z \in S_{11}^n \cap D]) \vee \\ ((\forall x)\,[x \notin S_0^n \cap D] \ \wedge\ (\forall y)\,[y \notin S_{10}^n \cap D] \ \wedge\ (\exists z)\,[z \in S_{11}^n \cap D]) \vee \\ ((\exists x)\,[x \in S_0^n \cap D] \ \wedge\ (\forall y)\,[y \notin S_{10}^n \cap D] \ \wedge\ (\forall z)\,[z \notin S_{11}^n \cap D]) \vee \\ ((\forall x)\,[x \notin S_0^n \cap D] \ \wedge\ (\exists y)\,[y \in S_{10}^n \cap D] \ \wedge\ (\forall z)\,[z \notin S_{11}^n \cap D]) \end{array} \right\}.$$

Again, provided that the invariant $\|S_v^n \cap D\| \leq 1$ is maintained for $v \in \{0, 10, 11\}$ and every $n \geq 2$ throughout the construction, $L_D$ is clearly in $\mathrm{SD}_3(\mathrm{UP}^D)$, as for all sets $A$, $B$, and $C$,

$$A \Delta B \Delta C = (A \cap B \cap C) \cup (\overline{A} \cap \overline{B} \cap C) \cup (A \cap \overline{B} \cap \overline{C}) \cup (\overline{A} \cap B \cap \overline{C}).$$

Stage $j > 0$ of the construction of $D$ is as follows.

**Stage j:** Choose $n > r(j)$ so large that $2^{n-2} > 3p(n)$.

**Case 1:** $0^n \in L(H^{D_{j-1}})$. Since $0^n \notin L_D$, we have $L(H^D) \neq L_D$.

**Case 2:** $0^n \notin L(H^{D_{j-1}})$. Choose some $x \in S_0^n$ and set $E_j := D_{j-1} \cup \{x\}$.

**Case 2.1:** $0^n \notin L(H^{E_j})$. Letting $D_j := E_j$ implies $0^n \in L_D$, so $L(H^D) \neq L_D$.

**Case 2.2:** $0^n \in L(H^{E_j})$. Then, there is an $i$, $1 \leq i \leq k$, such that $0^n \in L(N_{i,1}^{E_j})$ and $0^n \notin L(N_{i,2}^{E_j})$. "Freeze" an accepting path of $N_{i,1}^{E_j}(0^n)$ into $D'_j$. Again, at most $p(n)$ strings are "frozen."

**Case 2.2.1:** $\left(\exists w \in (S^n_{10} \cup S^n_{11}) - D'_j\right) \left[0^n \notin L(N^{E_j \cup \{w\}}_{i,2})\right]$.

Choose any such $w$ and set $D_j := E_j \cup \{w\}$. We have $0^n \in L(H^D) - L_D$.

**Case 2.2.2:** $\left(\forall w \in (S^n_{10} \cup S^n_{11}) - D'_j\right) \left[0^n \in L(N^{E_j \cup \{w\}}_{i,2})\right]$.

As before, Lemma 3.2.12 yields two strings $s \in S^n_{10} - D'_j$ and $t \in S^n_{11} - D'_j$ such that $N^{E_j \cup \{s,t\}}_{i,2}(0^n)$ is ambiguous. Set $D_j := E_j \cup \{s, t\}$.

Again, $R_j$ is always fulfilled. Define $r(j+1)$ as before.

**End of stage j.** □

Finally, we note that a slight modification of the above proof establishes the analogous result (of Theorem 3.2.13) for the case of US [BG82] (which is denoted 1NP in [GW87, Cro94]).

## 3.3 Sparse Turing-complete and Turing-hard Sets for UP

In this section, we show some consequences of the existence of sparse Turing-complete and Turing-hard sets for UP. This question has been carefully investigated for the class NP [KL80, Hop81, KS85, BBS86a, LS86, Sch86, Kad89].[5] Kadin showed that if there is a sparse $\leq^p_T$-complete set in NP, then the polynomial hierarchy collapses to $P^{NP[\log]}$ [Kad89]. Due to the promise nature of UP (in particular, UP probably lacks complete sets [HH88]), Kadin's proof does not seem to apply here. But does the existence of a sparse Turing-complete set in UP cause at least some collapse of the unambiguous polynomial hierarchy (which was introduced recently in [NR93])?[6]

Cai, Hemachandra, and Vyskoč [CHV93] observe that ordinary Turing access to UP, as formalized by $P^{UP}$, may be too restrictive a notion to capture adequately one's intuition of Turing access to unambiguous computation, since in that model the oracle machine has to be unambiguous on *every* input—even those the base DPOM never asks (on any of *its* inputs). To relax that unnaturally strong uniformity requirement they introduce the class denoted $P^{\mathcal{UP}}$, in which NP oracles are accessed in a *guardedly* unambiguous manner, a natural notion of access to unambiguous computation—suggested in the rather analogous case

[5]For reductions less flexible than Turing reductions (e.g., $\leq^p_m$, $\leq^p_{btt}$, etc.), this issue has been studied even more intensely (see, e.g., the surveys [You92, HOW92]).

[6]Note that it is not known whether such a collapse implies a collapse of PH. Note also that Toda's [Tod91] result on whether P-selective sets can be truth-table-hard for UP does not imply such a collapse, as truth-table reductions are less flexible than Turing reductions.

of NP $\cap$ coNP by Grollmann and Selman [GS88]—in which *only computations actually executed need be unambiguous*. Lange, Niedermeier, and Rossmanith [LR94][NR93, p. 483] generalize this approach to build up an entire hierarchy of unambiguous computations in which the oracle levels are guardedly accessed (Definition 3.3.1, Part 3)—the *promise unambiguous polynomial hierarchy*. Since the unambiguous polynomial hierarchy and the promise unambiguous polynomial hierarchy are analogs of the polynomial hierarchy, we recall from Chapter 2 the definition of the polynomial hierarchy in Definition 3.3.1 below.

**Definition 3.3.1**

1. The *polynomial hierarchy* [MS72, Sto77] is defined as follows: $\Sigma_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\Delta_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\Sigma_k^p \stackrel{\text{df}}{=} \mathrm{NP}^{\Sigma_{k-1}^p}$, $\Pi_k^p \stackrel{\text{df}}{=} \mathrm{co}\Sigma_k^p$, $\Delta_k^p \stackrel{\text{df}}{=} \mathrm{P}^{\Sigma_{k-1}^p}$, $k \geq 1$, and $\mathrm{PH} \stackrel{\text{df}}{=} \bigcup_{k\geq 0} \Sigma_k^p$.

2. The *unambiguous polynomial hierarchy* [NR93] is defined as follows: $\mathrm{U}\Sigma_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\mathrm{U}\Delta_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\mathrm{U}\Sigma_k^p \stackrel{\text{df}}{=} \mathrm{UP}^{\mathrm{U}\Sigma_{k-1}^p}$, $\mathrm{U}\Pi_k^p \stackrel{\text{df}}{=} \mathrm{coU}\Sigma_k^p$, $\mathrm{U}\Delta_k^p \stackrel{\text{df}}{=} \mathrm{P}^{\mathrm{U}\Sigma_{k-1}^p}$, $k \geq 1$, and $\mathrm{UPH} \stackrel{\text{df}}{=} \bigcup_{k\geq 0} \mathrm{U}\Sigma_k^p$.

3. The *promise unambiguous polynomial hierarchy* ([LR94][NR93, p. 483]) is defined as follows: $\mathcal{U}\Sigma_0^p \stackrel{\text{df}}{=} \mathrm{P}$, $\mathcal{U}\Sigma_1^p \stackrel{\text{df}}{=} \mathrm{UP}$, and for $k \geq 2$, $L \in \mathcal{U}\Sigma_k^p$ if and only if $L \in \Sigma_k^p$ via NPOMs $N_1, \ldots, N_k$ satisfying for all inputs $x$ and every $i$, $1 \leq i \leq k-1$, that if $N_i$ asks some query $q$ during the computation of $N_1(x)$, then $N_{i+1}(q)$ with oracle $L(N_{i+2}^{L(N_{i+3}^{\cdot^{\cdot^{L(N_k)}}})})$ has at most one accepting path. $\mathcal{UPH} \stackrel{\text{df}}{=} \bigcup_{k\geq 0} \mathcal{U}\Sigma_k^p$. The classes $\mathcal{U}\Delta_k^p$ and $\mathcal{U}\Pi_k^p$, $k \geq 0$, are defined analogously. As a notational shorthand, we often use $\mathrm{P}^{\mathcal{UP}}$ to represent $\mathcal{U}\Delta_2^p$; we stress that both notations are used here to represent the class of sets accepted via *guardedly unambiguous* access to an NP oracle (that is, the class of sets accepted by some P machine with an NP machine's language as its oracle such that on no input does the P machine ask its oracle machine any question on which the oracle machine has more than one accepting path).

4. For each of the above hierarchies, we use $\Sigma_k^{p,A}$ (respectively, $\mathrm{U}\Sigma_k^{p,A}$ and $\mathcal{U}\Sigma_k^{p,A}$) to denote that the $\Sigma_k^p$ (respectively, $\mathrm{U}\Sigma_k^p$ and $\mathcal{U}\Sigma_k^p$) computation is performed relative to oracle $A$; similar notation is used for the $\Pi$ and $\Delta$ classes of the hierarchies.

The following facts follow from the definition (see also [NR93]) or can easily be shown.

**Fact 3.3.2** For every $k \geq 1$,

1. $U\Sigma_k^p \subseteq \mathcal{U}\Sigma_k^p \subseteq \Sigma_k^p$ and $U\Delta_k^p \subseteq \mathcal{U}\Delta_k^p \subseteq \Delta_k^p$.
2. If $U\Sigma_k^p = U\Pi_k^p$, then $UPH = U\Sigma_k^p$.
3. If $U\Sigma_k^p = U\Sigma_{k-1}^p$, then $UPH = U\Sigma_{k-1}^p$.
4. $U\Sigma_k^{p,UP\cap coUP} = U\Sigma_k^p$ and $P^{U\Sigma_k^p \cap U\Pi_k^p} = U\Sigma_k^p \cap U\Pi_k^p$.

"$UP_{\leq k}$," the analogs of UP in which up to $k$ accepting paths are allowed, has been studied in various contexts [Wat88, Hem87, Bei89, CHV93, HH94, HZ93]. One motivation for $U\Sigma_k^p$ is that, for each $k$, $UP_{\leq k} \subseteq U\Sigma_k^p$ [NR93].

Although we are not able to settle affirmatively the question posed at the end of the first paragraph of this section, we do prove in the theorem below that if there is a sparse Turing-complete set for UP, then the levels of the unambiguous polynomial hierarchy are simpler than one would otherwise expect: they "slip down" slightly in terms of their location within the promise unambiguous polynomial hierarchy, i.e., for each $k \geq 3$, the $k$th level of UPH is contained in the $(k-1)$st level of $\mathcal{UPH}$.

**Theorem 3.3.3** If there exists a sparse Turing-complete set for UP, then

1. $UP^{UP} \subseteq P^{\mathcal{UP}}$, and
2. $U\Sigma_k^p \subseteq \mathcal{U}\Sigma_{k-1}^p$ for every $k \geq 3$.

**Proof.** For the first statement, let $L$ be any set in $UP^{UP}$. By assumption, $L \in UP^{P^S} = UP^S$ for some sparse set $S \in UP$. Let $q$ be a polynomial bounding the density of $S$, that is, $\|S^{\leq m}\| \leq q(m)$ for every $m \geq 0$, and let $N_S$ be a UPM for $S$. Let $N_L$ be a UPOM witnessing that $L \in UP^S$, that is, $L = L(N_L^S)$. Let $p(n)$ be a polynomial bounding the length of all query strings that can be asked during the computation of $N_L$ on inputs of length $n$. Define the polynomial $r(n) \stackrel{\text{df}}{=} q(p(n))$ that bounds the number of strings in $S$ that can be queried in the run of $N_L$ on inputs of length $n$.

To show that $L \in P^{\mathcal{UP}}$, we shall construct a DPOM $M$ that may access its $\mathcal{UP}$ oracle $D$ in a guarded manner (more formally, "may access its NP oracle $D$ in a guardedly unambiguous manner," but we will henceforward use $\mathcal{UP}$ and other $\mathcal{U}\cdots$ notations in this informal manner). Before formally describing machine $M$ (Figure 3.1), we give some informal explanations. $M$ will proceed in three basic steps: First, $M$ determines the exact census of that part of $S$ that is relevant for the given input length, $\|S^{\leq p(n)}\|$. Knowing the exact census, $M$ can construct (by prefix search) a table $T$ of all strings in $S^{\leq p(n)}$ without asking queries that make its oracle's machine ambiguous, so the $\mathrm{P}^{\mathcal{UP}}$-like behavior is guaranteed. Finally, $M$ asks its oracle $D$ to simulate the computation of $N_L$ on input $x$ (answering $N_L$'s oracle queries by table-lookup using table $T$), and accepts accordingly.

In the formal description of machine $M$ (given in Figure 3.1), three oracle sets $A$, $B$, and $C$ are used. Since $M$ has only one $\mathcal{UP}$ oracle, the actual set to be used is $D = A \oplus B \oplus C$ (with suitably modified queries to $D$). $A$, $B$, and $C$ are defined as follows (we assume the set $T$ below is coded in some standard reasonable way):

$$A \stackrel{\mathrm{df}}{=} \left\{ \langle 1^n, k \rangle \,\middle|\, \begin{array}{l} n \geq 0 \wedge 0 \leq k \leq r(n) \wedge (\exists c_1 <_{\mathrm{lex}} c_2 <_{\mathrm{lex}} \cdots <_{\mathrm{lex}} c_k) \\ (\forall \ell : 1 \leq \ell \leq k)\, [|c_\ell| \leq p(n) \wedge N_S(c_\ell) \text{ accepts}] \end{array} \right\},$$

$$B \stackrel{\mathrm{df}}{=} \left\{ \langle 1^n, i, j, k, b \rangle \,\middle|\, \begin{array}{l} n \geq 0 \wedge 1 \leq j \leq k \wedge 0 \leq k \leq r(n) \wedge \\ (\exists c_1 <_{\mathrm{lex}} c_2 <_{\mathrm{lex}} \cdots <_{\mathrm{lex}} c_k)\, (\forall \ell : 1 \leq \ell \leq k) \\ [|c_\ell| \leq p(n) \wedge N_S(c_\ell) \text{ accepts} \wedge \text{the } i^{\mathrm{th}} \text{ bit of } c_j \text{ is } b] \end{array} \right\},$$

$$C \stackrel{\mathrm{df}}{=} \{ \langle x, T \rangle \mid \|T\| \leq r(|x|) \wedge N_L^T(x) \text{ accepts} \}.$$

It is easy to see that $M$ runs deterministically in polynomial time. This proves that $L \in \mathrm{P}^{\mathcal{UP}}$.

In order to prove the second statement, let $L$ be a set in $\mathrm{U}\Sigma_k^p$ for any fixed $k \geq 3$. By assumption, there exists a sparse set $S$ in UP such that $L \in \mathrm{U}\Sigma_{k-1}^{p,\mathrm{P}^S} = \mathrm{U}\Sigma_{k-1}^{p,S}$; let $N_1, \ldots, N_{k-1}$ be the UPOMs that witness this fact, that is, $L = L(N_1^{L(N_2^{\cdot^{\cdot^{L(N_{k-1}^S)}}})})$.

Now we describe the computation of a $\mathcal{U}\Sigma_{k-1}^p$ machine $N$ recognizing $L$. As before, $N$ on input $x$ computes in $\mathrm{P}^{\mathcal{UP}}$ its table of advice strings, $T = S^{\leq p(|x|)}$, and then simulates the $\mathrm{U}\Sigma_{k-1}^{p,S}$ computation of $N_1^{L(N_2^{\cdot^{\cdot^{L(N_{k-1}^S)}}})}(x)$ except with $N_1, N_2, \ldots, N_{k-1}$ modified as follows. If in the simulation some machine $N_i$, $1 \leq i \leq k-2$, consults its original oracle $L(N_{i+1}^{(\cdot)})$ about some string, say $z$, then the modified machine $N_i'$ queries the modified machine at the next level, $N_{i+1}'$, about the string $\langle z, T \rangle$ instead. Finally, the advice table $T$, which has been "passed up" in this manner, is used to correctly answer all queries of $N_{k-1}$.

Note that $N$'s oracle in this simulation, $L(N_2'^{L(N_3'^{\cdot^{\cdot^{L(N_{k-1}')}}})})$, is not in general a $\mathrm{U}\Sigma_{k-2}^p$ set (and $L$ is thus not in $\mathrm{U}\Sigma_{k-1}^p$ in general), as the above-described computation depends

**Description of DPOM $M$.**

**input** $x$;
**begin**
  $n := |x|$;
  $k := r(n)$;
  **loop**
    **if** $\langle 1^n, k\rangle \in A$ **then exit loop**
    **else** $k := k - 1$
  **end loop** (* $k$ is now the exact census of $S^{\leq p(n)}$ *)
  $T := \emptyset$; (* $T$ collects the strings of $S^{\leq p(n)}$ *)
  **for** $j = 1$ **to** $k$ **do**
    $c_j := \epsilon$;
    $i := 1$;
    **repeat**
      **if** $\langle 1^n, i, j, k, 0\rangle \in B$ **then** $c_j := c_j 0$; $i := i + 1$
      **else**
        **if** $\langle 1^n, i, j, k, 1\rangle \in B$ **then** $c_j := c_j 1$; $i := i + 1$
        **else** $i := 0$ (* the lex. $j^{\text{th}}$ string of $S^{\leq p(n)}$ has no $i^{\text{th}}$ bit *)
    **until** $i = 0$;
    $T := T \cup \{c_j\}$
  **end for**
  **if** $\langle x, T\rangle \in C$ **then accept**
  **else reject**
**end**

**End of Description of DPOM $M$.**

Figure 3.1: DPOM $M$ guardedly accessing an oracle from $\mathcal{UP}$ to accept a set in $\mathrm{UP}^{\mathrm{UP}}$.

on the advice table $T$, and so, for some bad advice $T$, the unambiguity of the modified UP machines $N'_1, N'_2, \ldots, N'_{k-1}$ is no longer guaranteed. But since our base machine $N$ is able to provide *correct* advice $T$, we have indeed shown that $L \in \mathcal{U}\Sigma^p_{k-1}$. □

In the above proof, the assumption that the sparse set $S$ is in UP is needed to determine the exact census of $S$ (up to a certain length) using the UPM for $S$. Let us now consider the weaker assumption that UP has only a Turing-*hard* sparse set. Karp and Lipton have shown that if there is a sparse Turing-hard set for NP, then the polynomial hierarchy collapses to its second level [KL80].[7] Hopcroft [Hop81] dramatically simplified their proof, and Balcázar, Book, and Schöning [BBS86a, Sch86] generalized, as Theorem 3.3.6, the Karp-Lipton result; the general approach of Hopcroft and Balcázar, Book, and Schöning will be central to our upcoming proof of Theorem 3.3.7.

**Definition 3.3.4** [MP79]

1. A partial order $<_{\mathrm{pwl}}$ on $\Sigma^*$ is *polynomially well-founded and length-related* if and only if (a) every strictly decreasing chain is finite and there is a polynomial $p$ such that every finite $<_{\mathrm{pwl}}$-decreasing chain is shorter than $p$ of the length of its maximum element, and (b) $(\exists q \in \mathbb{P}\mathrm{ol})\ (\forall x, y \in \Sigma^*)\ [x <_{\mathrm{pwl}} y \implies |x| \leq q(|y|)]$.

2. A set $A$ is *self-reducible* if and only if there exist a polynomially well-founded and length-related order $<_{\mathrm{pwl}}$ on $\Sigma^*$ and a DPOM $M$ such that $A = L(M^A)$ and on any input $x \in \Sigma^*$, $M$ queries only strings $y$ with $y <_{\mathrm{pwl}} x$.

**Lemma 3.3.5** [BBS86a] Let $A$ be a self-reducible set and let $M$ witness $A$'s self-reducibility. For any set $B$ and any $n$, if $\left(L(M^B)\right)^{\leq n} = B^{\leq n}$, then $A^{\leq n} = B^{\leq n}$.[8]

Recall the definition of Schöning's low hierarchy [Sch83] from Chapter 2. Of particular interest to us is the class $\mathrm{Low}_2 \stackrel{\mathrm{df}}{=} \{A \mid A \in \mathrm{NP} \text{ and } \mathrm{NP}^{\mathrm{NP}^A} \subseteq \mathrm{NP}^{\mathrm{NP}}\}$. Note that for the special case $k = 0$, Theorem 3.3.7 below says that sets meeting its hypothesis are $\mathrm{Low}_2$.

**Theorem 3.3.6** [BBS86a] If $A$ is a self-reducible set and there is a $k \geq 0$ and a sparse set $S$ such that $A \in \Sigma^{p,S}_k$, then $\Sigma^{p,A}_2 \subseteq \Sigma^p_{k+2}$.

---

[7]Very recently, Köbler and Watanabe [KW95] have improved this collapse to $\mathrm{ZPP}^{\mathrm{NP}}$, and have also obtained new consequences from the assumption that $\mathrm{UP} \subseteq (\mathrm{NP} \cap \mathrm{coNP})/\mathrm{poly}$, whereas we obtain different consequences from the assumption that $\mathrm{UP} \subseteq \mathrm{P}/\mathrm{poly}$.

[8]$A$ can be viewed as a "fixed point" of $M$.

We now state and prove our results regarding sparse Turing-hard sets for UP.

**Theorem 3.3.7** If there exists a sparse Turing-hard set for UP, then

1. $\mathrm{UP} \subseteq \mathrm{Low}_2$, and
2. $\mathrm{U}\Sigma_k^p \subseteq \mathrm{U}\Sigma_j^{p,\Sigma_2^{p,\mathcal{U}\Sigma_{k-j-3}^p}} \cap \mathrm{P}^{\mathcal{U}\Sigma_{k-1}^p \oplus \Sigma_2^p}$ for every $k \geq 3$ and every $j$, with $0 \leq j \leq k-3$.

**Proof.** 1. Let $L \in \Sigma_2^{p,A}$, where $A \in \mathrm{UP}$ via UPM $N_A$ and polynomial-time bound $t$ (we assume that each step is nondeterministic—one can require this, without loss of generality, while maintaining categoricity). Our proof uses the well-known fact that the "left set" [Sel88, OW91] of any UP set is self-reducible and is in UP. More precisely, to apply Theorem 3.3.6 we would need $A$ to be self-reducible. Although that can't be assumed in general of an arbitrary UP set, the left set of $A$, i.e., the set of prefixes of witnesses for elements in $A$ defined by

$$B \stackrel{\mathrm{df}}{=} \{\langle x, y\rangle \mid (\exists z)\,[|yz| = t(|x|) \wedge N_A(x) \text{ accepts on path } yz]\},$$

does have this property and is also in UP. A self-reducing machine $M_{self}$ for $B$ is given in Figure 3.2. Note that the queries asked in the self-reduction are strictly less than the input with respect to a polynomially well-founded and length-related partial order $<_{\mathrm{pwl}}$ defined by: For fixed $x$ and all strings $y_1, y_2 \in \Sigma^{\leq p(|x|)}$, $\langle x, y_1\rangle <_{\mathrm{pwl}} \langle x, y_2\rangle$ if and only if $y_2$ is prefix of $y_1$.

By assumption, since $B$ is a UP set, $B \in \mathrm{P}^S$ for some sparse set $S$, so Theorem 3.3.6 with $k = 0$ applies to $B$. Furthermore, $A$ is in $\mathrm{P}^B$, via prefix search by DPOM $M_A$ (Figure 3.3). Thus, $L \in \Sigma_2^{p,\mathrm{P}^B} \subseteq \Sigma_2^{p,B} \subseteq \Sigma_2^p$, which shows that $A \in \mathrm{Low}_2$.

2. For $k = 3$ (thus $j = 0$), both inclusions have already been shown in Part 1, as $\Sigma_2^p \subseteq \Delta_3^p$. Now fix any $k > 3$, and let $L \in \mathrm{U}\Sigma_k^p = \mathrm{U}\Sigma_{k-1}^{p,A}$ be witnessed by UPOMs $N_1, N_2, \ldots, N_{k-1}$ and $A \in \mathrm{UP}$. Define $B$ to be the left set of $A$ as in Part 1, so $A \in \mathrm{P}^B$ via DPOM $M_A$ (see Figure 3.3), and $B$ is self-reducible via $M_{self}$ (see Figure 3.2), and $B$ is in UP. By hypothesis, $B \in \mathrm{P}^S$ for some sparse set $S$; let $M_B$ be the reducing machine, that is $B = L(M_B^S)$, and let $m$ be a polynomial bound on the runtime of $M_B$. Let $q$ be a polynomial such that $\|S^{\leq m}\| \leq q(m)$ for every $m \geq 0$. Let $p(n)$ be a polynomial bounding the length of all query strings whose membership in the oracle set $B$ can be asked in the run of $N_1$ (with oracle machines $N_2, N_3, \ldots, N_{k-1}, M_A^B$) on inputs of length $n$. Define the polynomials $r(n) \stackrel{\mathrm{df}}{=} m(p(n))$ and $s(n) \stackrel{\mathrm{df}}{=} q(r(n))$.

**Description of Self-reducer $M_{\text{self}}$ for $B$.**

**input** $\langle x, y\rangle$;
**begin**
  **if** $|y| > t(|x|)$ **then reject**;
  **if** $N_A(x)$ accepts on path $y$ **then accept**
  **else**
    **if** $\langle x, y0\rangle \in B$ or $\langle x, y1\rangle \in B$ **then accept**
    **else reject**
**end**

**End of Description of Self-reducer $M_{\text{self}}$ for $B$.**

Figure 3.2: A self-reducing machine for the left set of a UP set.

**Description of DPOM $M_A$.**

**input** $x$;
**begin**
  $y := \epsilon$;
  **while** $|y| < t(|x|)$ **do**
    **if** $\langle x, y0\rangle \in B$ **then accept**
    **else** $y := y1$
  **end while**
  **if** $\langle x, y\rangle \in B$ **then accept**
  **else reject**
**end**

**End of Description of DPOM $M_A$.**

Figure 3.3: A Turing reduction from a UP set $A$ to its left set $B$ via prefix search.

To show that $L \in P^{\mathcal{U}\Sigma^p_{k-1} \oplus \Sigma^p_2}$, we will describe a DPOM $M$ that on input $x$, $|x| = n$, using the $\Sigma^p_2$ part $D$ (defined below) of its oracle, performs a prefix search to extract the lexicographically smallest of all "good" advice sets (this informal term will be formally defined in the next paragraph), say $T$, and then calls the $\mathcal{U}\Sigma^p_{k-1}$ part of its oracle to simulate the $U\Sigma^{p,A}_{k-1}$ computation of $N_1^{L(N_2^{\cdot^{\cdot^{L(N^A_{k-1})}}})}(x)$ except with $N_1, N_2, \ldots, N_{k-1}$ modified in the same way as was described in the proof of Theorem 3.3.3. In more detail, if in the simulation some machine $N_i$, $1 \leq i \leq k-2$, consults its original oracle $L(N^{(\cdot)}_{i+1})$ about some string, say $z$, then the modified machine $N'_i$ queries the modified machine at the next level, $N'_{i+1}$, about the string $\langle z, T\rangle$ instead. Finally, if $N_{k-1}$ consults its original oracle $A$ about some query $y$, then the modified machine $N'_{k-1}$ runs the P computation $M_A^{L(M^T_B)}$ on input $\langle y, T\rangle$ instead to correctly answer this query without consulting an oracle.

An advice set $T$ is said to be *good* if the set $L(M^T_B)$ is a fixed point of $B$'s self-reducer $M_{self}$ up to length $p(n)$, that is, $\left(L(M_{self}^{L(M^T_B)})\right)^{\leq p(n)} = \left(L(M^T_B)\right)^{\leq p(n)}$, and thus $B^{\leq p(n)} = \left(L(M^T_B)\right)^{\leq p(n)}$ by Lemma 3.3.5. This property is checked for each guessed $T$ in the $\Sigma^p_2$ part of the oracle. Formally,

$$D \stackrel{\text{df}}{=} \left\{ \langle 1^n, i, j, b\rangle \;\middle|\; \begin{array}{l} n \geq 0 \wedge (\exists T \subseteq \Sigma^{\leq r(n)})\,(\forall w : |w| \leq p(n)\,)\,[T = \{c_1, \ldots, c_k\} \\ \wedge\, 0 \leq k \leq s(n) \wedge c_1 <_{\text{lex}} \cdots <_{\text{lex}} c_k \wedge \text{the } i^{\text{th}} \text{ bit of } c_j \text{ is } b \\ \wedge\, (w \in L(M^T_B) \iff w \in L(M_{self}^{L(M^T_B)}))] \end{array} \right\}.$$

The prefix search of $M$ is similar to the one performed in the proof of Theorem 3.3.3 (see Figure 3.1); $M$ queries $D$ to construct each string of $T$ bit by bit.

To prove the other inclusion, fix any $j$, $0 \leq j \leq k-3$. We describe a UPOM $N$ witnessing that $L \in U\Sigma_j^{p,\Sigma_2^{p,\mathcal{U}\Sigma^p_{k-j-3}}}$. On input $x$, $N$ simulates the $U\Sigma^p_j$ computation of the first $j$ UPOMs $N_1, \ldots, N_j$. In the subsequent $\Sigma^p_2$ computation, two tasks have to be solved in parallel: the computation of $N_{j+1}$ and $N_{j+2}$ is to be simulated, and good advice sets $T$ have to be determined. For the latter task, the base machine of the $\Sigma^p_2$ computation guesses all possible advice sets and the top machine checks if the guessed advice is good (that is, if $L(M^T_B)$ is a fixed point of $M_{self}$). Again, each good advice set $T$ is "passed up" to the machines at higher levels $N_{j+3}, \ldots, N_{k-1}$ (in the same fashion as was employed earlier in this proof and also in the proof of Theorem 3.3.3), and is used to correctly answer all queries of $N_{k-1}$ without consulting an oracle. This proves the theorem. □

Since Theorem 3.3.7 relativizes and there are relativized worlds in which $UP^A$ is not

$\mathrm{Low}_2^{\mathrm{A}}$ [SL92], we have the following corollary.

**Corollary 3.3.8** There is a relativized world in which (relativized) UP has no sparse Turing-hard sets.

## 3.4 Promise SPP is at Least as Hard as the Polynomial Hierarchy

The promise unambiguous polynomial hierarchy, $\mathcal{UPH}$, is by definition contained in the polynomial hierarchy. Lange and Rossmanith [LR94] have shown that $\mathcal{UPH}$ is also contained in SPP. The somewhat complicated proof given in [LR94] draws upon the characterization of $\mathcal{UPH}$ by "weakly unambiguous circuits of exponential size and bounded depth." Alternatively, the result easily follows from the observation that the proof of the self-lowness of SPP, i.e., $\mathrm{SPP}^{\mathrm{SPP}} = \mathrm{SPP}$ [FFK94], can straightforwardly be modified to even establish $\mathrm{SPP}^{\mathcal{SPP}} = \mathrm{SPP}$, provided that the $\mathcal{SPP}$ oracle is accessed in a *guarded* manner. Consequently, if one defines $\mathcal{SPH}$ to be the "gap analog" of $\mathcal{UPH}$, then $\mathcal{SPH}$ collapses to SPP, and hence, $\mathcal{UPH} \subseteq \mathcal{SPH} = \mathrm{SPP}$.

This section addresses a question that, quite generally speaking, is motivated by the fact that the relation between PH and SPP is unknown.

Toda and Ogiwara have shown that for a large family of counting classes $\mathcal{K}$ such as PP, $\mathrm{C_{=}P}$, and $\oplus\mathrm{P}$ (whose relation to PH also is not known), $\mathcal{K}^{\mathrm{PH}} \subseteq \mathrm{BP} \cdot \mathcal{K}$. Informally speaking, with respect to random reductions, each such counting class $\mathcal{K}$ is at least as hard as the polynomial hierarchy [TO92]. It is natural to ask whether such a result also holds for SPP.[9] Toda and Ogiwara conjectured that this is not the case, i.e., $\mathrm{SPP}^{\mathrm{PH}}$, or even PH, is unlikely to be contained in $\mathrm{BP} \cdot \mathrm{SPP}$ [TO92], due essentially to the promise inherent in the definition of SPP and to the fact that the method of [TO92] relies on there being no such promise for the class $\mathcal{K}$.

Further evidence for PH not being contained in $\mathrm{BP} \cdot \mathrm{SPP}$ is provided by the fact (noted in [FFK94]) that one can easily (i.e., using known results) construct an oracle relative to which Toda and Ogiwara's conjecture is true. Indeed, the following implications all

[9] More generally, Toda and Ogiwara pose the question of whether their technique applies to all the "gap-definable" classes [FFK94]—note that PP, $\mathrm{C_{=}P}$, $\oplus\mathrm{P}$, and SPP all are gap-definable. In particular, SPP is the smallest gap-definable class containing $\emptyset$ and $\Sigma^*$.

relativize (i.e., they hold relative to every oracle):

$$\begin{aligned} \mathrm{NP} \subseteq \mathrm{BP} \cdot \mathrm{SPP} = \mathrm{BPP}^{\mathrm{SPP}} & \implies \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{P}^{\mathrm{BPP}^{\mathrm{SPP}}} \\ & \implies \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{BPP}^{\mathrm{SPP}} \\ & \implies \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{PP}^{\mathrm{SPP}} \\ & \implies \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{PP}. \end{aligned}$$

Since Beigel has constructed an oracle $A$ such that $\mathrm{P}^{\mathrm{NP}^A} \not\subseteq \mathrm{PP}^A$ [Bei92], it follows from the above implications that $\mathrm{NP}^A \not\subseteq (\mathrm{BP} \cdot \mathrm{SPP})^A$ holds relative to the same oracle. Thus, any proof refuting the conjecture of Toda and Ogiwara would not relativize.

Toda and Ogiwara's main result can be stated as follows. Intuitively, it says that the characteristic function of any set in PH can be approximated by a GapP function with high probability.

**Lemma 3.4.1** ([TO92], see also [Gup91])

$$(\forall L \in \mathrm{PH})\,(\exists F \in \mathrm{GapP})\,(\exists r \in \mathbb{P}\mathrm{ol})\,(\forall x \in \Sigma^*)$$

$$\left[\Pr_{r(|x|)}[w \mid (x \in L \iff F(x,w) = 1) \wedge (x \notin L \iff F(x,w) = 0)] \geq \frac{3}{4}\right].$$

**Remark 3.4.2** 1. By applying a technique that is based on transforming Boolean circuits, Tarui [Tar91] provides a stronger version of this lemma that achieves even *one-sided* (rather than two-sided) error. He thus proves that PH is contained in $\mathrm{ZP} \cdot \mathrm{PP}$ and in $\mathrm{RP} \cdot \mathrm{C_{=}P}$, noting that his technique can also be applied to obtain $\mathrm{PP}^{\mathrm{PH}} \subseteq \mathrm{ZP} \cdot \mathrm{PP}$. In [RV92], it is shown that $\mathrm{C_{=}P}^{\mathrm{PH}} \subseteq \mathrm{RP} \cdot \mathrm{C_{=}P}$. This result, however, does not improve on the result of [TO92], since Gupta has shown that $\mathrm{BP} \cdot \mathrm{C_{=}P} = \mathrm{RP} \cdot \mathrm{C_{=}P}$ [Gup93].

2. Both [TO92] and [Tar91] heavily draw upon Valiant and Vazirani's technique of probabilistically restricting the solution space of NP sets so as to provide a random reduction from any NP set to every solution to (1SAT, SAT) [VV86]. As a corollary, $\mathrm{NP} \subseteq \mathrm{RP} \cdot \oplus\mathrm{P}$, and this *latter* result has been generalized in [TO92, Tar91] to all levels of PH and to non-promise counting classes other than $\oplus$P. Our goal here is to provide a generalization to all levels of PH that, formally, is closer tied to Valiant and Vazirani's actual result in terms of solutions to promise problems. It is worth noting that, when generalizing their result to all of PH, the class $\mathcal{SPP}$ is needed rather than

$\mathcal{UP}$ (or, equivalently, $(1\mathrm{SAT}, \mathrm{SAT})$) which suffices in the NP case—the reason is that the *alternation* of $\exists$ and $\forall$ quantifiers requires the use of GapP functions rather than #P functions.

The definition of the BP operator is below extended to apply also to classes of promise problems, following Selman's approach. Selman [Sel88] defines polynomial-time reducibilities between promise problems according to the following definition template: Let $\leq_r^p$ be an arbitrary polynomial-time reducibility. Then, a promise problem $(Q, R)$ is $\leq_r^{\text{Promise Problem}}$-reducible to a promise problem $(S, T)$ if for every solution $A$ of $(S, T)$ there is a solution $B$ of $(Q, R)$ such that $B \leq_r^p A$.

**Remark 3.4.3** It might be tempting to change Selman's definition template so as to define "$(Q, R) \leq_r^{\text{Promise Problem}} (S, T)$" if each solution $B$ of $(Q, R)$ $\leq_r^p$-reduces to *some* solution $A$ of $(S, T)$. However, as pointed out to this author by Hemaspaandra, this approach would be less useful than Selman's, since under this definition even the trivial promise problem $\mathcal{E} \stackrel{\text{df}}{=} (\emptyset, \emptyset)$ has the property that $(\Sigma^*, \mathrm{SAT}) \leq_T^{\text{Promise Problem}} \mathcal{E}$. In fact, one can replace SAT here with some problem complete for any huge complexity class much bigger than NP and the claim holds. In contrast, Selman's definition sets its quantification so as to make the requirements to the "usefulness of a promise problem as a database to solve some given problem" (and this is the general intuition behind any type of Turing reductions between problems) *as demanding as possible*. Therefore, Selman's definition template is the right and natural approach to reducibilities between promise problems.

**Definition 3.4.4** Let $\mathcal{C}$ be any class of promise problems.

$$\mathrm{BP} \cdot \mathcal{C} \stackrel{\text{df}}{=} \left\{ (Q, R) \;\middle|\; \begin{array}{l} (\exists (S, T) \in \mathcal{C})\, (\exists p \in \mathbb{P}\mathrm{ol})\, (\forall A \in \mathrm{solns}(S, T)) \\ (\exists B \in \mathrm{solns}(Q, R))\, (\forall x)\ \left[\Pr_{p(|x|)}[y \mid \chi_B(x) = \chi_A(x, y)] \geq \frac{3}{4}\right] \end{array} \right\}.$$

Lemma 3.4.5 below says that for all classes $\mathcal{K}$ closed under truth-table reductions, the (in general less flexible) "operator-based access" to $\mathcal{K}$ is as powerful as accessing $\mathcal{K}$ via the corresponding oracle machines. That is, using the notations of Part 4 of Remark 2.3.2 on page 12 and instantiating our assertion to the case of the FEW and the SP operator, if $\mathfrak{R}_{tt}^{p}(\mathcal{K}) \subseteq \mathcal{K}$, then $\mathfrak{R}_{m}^{\mathrm{FewP}}(\mathcal{K}) = \mathrm{FewP}^{\mathcal{K}}$ and $\mathfrak{R}_{m}^{\mathrm{SPP}}(\mathcal{K}) = \mathrm{SPP}^{\mathcal{K}}$. We stress that this claim holds true for many more polynomial-time operators than only FEW or SP; in fact, it applies to any polynomial-time operator defined in this thesis. Lemma 3.4.5 will be applied in the upcoming proof of Theorem 3.4.6, and it will also be useful in several places of Chapter 4.

**Lemma 3.4.5** Let $\mathcal{K}$ be any class of sets closed under truth-table reductions. Then,

$$\mathrm{FewP}^{\mathcal{K}} = \mathrm{F{\scriptsize EW}} \cdot \mathcal{K} \quad \text{and} \quad \mathrm{SPP}^{\mathcal{K}} = \mathrm{SP} \cdot \mathcal{K}.$$

**Proof.** We will only prove $\mathrm{FewP}^{\mathcal{K}} = \mathrm{F{\scriptsize EW}} \cdot \mathcal{K}$, as the other equality can be shown analogously. The inclusion $\mathrm{F{\scriptsize EW}} \cdot \mathcal{K} \subseteq \mathrm{FewP}^{\mathcal{K}}$ is obvious, as a FewP oracle machine on input $x$, in order to mimic the acceptance mechanism of $\mathrm{F{\scriptsize EW}} \cdot \mathcal{K}$, simply generates all strings of length $p(|x|)$ for some suitable polynomial $p$, queries "$\langle x, y\rangle \in \mathcal{K}$?" on each path generated, and accepts if and only if the answer is "yes."

Conversely, let $L \in \mathrm{FewP}^{\mathcal{K}}$ via some FewP oracle machine $M$ with oracle $A \in \mathcal{K}$.[10] Define a set $B$ of all strings $\langle x, y\rangle$ such that $y = \langle w, q_1, a_1, \ldots, q_k, a_k\rangle$, where $k \in \mathrm{FP}$ depends on $x$, $w$ is an accepting computation path of $M$ on input $x$ with queries $q_1, \ldots, q_k$, and $a_1, \ldots, a_k$ are the correct answers to these queries. Then, $B$ truth-table reduces to $A$ and is thus in $\mathcal{K}$. Since for each $x$, $|y| = q(|x|)$ for some $q \in \mathbb{P}\mathrm{ol}$ and the number of strings $y$ such that $\langle x, y\rangle \in B$ is polynomially bounded in $|x|$, $B$ witnesses that $L \in \mathrm{F{\scriptsize EW}} \cdot \mathcal{K}$. $\square$

**Theorem 3.4.6** $\mathrm{SPP}^{\mathrm{PH}} \subseteq \mathrm{BP} \cdot \mathcal{SPP}$.

**Corollary 3.4.7** $\mathrm{SPP}^{\mathrm{BPP}} \subseteq \mathrm{BP} \cdot \mathcal{SPP}$.

**Proof of Theorem 3.4.6.** Let $L$ be any set in $\mathrm{SPP}^{\mathrm{PH}}$. By Lemma 3.4.5, $L \in \mathrm{SP} \cdot \mathrm{PH}$. Then, there exists a function $g \in \mathrm{GAP} \cdot \mathrm{PH}$ such that $g(x) = \chi_L(x)$ for each $x$. Since each $\mathrm{GAP} \cdot \mathrm{PH}$ function can be represented as the difference of a $\mathrm{NUM} \cdot \mathrm{PH}$ function and an FP function (this is a straightforward generalization of the corresponding result for GapP [FFK94]), there exist a set $A \in \mathrm{PH}$, an FP function $f$, and a polynomial $p$ such that for each $x \in \Sigma^*$,

$$\|\{y \mid \langle x, y\rangle \in A \wedge |y| = p(|x|)\}\| = \begin{cases} f(x) + 1 & \text{if } x \in L \\ f(x) & \text{if } x \notin L. \end{cases}$$

Fix any $x$ and $y$ with $|y| = p(|x|)$. By Lemma 3.4.1, for $A \in \mathrm{PH}$, there exist a function $F \in \mathrm{GapP}$ and a polynomial $r$ such that

$$\Pr_{r(|x|)}[w \mid Z_{x,y}(w)] \geq \frac{3}{4},$$

[10] As in the case of UPOMs, whether $M$ is a FewP oracle machine depends crucially on its oracle. So, to be definite, $L = L(M^A) \in \mathrm{FewP}^{\mathcal{K}}$.

where the predicate $Z_{x,y}(w)$ is defined on $\Sigma^{r(|x|)}$ by

$$Z_{x,y}(w) \stackrel{\text{df}}{=} (\langle x,y\rangle \in A \Longleftrightarrow F(x,y,w)=1) \wedge (\langle x,y\rangle \notin A \Longleftrightarrow F(x,y,w)=0).$$

For any $x$ and $w$, with $|w| = r(|x|)$, define

$$G(x,w) \stackrel{\text{df}}{=} -f(x) + \sum_{y\,:\,|y|=p(|x|)} F(x,y,w).$$

By the closure properties of GapP [FFK94], we clearly have $G \in \text{GapP}$. Now define the promise problem $(Q,R)$ by

$$\begin{aligned} Q &\stackrel{\text{df}}{=} \{\langle x,w\rangle \mid G(x,w) \in \{0,1\} \wedge |w| = r(|x|)\},\\ R &\stackrel{\text{df}}{=} \{\langle x,w\rangle \mid G(x,w) = 1 \wedge |w| = r(|x|)\}. \end{aligned}$$

Clearly, $(Q,R) \in \mathcal{SPP}$. Let $S$ be any solution to $(Q,R)$. For fixed $x$ and $y$ and for any $w \in \Sigma^{r(|x|)}$ for which $Z_{x,y}(w)$ is true, it holds that $x \in L$ implies $G(x,w) = 1$, and $x \notin L$ implies $G(x,w) = 0$. That is, if $w$ satisfies $Z_{x,y}(w)$, then $\langle x,w\rangle \in Q$, and thus,

$$x \in L \iff \chi_S(x,w) = \chi_R(x,w) = 1.$$

It follows that

$$\Pr_{r(|x|)}[w \mid \chi_L(x) = \chi_S(x,w)] \geq \Pr_{r(|x|)}[w \mid Z_{x,y}(w)] \geq \frac{3}{4}.$$

Hence, $(\Sigma^*, L) \in \text{BP} \cdot \mathcal{SPP}$. This completes the proof. □

We conclude this section with the remark that an easy modification of the above proof establishes a slight generalization of Corollary 3.4.7: For classes $\mathcal{K}$ which are closed under padding, join, and complementation,

$$\text{SP} \cdot \text{BP} \cdot \mathcal{K} \subseteq \text{BP} \cdot \mathcal{SP} \cdot \mathcal{K}. \tag{3.3}$$

In recent years, much attention has been paid to switching operators, for this—besides being interesting in its own right—yields new insights into the structure and power of hierarchies, such as PH, built upon operators. In particular, it is known that $\text{Op}\cdot\text{BP}\cdot\mathcal{K} \subseteq \text{BP}\cdot\text{Op}\cdot\mathcal{K}$ for any operator Op chosen from $\{\exists, \forall, \text{C}, \text{C}_{=}, \oplus\}$ ([TO92, Tod91, RR91], see the survey [Sch91]). However, the "switch" between the BP and the SP operator stated in (3.3) above is the best result that can be proven by current techniques. The question of whether (3.3) can be strengthened to $\text{Op} \cdot \text{BP} \cdot \mathcal{K} \subseteq \text{BP} \cdot \text{Op} \cdot \mathcal{K}$, where Op is either SP or $\mathcal{SP}$, remains open.

# Chapter 4

# Upward Separation for FewP and Related Classes

## 4.1 Introduction

A main task in complexity theory is to prove collapses or separations between complexity classes, or, if this doesn't succeed (as is often the case), to provide structural consequences from some collapse or separation. The techniques of *upward* and *downward separation* deal with the link of small and large classes: downward separation typically shows that the separation of large classes is downwards translated to smaller ones (e.g., if some level of the polynomial hierarchy differs from the succeeding one, then all smaller levels form a strict hierarchy [Sto77, MS72]), whereas upward separation results state that if small (i.e., polynomial-time) classes differ on sets of small density such as sparse or tally sets, then their exponential-time counterparts are separated. The first results of this kind are due to Book who has shown that $\mathrm{E} \neq \mathrm{NE}$ if and only if there exist tally sets in $\mathrm{NP} - \mathrm{P}$ [Boo74] (see Lemma 4.2.2), and to Hartmanis et al. who have shown that $\mathrm{E} \neq \mathrm{NE}$ if and only if there exist sparse sets in $\mathrm{NP} - \mathrm{P}$ [Har83, HIS85]. Any class sharing with NP this property w.r.t. sparse sets is said to possess (or to display) upward separation.

In contrast to the NP case, several results have been established that reveal the limitations of the upward separation technique by showing that certain classes do not robustly (i.e., with respect to all oracles) display upward separation (we will say those classes "defy" upward separation). Hartmanis, Immerman, and Sewelson have shown that the upward separation technique fails for coNP relative to an oracle [HIS85], and Hemaspaandra and Jha provided relativizations in which the promise classes BPP, R, and ZPP defy upward separation [HJ93]. They posed the question of whether one can prove similar failings regarding upward separation for other promise classes, and even the non-promise class PP. Allender constructed an oracle relative to which $\bigcup_{c>0} \mathrm{DTIME}[2^{c2^n}] = \bigcup_{c>0} \mathrm{NTIME}[2^{c2^n}]$ and yet NP $-$ P contains extremely sparse sets [All91] (see also [AW90]). In addition, his paper presents some new—even though restricted—upward separation results regarding the (promise) classes UP and FewP: there exist sets of constant (respectively, logarithmic) density in UP $-$ P (respectively, FewP $-$ P) if and only if the respective exponential-time analogs differ [All91]. The natural question arises whether or not, in FewP$-$P, the existence of log-sparse sets is equivalent to the existence of sparse sets; Allender suspected that this equivalence does not robustly hold [All91]. In this chapter, we refute this conjecture by showing that FewP *does* robustly display upward separation. In fact, this follows from a more general result (Theorem 4.3.6) that provides a simple sufficient condition for a class to possess upward separation:[1] all the class is required to satisfy is closure under the FEW operator (defined in Section 4.3). As a consequence, upward separation results are obtained for a variety of known counting classes, including $\oplus$P, coC$_=$P, SPP, and LWPP. In contrast to the work of Hemaspaandra and Jha [HJ93], who gave the first examples of promise classes that fail to robustly display upward separation, we show that this behavior is not typical for promise classes in general by providing the first examples of promise classes, specifically FewP, SPP, and LWPP, that *do* have upward separation.

Buhrman, E. Hemaspaandra, and Longpré's tally encoding of sparse sets, introduced to prove the surprising result that any sparse set conjunctively truth-table reduces to some tally set [BHL] (see [Sal93] for an alternative proof and [Sch93] for another application of their technique), is central to the proof of our main result. Buhrman, E. Hemaspaandra, and Longpré's coding of a sparse set improves upon the one used by Hartmanis, Immerman, and Sewelson [HIS85] in order to establish (and to apply to NP) the upward separation technique.

[1]Another structural sufficient condition for a different type of upward separation (giving results of the form: "NP $-$ BPP contains sparse sets if and only if NE $\not\subseteq$ BPE") is observed in [HJ93]. Unlike our results, those are in fact established via the technique of Hartmanis et al. [Har83, HIS85].

## 4.2 Preliminaries

The upward separation technique relates certain structural properties of polynomial-time complexity classes to their "exponential-time analogs." Adopting the notation of [HJ93], we can precisely formalize such a coupling of classes in a unifying way.

**Definition 4.2.1** [HJ93]

1. $A \leq_m^e B$ if $A$ exponential-time (i.e., $\bigcup_{c>0}$ DTIME$[2^{cn}]$) many-one reduces to $B$.
2. $A \leq_{m,\,e\ell d}^{p} B$ if $A \leq_m^p B$ via a reduction $f$ that is exponentially length-decreasing (i.e., $(\exists c > 0)\ (\forall x : |x| \geq 2)\ [2^{c|f(x)|} \leq |x|]$).
3. We say that a pair of classes $(\mathcal{A}, \mathcal{B})$ is an *associated pair* if $\mathfrak{R}_m^e(\mathcal{A}) \subseteq \mathcal{B}$ and $\mathfrak{R}_{m,\,e\ell d}^{p}(\mathcal{B}) \subseteq \mathcal{A}$.

Clearly, $(\mathrm{P}, \mathrm{E})$ is an associated pair. Now consider any class $\mathcal{K}$ that is defined via a certain acceptance mode of NPMs (for example, think of any $\mathcal{K}$ chosen from $\{\mathrm{NP}, \mathrm{FewP}, \oplus\mathrm{P}, \mathrm{PP}, \mathrm{C}_=\mathrm{P}, \mathrm{SPP}\}$). Then, the associated exponential-time analog, $\mathcal{L}$, is defined via the same acceptance mechanism in terms of $2^{cn}$-time bounded NTMs—notationally, $\mathcal{L}$ thus differs from $\mathcal{K}$ just in the extension "E" rather than "P" indicating the different time bound. For example, $(\mathrm{NP}, \mathrm{NE})$, $(\mathrm{FewP}, \mathrm{FewE})$,[2] $(\oplus\mathrm{P}, \oplus\mathrm{E})$, $(\mathrm{PP}, \mathrm{PE})$, $(\mathrm{C}_=\mathrm{P}, \mathrm{C}_=\mathrm{E})$, and $(\mathrm{SPP}, \mathrm{SPE})$ all are associated pairs.

Given any set $L \subseteq \Sigma^*$, we can prefix its strings $x$ by a 1 and then interpret as natural numbers $\mathrm{bin}(1x)$ in binary representation (see [Boo74, Har83]), thus converting $L$ to a tally set:

$$\mathrm{tally}(L) \stackrel{\mathrm{df}}{=} \{0^{\mathrm{bin}(1x)} \mid x \in L\}.$$

Conversely, any tally set $T$ can be transformed into a set of strings over $\Sigma$:

$$\mathrm{bin}(T) \stackrel{\mathrm{df}}{=} \{x \mid 0^{\mathrm{bin}(1x)} \in T\}$$

containing the same information as $T$ in "logarithmically compressed" form. Clearly, for any set $L$, $\mathrm{bin}(\mathrm{tally}(L)) = L$. Using the above notations, the key observation Book's results

[2] The promise of a FewP machine to have at most polynomially many accepting paths translates in the FewE case to the promise of having at most $2^{\mathcal{O}(n)}$ accepting paths, which still are few compared with the double-exponential total number of paths an exponential-time NTM can have [AR88].

essentially draw upon [Boo74] can be stated as follows: For any set $L \subseteq \Sigma^*$, $L \leq_m^e \mathrm{tally}(L)$ and $\mathrm{tally}(L) \leq_{m,\,e\ell d}^p L$. For completeness, the straightforward generalization of Book's results about $(\mathrm{NP}, \mathrm{NE})$ to every associated pair containing $(\mathrm{P}, \mathrm{E})$ is presented.

**Lemma 4.2.2** If $\mathrm{P} \subseteq \mathcal{K}$ and $(\mathcal{K}, \mathcal{L})$ is an associated pair, then $\mathcal{K} - \mathrm{P}$ contains tally sets if and only if $\mathcal{L} \neq \mathrm{E}$.

**Proof.** Since $(\mathcal{K}, \mathcal{L})$ and $(\mathrm{P}, \mathrm{E})$ are both associated pairs, we have $\mathfrak{R}_m^e(\mathcal{K}) \subseteq \mathcal{L}$, $\mathfrak{R}_{m,\,e\ell d}^p(\mathcal{L}) \subseteq \mathcal{K}$, $\mathfrak{R}_m^e(\mathrm{P}) \subseteq \mathrm{E}$, and $\mathfrak{R}_{m,\,e\ell d}^p(\mathrm{E}) \subseteq \mathrm{P}$. Assume $\mathcal{L} \neq \mathrm{E}$, and let $L \subseteq \Sigma^*$ be some set in $\mathcal{L} - \mathrm{E}$. Then, $\mathrm{tally}(L) \leq_{m,\,e\ell d}^p L$ implies $\mathrm{tally}(L) \in \mathcal{K}$. Suppose $\mathrm{tally}(L) \in \mathrm{P}$. Then, $L \leq_m^e \mathrm{tally}(L)$ implies $L \in \mathrm{E}$, a contradiction. Thus, there exists a tally set $T = \mathrm{tally}(L)$ in $\mathcal{K} - \mathrm{P}$. Conversely, let $T$ be some tally set in $\mathcal{K} - \mathrm{P}$. A similar argument as above—now using that $\mathfrak{R}_m^e(\mathcal{K}) \subseteq \mathcal{L}$ and $\mathfrak{R}_{m,\,e\ell d}^p(\mathrm{E}) \subseteq \mathrm{P}$—shows that the binary encoding of $T$, $L = \mathrm{bin}(T)$, is in $\mathcal{L} - \mathrm{E}$. □

## 4.3 Upward Separation Results

Recall from Chapter 2 the definition of the FEW operator.

**Definition 4.3.1** Let $\mathcal{K}$ be any polynomial-time bounded complexity class. A set $L$ is in $\mathrm{FEW} \cdot \mathcal{K}$ if and only if there exist a set $A \in \mathcal{K}$ and polynomials $p$ and $q$ such that for every $x \in \Sigma^*$,

1. $\|\{y \mid \langle x, y\rangle \in A \wedge |y| = p(|x|)\}\| \leq q(|x|)$, and
2. $x \in L \iff \|\{y \mid \langle x, y\rangle \in A \wedge |y| = p(|x|)\}\| > 0$.

In this section, we provide a structural sufficient condition for upward separation. We show that any polynomial-time bounded complexity class $\mathcal{K}$ that is closed under the FEW operator possesses this property.

Clearly, $\mathrm{FEW} \cdot \mathrm{P} = \mathrm{FEW} \cdot \mathrm{UP} = \mathrm{FEW} \cdot \mathrm{FewP} = \mathrm{FewP}$. Furthermore, $\mathrm{FEW} \cdot \mathcal{K} \subseteq \mathrm{FewP}^{\mathcal{K}}$ for any class $\mathcal{K}$. By Lemma 3.4.5 from the previous chapter, if $\mathcal{K}$ is closed under truth-table reductions, then we even have $\mathrm{FewP}^{\mathcal{K}} = \mathrm{FEW} \cdot \mathcal{K}$.

Note that Definition 4.3.1 doesn't work for exponential-time bounded classes; in particular, FewE and $\mathrm{FEW} \cdot \mathrm{E}$ are probably not the same (see Footnote 2). As we'll apply the

Few operator to polynomial-time bounded complexity classes only, however, this causes no problems here.

In this chapter, we focus on the following (promise and non-promise) counting classes: UP, FewP, $\oplus$P, PP, $\mathrm{C}_{=}\mathrm{P}$, SPP, and LWPP. Below we summarize the known relations among these classes and state some known properties to be applied in the proof of Corollary 4.3.7.

**Fact 4.3.2**

1. UP $\subseteq$ FewP $\subseteq$ NP $\subseteq$ co$\mathrm{C}_{=}\mathrm{P}$ $\subseteq$ PP.
2. FewP $\subseteq$ SPP $\subseteq$ LWPP $\subseteq$ $\mathrm{C}_{=}\mathrm{P}$ $\subseteq$ PP.
3. SPP $\subseteq$ $\oplus$P.
4. [PZ83, FFK94] $\oplus$P and SPP are self-low.
5. [FFK94] $\mathrm{SPP}^{\mathrm{LWPP}} = \mathrm{LWPP}$.

**Remark 4.3.3**

1. All the results in Fact 4.3.2 relativize, i.e., they hold relative to every oracle. Note that the inclusions given in this fact straightforwardly translate into operator notation. For instance, $\mathrm{Few} \cdot \mathcal{K} \subseteq \exists \cdot \mathcal{K}$ holds for any class $\mathcal{K}$. The proof of the self-lowness of $\oplus$P is due to Papadimitriou and Zachos [PZ83]. Using a similar technique, Fenner, Fortnow, and Kurtz have shown this property to hold for SPP as well [FFK91].
2. As a sidenote, Köbler, Schöning, and Torán proved the interesting result that SPP contains the graph automorphism problem and LWPP contains the graph isomorphism problem [KST92]. This combined with the results of Fact 4.3.2 implies that these two problems are low for various counting classes such as $\mathrm{C}_{=}\mathrm{P}$ and PP.
3. In [RRW94], we claimed that, among several other classes, the promise class LWPP is closed under the Few operator and thus displays upward separation. Though this result indeed is valid, we note here that the proof given in [RRW94] is not correct, since the proof that LWPP is self-low (claimed in [FFK91] and referred to in [RRW94]) is not correct. That is, referring to Fenner, Fortnow, and Kurtz's claim that the proof of the self-lowness of SPP can be modified so as to establish the self-lowness of LWPP [FFK91], we conclude in [RRW94] that LWPP is closed under the Few operator, and therefore displays upward separation. In the journal version [FFK94]

of their paper, however, Fenner, Fortnow, and Kurtz withdraw their claim that LWPP was self-low, reasoning that the way LWPP is relativized causes problems. On the other hand, they notice that the self-lowness proof for SPP can indeed be modified so as to establish the weaker claim stated in Part 5 of the above fact, which already suffices to prove that $\text{F}\textsc{ew} \cdot \text{LWPP} = \text{LWPP}$, since clearly $\text{F}\textsc{ew} \cdot \text{LWPP} \subseteq \text{SPP}^{\text{LWPP}}$ by Part 2 of Fact 4.3.2, Lemma 3.4.5, and the fact that LWPP is closed under truth-table (and even Turing) reductions due to Part 5 of Fact 4.3.2. A corrigendum to [RRW94] has been sent to the journal *Information Processing Letters* in April, 1995.

The fact that LWPP may fail to be self-low in the machine-based setting notwithstanding, this corrigendum in addition proves that in the operator-based setting, LWPP indeed is "self-low," i.e., LWPP is closed under the Lwp operator, which is defined by

$$\text{L}\textsc{wp} \cdot \mathcal{K} \stackrel{\text{df}}{=} \{L \mid (\exists f \in \text{GAP} \cdot \mathcal{K})\ (\exists g \in \text{FP};\ g : \mathbb{N} \rightarrow \mathbb{N}^+)\ (\forall w)\ [g(|w|) \cdot \chi_L(w) = f(w)]\}.$$

Below we give a short description of Buhrman, E. Hemaspaandra, and Longpré's tally encoding of sparse sets (see [BHL] for some algebraic background that explains the specific choice of the parameters), who introduced this coding to prove the surprising result that any sparse set $S$ conjunctively truth-table reduces to the tally set $\text{BLS}(S)$. Their coding is central to the proof of our main result (Theorem 4.3.6).

**Definition 4.3.4 (BLS encoding of sparse sets)** [BHL] Let $S$ be any sparse set of density $d$ for some polynomial $d$. For fixed $n \geq 0$, define $r(n) \stackrel{\text{df}}{=} \left\lceil \frac{2n}{\log n} \right\rceil$ and let $p_{n,d}$ be the smallest prime larger than $r(n) \cdot d(n)$. Consider the finite field $\text{GF}(p_{n,d})$ with $p_{n,d}$ elements. As each polynomial over $\text{GF}(p_{n,d})$ of degree $\leq r(n)$ can be represented by its $r(n) + 1$ coefficients, it may be viewed as an $(r(n)+1)$-digit number in base $p_{n,d}$. Thus, each string $x \in \Sigma^n$ corresponds to some polynomial

$$q_x(a) \stackrel{\text{df}}{=} x_{r(n)}a^{r(n)} + x_{r(n)-1}a^{r(n)-1} + \cdots + x_1 a + x_0,$$

where $x_{r(n)} \cdots x_0$ is the representation of $x$ in base $p_{n,d}$ with leading zeros. To encode the length $n$ strings of $S$, define the $n$th segment of the tally set $\text{BLS}(S) \stackrel{\text{df}}{=} \bigcup_{n \geq 0} T_n(S)$ by

$$T_n(S) \stackrel{\text{df}}{=} \left\{ 0^{\langle n, a, q_x(a) \rangle} \,\middle|\, a \in \text{GF}(p_{n,d}) \wedge x \in S^{=n} \right\}.$$

**Lemma 4.3.5** For any class $\mathcal{K}$, if $S \in \text{SPARSE} \cap \mathcal{K}$, then $\text{BLS}(S)$ is a tally set in $\text{FEW} \cdot \mathcal{K}$.

**Proof.** Let $S$ be any sparse set in $\mathcal{K}$ of density $d$ for some polynomial $d$. Consider the following algorithm for $\text{BLS}(S)$: On input $0^{\langle n,a,b \rangle}$, guess a string $x$ of length $n$, compute $r(n)$ and $p_{n,d}$ in polynomial time, and verify $a \in \text{GF}(p_{n,d})$ and $q_x(a) = b$. If this is not the case, then reject, otherwise simulate the $\mathcal{K}$ machine for $S$ on input $x$ and accept accordingly. Since there are only a polynomial number of strings in $S^{=n}$, this shows that $\text{BLS}(S) \in \text{FEW} \cdot \mathcal{K}$. $\square$

**Theorem 4.3.6** Let $(\mathcal{K}, \mathcal{L})$ be an associated pair such that $\text{P} \subseteq \mathcal{K}$ and $\text{FEW} \cdot \mathcal{K} = \mathcal{K}$. Then, $\mathcal{K} - \text{P}$ contains sparse sets if and only if $\mathcal{L} \neq \text{E}$.

**Proof.** The "if" part holds by Lemma 4.2.2. For proving the "only if," we show the contrapositive: the supposition $\mathcal{L} = \text{E}$ forces all sparse sets from $\mathcal{K}$ into P. Suppose $\mathcal{L} = \text{E}$, and let $S$ be any sparse set in $\mathcal{K}$. By Lemma 4.3.5, $\text{BLS}(S) \in \text{FEW} \cdot \mathcal{K} = \mathcal{K}$. Thus, $\text{bin}(\text{BLS}(S))$ is in $\mathcal{L}$, which equals E by our supposition. Hence, $\text{tally}(\text{bin}(\text{BLS}(S))) = \text{BLS}(S)$ is in P, and since $S$ conjunctively truth-table reduces to $\text{BLS}(S)$, it follows that $S \in \text{P}$. $\square$

**Corollary 4.3.7** Let $\mathcal{K}$ be any of the classes FewP, NP, $\text{co}\mathrm{C_=P}$, $\oplus\text{P}$, SPP, or LWPP, and let $(\mathcal{K}, \mathcal{L})$ be the respective associated pair. Then, $(\mathcal{K}, \mathcal{L})$ displays upward separation, that is, $\mathcal{K} - \text{P}$ contains sparse sets if and only if $\mathcal{L} \neq \text{E}$.

**Proof.** By Theorem 4.3.6, it suffices to show that each of the classes $\mathcal{K}$ considered is closed under the FEW operator. This is easily observed for FewP. For $\mathcal{K} = \text{NP}$ and $\mathcal{K} = \text{co}\mathrm{C_=P}$, the result follows from the well-known or obvious facts that $\text{FEW} \cdot \mathcal{K} \subseteq \exists \cdot \mathcal{K}$, $\exists \cdot \text{NP} = \text{NP}$ [Sto77, MS72], and $\forall \cdot \mathrm{C_=P} = \mathrm{C_=P}$ (see, e.g., [Tod91]). Thus, we have $\text{FEW} \cdot \text{NP} \subseteq \exists \cdot \text{NP} = \text{NP}$ and $\text{FEW} \cdot \text{co}\mathrm{C_=P} \subseteq \exists \cdot \text{co}\mathrm{C_=P} = \text{co}\forall \cdot \mathrm{C_=P} = \text{co}\mathrm{C_=P}$. If $\mathcal{K}$ is chosen from $\{\oplus\text{P}, \text{SPP}, \text{LWPP}\}$, then Lemma 3.4.5 and the relativized version of Fact 4.3.2 imply $\text{FEW} \cdot \mathcal{K} \subseteq \text{FewP}^{\mathcal{K}} \subseteq \mathcal{K}^{\mathcal{K}} = \mathcal{K}$, since any class which is self-low, clearly is closed under truth-table (and even Turing) reductions.[3] $\square$

Note that, in the above proof, there is nothing special about the mod 2 defining $\oplus\text{P}$ [PZ83, GP86]—all we need is its self-lowness and that $\text{FewP} \subseteq \oplus\text{P}$ [CH90]. Thus, the result holds as well for all classes $\text{Mod}_p\text{P}$ (defined in [CH90, BG92, Her90]), for prime $p$.

[3] Regarding LWPP, see the discussion in Remark 3.

## 4.4 Conclusions and Open Problems

We have presented several new upward separation results contrasting recently discovered results about some promise classes that fail to have upward separation in all relativized worlds. As an immediate consequence, this, combined with the fact that equality of classes obeys standard upward translation, yields relativizations separating any two classes that differ in their property of displaying or defying upward separation, e.g., $\mathrm{BPP}^A \neq \oplus\mathrm{P}^A$, $\mathrm{FewP}^A \neq \mathrm{ZPP}^A$, etc., where $A$ is the oracle constructed in [HJ93]. More precisely, the proof of, e.g., $(\exists A)$ $[\mathrm{BPP}^A \neq \oplus\mathrm{P}^A]$ is as follows: Suppose $\mathrm{BPP}^B = \oplus\mathrm{P}^B$ for all oracles $B$. Then, by standard padding arguments, $\mathrm{BPE}^B = \oplus\mathrm{E}^B$ for all oracles $B$. But there exists an oracle $A$ (constructed in [HJ93]) such that $\mathrm{BPE}^A = \oplus\mathrm{E}^A = \mathrm{E}^A$ and yet $\mathrm{BPP}^A = \oplus\mathrm{P}^A$ contains sparse sets not in $\mathrm{P}^A$, which contradicts that, by the relativized version of Corollary 4.3.7, $\oplus\mathrm{P}^A - \mathrm{P}^A$ lacks sparse sets if $\oplus\mathrm{E}^A = \mathrm{E}^A$. Observe also that Corollary 4.3.7 adds "$\mathrm{FewE} \neq \mathrm{E}$" to Allender and Rubinstein's [AR88] list of characterizations of the existence of sparse sets in P that are not P-printable [HY84], a notion arising in the study of generalized Kolmogorov complexity and data compression.

In particular, we have invalidated the conjecture that a class must not be defined in a promise-like way to possess upward separation by giving the counterexamples of FewP, SPP, and LWPP. However, our technique does not seem to apply to the promise classes UP or $\mathrm{NP} \cap \mathrm{coNP}$, and neither does it seem to apply to the non-promise classes PP or $\mathrm{C_{=}P}$. Although Theorem 4.3.6 immediately gives upward separation results for some exotic classes such as $\mathrm{FEW} \cdot \mathrm{PP}$ or $\mathrm{FEW} \cdot \mathrm{C_{=}P}$ that are trivially closed under the FEW operator, it does not apply to PP or $\mathrm{C_{=}P}$ itself, as these classes are unlikely to satisfy the assumption of the theorem. For instance, supposing PP were closed under the FEW operator, then the closure of PP under truth-table reductions [FR91] implies $\mathrm{P}^{\mathrm{PP}} \subseteq \mathrm{FewP}^{\mathrm{PP}} = \mathrm{FEW} \cdot \mathrm{PP} = \mathrm{PP}$ by Lemma 3.4.5, thus settling the major open question of whether PP is closed under Turing reductions. Likewise, $\mathrm{FEW} \cdot \mathrm{UP} = \mathrm{UP}$ is equivalent to $\mathrm{FewP} = \mathrm{UP}$, another important open problem.

Regarding PP, all we can prove is the following weak result: If $\mathrm{BPP} - \mathrm{P}$ contains sparse sets, then $\mathrm{PE} \neq \mathrm{E}$. For proving the contrapositive, consider any sparse set $S \in \mathrm{BPP}$. By Lemma 4.3.5 and since $\mathrm{FewP} \subseteq \mathrm{PP}$ and BPP is low for PP [KST+93], we have $\mathrm{BLS}(S) \in \mathrm{FEW} \cdot \mathrm{BPP} \subseteq \mathrm{PP}^{\mathrm{BPP}} = \mathrm{PP}$. Then, as in the proof of Theorem 4.3.6, the hypothesis $\mathrm{PE} = \mathrm{E}$ implies that $S \in \mathrm{P}$. Clearly, this applies to every class that is low for PP.

Regarding $\mathrm{C}_{=}\mathrm{P}$, we conjecture that (unless closed under complementation) it resembles coNP in that it also fails to robustly have upward separation, as is suggested by the fact that their respective classes of (set-wise) complements, $\mathrm{coC}_{=}\mathrm{P}$ and NP, possess this property jointly.

# Chapter 5

# Multi-Selectivity and Complexity-Lowering Joins

## 5.1 Introduction

Selman introduced the P-selective sets (P-Sel, for short) [Sel79] as the complexity-theoretic analogs of Jockusch's semi-recursive sets [Joc68]: a set is P-selective if there exists a polynomial-time transducer (henceforward called a selector) that, given any two input strings, outputs one that is logically no less likely to be in the set than the other one. There has been much progress recently in the study of P-selective sets (see the survey [DHHT94]). In this paper, we introduce a more flexible notion of selectivity that allows the selector to operate on multiple input strings, and that thus generalizes Selman's P-selectivity in the following promise-like way: Depending on two parameters, say $i$ and $j$ with $i \geq j \geq 1$, a set $L$ is $(i, j)$-selective if there is a selector that, given any finite set of distinct input strings, outputs some subset of at least $j$ elements each belonging to $L$ if $L$ contains at least $i$ of the input strings; otherwise, it may output an arbitrary subset of the inputs.

This hierarchy of generalized selectivity classes (denoted by SH) is studied in Section 5.2. First we show that only the difference of $i$ and $j$ is relevant in the above definition of $(i, j)$-selectivity: a set $L$ is $(i, j)$-selective if and only if $L$ is $(i - j + 1, 1)$-selective. Let $S(k)$ denote the class of $(k, 1)$-selective sets. Clearly, $S(1) =$ P-Sel and for each $k \geq 1$, $S(k) \subseteq S(k+1)$. We further show that SH is properly infinite, and we relatedly prove that, unlike P-Sel, none of the $S(k)$ for $k \geq 2$ is closed under $\leq_m^p$-reductions, and also that sets in $S(2)$ that are many-one reducible to their complements may already go beyond P, which contrasts with Selman's result that a set $A$ is in P if and only if $A \leq_m^p \overline{A}$ and $A$ is P-selective [Sel79]. Consequently, the class P cannot be characterized by the auto-reducible sets in any of the higher levels of SH.

Ogihara [Ogi94] has recently introduced the polynomial-time membership-comparable (P-mc, for short) sets as another generalization of the P-selective sets. Since P-mc$(k)$ (see Definition 5.2.10) is closed under $\leq_{1\text{-}tt}^p$-reductions for each $k$ [Ogi94], it is clear that Ogihara's approach to generalized selectivity is different from ours, and in Theorem 5.2.12, we completely establish, in terms of incomparability and strict inclusion, the relations between his and our generalized selectivity classes. In particular, since P-mc$(poly)$ is contained in P/poly [Ogi94] and SH is (strictly) contained in P-mc$(poly)$, it follows that every set in SH has polynomial-size circuits. On the other hand, P-selective NP sets can even be shown to be in $\mathrm{Low}_2$ [KS85]. Since such a result is not known to hold for the polynomial-time membership-comparable NP sets, our $\mathrm{Low}_2$-ness results in Theorem 5.2.16 are the strongest known for generalized selectivity-like classes.[1]

Selman proved that NP-complete sets such as SAT cannot be P-selective unless $\mathrm{P} = \mathrm{NP}$ [Sel79]. Ogihara extended this collapse result to the case of certain P-mc classes strictly larger than P-Sel. By the inclusions stated in Theorem 5.2.12, this extension applies to many of our selectivity classes as well; in particular, SH cannot contain all of NP unless $\mathrm{P} = \mathrm{NP}$.

To summarize, this demonstrates that the core results holding for the P-selective sets, and proving them structurally simply, also hold for SH.

An even stronger motivation for introducing and studying generalized selectivity is given in Section 5.3, in which we establish a result that sharply contrasts with a known result about P-Sel. Though P-Sel $\subseteq \mathrm{EL}_2$, we prove that not all sparse sets in SH are in $\mathrm{EL}_2$. This is the strongest known $\mathrm{EL}_2$ lower bound, strengthening the result that P/poly, and indeed SPARSE, is not contained in $\mathrm{EL}_2$ [AH92]. The proof of this result also establishes that $\mathrm{EL}_2$ is not closed under certain Boolean operations such as intersection and union. Relatedly, we prove that there exist sets that are not in $\mathrm{EL}_2$, yet their join (marked union) is in $\mathrm{EL}_2$. That is, *in terms of extended lowness, the join operator can lower complexity*.

---

[1] A bit more carefully rephrased, this sentence would say: "... *have been* the strongest known for generalized selectivity-like classes until Köbler extended them even further in [Köb95], simultaneously subsuming some results of [ABG90, HNOS94]." See Footnote 4 on page 70 for more details.

It is known that P-Sel is not closed under union or intersection [HJ]. However, in Section 5.4, we provide an extended selectivity hierarchy that is based on SH and is large enough to capture those closures of the P-selective sets, and yet, in contrast with the P-mc classes, is refined enough to distinguish them.

# 5.2 A Basic Hierarchy of Generalized Selectivity Classes

## 5.2.1 Structure, Properties, and Relationships with P-mc Classes

Before we define our generalized concept of selectivity, a technical remark is in order. Each selector function considered in this chapter is computed by a polynomial-time transducer that takes a set of strings as input and outputs some set of strings. As the order of the strings in these sets doesn't matter, we may assume that, without loss of generality, they are given in lexicographical order (i.e., $x_1 \leq_{\text{lex}} x_2 \leq_{\text{lex}} \cdots \leq_{\text{lex}} x_m$), and are coded into one string over $\Sigma$ using our pairing function. As a notational convenience, we'll identify these sets with their codings and simply write (unless a more complete notation is needed) $f(x_1, \ldots, x_m)$ to indicate that selector $f$ runs on the inputs $x_1, \ldots, x_m$ coded as $\langle x_1, \ldots, x_m \rangle$.

**Definition 5.2.1** Let $g_1$ and $g_2$ be non-decreasing functions from $\mathbb{N}^+$ into $\mathbb{N}^+$ (henceforward called *threshold functions*) such that $g_1 \geq g_2$. $S(g_1(n), g_2(n))$ is the class of all sets $L$ for which there exists an FP function $f$ such that for each $n \geq 1$ and any distinct input strings $y_1, \ldots, y_n$,

1. $f(y_1, \ldots, y_n) \subseteq \{y_1, \ldots, y_n\}$, and
2. $\|L \cap \{y_1, \ldots, y_n\}\| \geq g_1(n) \Longrightarrow (f(y_1, \ldots, y_n) \subseteq L \land \|f(y_1, \ldots, y_n)\| \geq g_2(n))$.

We also consider classes fair-$S(g_1(n), g_2(n))$ in which the selector $f$ is required to satisfy the above conditions only when applied to any $n$ distinct input strings each having length at most $n$. As a notational convention, for *non*-constant threshold functions, we will use "expressions in $n$," and we use $i$, $j$, or $k$ if the threshold is constant. The definition immediately implies the following:

**Fact 5.2.2** Let $g_1$, $g_2$, and $c$ be threshold functions such that $g_1 \geq g_2$.

1. $S(g_1(n), g_2(n)) \subseteq S(g_1(n) + c(n), g_2(n))$ and
   $S(g_1(n), g_2(n) + c(n)) \subseteq S(g_1(n), g_2(n))$.
   These inclusions also hold for the corresponding fair-S classes.

2. If $g_1(n) \geq n$ for any $n$, then $S(g_1(n), g_2(n)) =$ fair-$S(g_1(n), g_2(n)) = \mathfrak{P}(\Sigma^*)$.

3. $S(g_1(n), g_2(n)) \subseteq$ fair-$S(g_1(n), g_2(n)) \subseteq$ fair-$S(n-1, 1)$ if $g_2(n) \leq g_1(n) < n$ for any $n$.

In particular, we are interested in classes $S(i, j)$ parameterized by constants $i$ and $j$. Theorem 5.2.3 reveals that, in fact, there is only *one* significant parameter, the difference of $i$ and $j$. This suggests the simpler notation $S(k) \stackrel{\text{df}}{=} S(k, 1)$ for all $k \geq 1$. Let SH denote the hierarchy $\bigcup_{k \geq 1} S(k)$. For simplicity, we henceforward (i.e., after the proof of Theorem 5.2.3) assume that selectors for any set in SH select *exactly one* input string rather than a subset of the inputs (i.e., they are viewed as FP functions mapping into $\Sigma^*$ rather than into $\mathfrak{P}(\Sigma^*)$).

**Theorem 5.2.3** $(\forall i \geq 1)\ (\forall k \geq 0)\ [S(i, 1) = S(i + k, 1 + k)]$.

**Proof.** For any fixed $i \geq 1$, the proof is done by induction on $k$. The induction base is trivial. Assume $S(i, 1) = S(i + k - 1, k)$ for $k > 0$. We show $S(i, 1) = S(i + k, 1 + k)$. For the first inclusion, assume $L \in S(i, 1)$, and let $f$ be an $S(i+k-1, k)$-selector for $L$ that exists by the inductive hypothesis. Given any distinct input strings $y_1, \ldots, y_m$, $m \geq 1$, an $S(i + k, 1 + k)$-selector $g$ for $L$ is defined by

$$g(y_1, \ldots, y_m) \stackrel{\text{df}}{=} \begin{cases} f(\{y_1, \ldots, y_m\} - \{z\}) \cup \{z\} & \text{if } f(y_1, \ldots, y_m) \neq \emptyset \\ Y & \text{otherwise,} \end{cases}$$

where $z \in f(y_1, \ldots, y_m)$ and $Y$ is an arbitrary subset of $\{y_1, \ldots, y_m\}$. Clearly, $g \in$ FP, $g(y_1, \ldots, y_m) \subseteq \{y_1, \ldots, y_m\}$, and if $\|L \cap \{y_1, \ldots, y_m\}\| \geq i + k$, then $g$ outputs at least $1 + k$ strings each belonging to $L$. Thus, $L \in S(i + k, 1 + k)$ via $g$.

For the converse inclusion, let $L \in S(i + k, 1 + k)$ via $g$. To define an $S(i + k - 1, k)$-selector $f$ for $L$, let $i + k$ strings $z_1, \ldots, z_{i+k} \in L$ (w.l.o.g., $L$ is infinite) be hardcoded into the machine computing $f$. Given $y_1, \ldots, y_m$ as input strings, $m \geq 1$, define

$$f(y_1, \ldots, y_m) \stackrel{\text{df}}{=} \begin{cases} g(y_1, \ldots, y_m) & \text{if } \{z_1, \ldots, z_{i+k}\} \subseteq \{y_1, \ldots, y_m\} \\ g(y_1, \ldots, y_m, z) - \{z\} & \text{otherwise,} \end{cases}$$

where $z \in \{z_1, \dots, z_{i+k}\} - \{y_1, \dots, y_m\}$. Clearly, $f \in$ FP, $f(y_1, \dots, y_m) \subseteq \{y_1, \dots, y_m\}$, and if $\|L \cap \{y_1, \dots, y_m\}\| \geq i + k - 1$, then $f$ outputs at least $k$ elements of $L$. Thus, $f$ witnesses that $L \in S(i + k - 1, k)$, which equals $S(i, 1)$ by the inductive hypothesis. □

**Fact 5.2.4** 1. $S(1) =$ P-Sel.

2. $(\forall k \geq 1)$ $[S(k) \subseteq S(k + 1)]$.

**Proof.** By definition, we have immediately Part 2 and the inclusion from left to right in Part 1, as in particular, given any pair of strings, an $S(1)$-selector $f$ is required to select a string (recall our assumption that all $S(k)$-selectors output *exactly* one input string) that is no less likely to be in the set than the other one. For the converse inclusion, fix any set of inputs $y_1, \dots, y_m$, $m \geq 1$, and let $f$ be a P-selector for L. Play a knock-out tournament among the strings $y_1, \dots, y_m$, where $x$ beats $y$ if and only if $f(x, y) = x$. Let $y_w$ be the winner. Clearly, $g(y_1, \dots, y_m) \stackrel{\text{df}}{=} y_w$ witnesses that $L \in S(1)$. □

Recall that, by convention, the "$n - 1$" in fair-S$(n - 1, 1)$ denotes the non-constant threshold functions $g(n) \stackrel{\text{df}}{=} n - 1$. Next we prove that SH is properly infinite and is strictly contained in fair-S$(n - 1, 1)$. Fix an enumeration $\{f_i\}_{i \geq 1}$ of FP functions, and define $e(0) \stackrel{\text{df}}{=} 2$ and $e(k) \stackrel{\text{df}}{=} 2^{e(k-1)}$ for $k \geq 1$. For any $i \geq 0$ and $s \leq 2^{e(i)}$, let $W_{i,s} \stackrel{\text{df}}{=} \{w_{i,1}, \dots, w_{i,s}\}$ be an enumeration of the lexicographically smallest $s$ strings in $\Sigma^{e(i)}$ (this notation will be used also in Section 5.4).

**Theorem 5.2.5** 1. For each $k \geq 1$, $S(k) \subset S(k + 1)$.

2. SH $\subset$ fair-S$(n - 1, 1)$.

**Proof.** 1. For fixed $k \geq 1$, choose $k + 1$ pairwise distinct strings $b_0, \dots, b_k$ of the same length. Define

$$A_k \stackrel{\text{df}}{=} \bigcup_{i \geq 1} \left( \left\{ b_0^{e(i)}, \dots, b_k^{e(i)} \right\} - \left\{ f_i(b_0^{e(i)}, \dots, b_k^{e(i)}) \right\} \right),$$

i.e., for each $i \geq 1$, $A_k$ can lack at most one out of the $k + 1$ strings $b_0^{e(i)}, \dots, b_k^{e(i)}$.

An $S(k + 1)$-selector $g$ for $A_k$ is given in Figure 5.1 below. W.l.o.g., assume each input in $Y = \{y_1, \dots, y_m\}$ to be of the form $b_j^{e(i)}$ for some $j \in \{0, \dots, k\}$ and $i \in \{i_1, \dots, i_s\}$, where $1 \leq i_1 < \dots < i_s$ and $s \leq m$. Clearly, $g(Y) \in Y$. Let $n = |\langle y_1, \dots, y_m \rangle|$. Since

**Description of an $S(k+1)$-selector $g$.**

**input** $Y = \{y_1, \ldots, y_m\}$
**begin** $t := s - 1$;
  **while** $t \geq 1$ **do**
    $Z := \left\{ y \in Y \,\middle|\, (\exists j \in \{0, \ldots, k\})\, [y = b_j^{e(i_t)}] \right\} - \left\{ f_{i_t}(b_0^{e(i_t)}, \ldots, b_k^{e(i_t)}) \right\}$;
    **if** $Z \neq \emptyset$ **then output** some element of $Z$ and **halt**
    **else** $t := t - 1$
  **end while**
  **output** an arbitrary input string and **halt**
**end**

**End of description of $g$.**

Figure 5.1: An $S(k+1)$-selector $g$ for $A_k$.

there are at most $m$ while loops to be executed and the polynomial-time transducers $f_{i_t}$, $t < s$, run on inputs of length at most $c \cdot \log e(i_s)$ for some constant $c$, the runtime of $g$ on *that* input is bounded above by some polylogarithmic function in $n$. Then, there is a polynomial in $n$ bounding $g$'s runtime on *any* input. Thus, $g \in \mathrm{FP}$. If some element $y$ is output during the while loop, then $y \in A_k$. If $g$ outputs an arbitrary input string after exiting the while loop, then no input of the form $b_j^{e(i_t)}$, $t < s$, is in $A_k$, and since $A_k$ has at most $k+1$ strings at each length, we have $\|A_k \cap Y\| \leq k$ if $g(Y) \notin A_k$. Thus, $A_k \in S(k+1)$ via $g$.

On the other hand, each potential $S(k)$-selector $f_i$, given $b_0^{e(i)}, \ldots, b_k^{e(i)}$ as input strings, outputs an element not in $A_k$ though $k$ of these strings are in $A_k$. Thus, $A_k \notin S(k)$.

2. Fix any $k \geq 1$, and let $L \in S(k)$ via selector $f$. For each of the finitely many tuples $y_1, \ldots, y_\ell$ such that $\ell \leq k$ and $|y_i| \leq \ell$, $1 \leq i \leq \ell$, let $z_{y_1,\ldots,y_\ell}$ be some fixed string in $L \cap \{y_1, \ldots, y_\ell\}$ if this set is non-empty, and an arbitrary string from $\{y_1, \ldots, y_\ell\}$ otherwise. Let these fixed strings be hardcoded into the machine computing the function $g$ defined by

$$g(y_1, \ldots, y_n) \stackrel{\mathrm{df}}{=} \begin{cases} \{z_{y_1,\ldots,y_n}\} & \text{if } n \leq k \\ \{f(y_1, \ldots, y_n)\} & \text{otherwise.} \end{cases}$$

Thus, $L \in \text{fair-}S(n-1, 1)$ via $g$, showing that $\mathrm{SH} \subseteq \text{fair-}S(n-1, 1)$.

The strictness of the inclusion is proven as in Part 1 of this proof. To define a set $A \notin$ SH we have here to diagonalize against all potential selectors $f_j$ and all levels of SH simultaneously. That is, in stage $i = \langle j, k \rangle$ of the construction of $A \stackrel{\text{df}}{=} \bigcup_{i \geq 1} A_i$, we will diagonalize against $f_j$ being an $S(k)$-selector for $A$. Fix $i = \langle j, k \rangle$. Recall that $W_{i,k+1}$ is the set of the smallest $k+1$ length $e(i)$ strings. Note that $2^{e(i)} \geq k+1$ holds for each $i$, since we can w.l.o.g. assume that the pairing function satisfies $u > \max\{v, w\}$ for all $u, v$, and $w$ with $u = \langle v, w \rangle$. Define $A_i \stackrel{\text{df}}{=} W_{i,k+1} - \{f_j(W_{i,k+1})\}$. Assume $A \in$ SH, i.e., there exists some $t$ such that $A \in S(t)$ via some selector $f_s$. But this contradicts that for $r = \langle s, t \rangle$, by construction of $A$, we have $\|A \cap W_{r,t+1}\| \geq t$, yet $f_s(W_{r,t+1})$ either doesn't output one of its inputs (and is thus no selector), or $f_s(W_{r,t+1}) \notin A$. Thus, $A \notin$ SH.

Now we prove that $A$ trivially is in fair-$S(n-1, 1)$, as $A$ is constructed such that the promise is *never* met. By way of contradiction, suppose a set $X$ of inputs is given, $\|X\| = n$, $\|A \cap X\| \geq n-1$, and $|x| \leq n$ for each $x \in X$. Let $e(i)$ be the maximum length of the strings in $A \cap X$, i.e., $A \cap X = \bigcup_{m=1}^{i} A_m \cap X$. Let $j$ and $k$ be such that $i = \langle j, k \rangle$. Since (by the above remark about our pairing function) $k+1 \leq i$, we have by construction of $A$,

$$e(i) - 1 \leq n - 1 \leq \|A \cap X\| = \|\bigcup_{m=1}^{i} A_m \cap X\| \leq \|\bigcup_{m=1}^{i} A_m\| \leq (k+1)i \leq i^2,$$

which is false for all $i \geq 0$. Hence, $A \in$ fair-$S(n-1, 1)$. □

A variation of this technique proves that, unlike P-Sel, none of the $S(k)$ for $k \geq 2$ is closed under $\leq_m^p$-reductions. (Of course, every class $S(k)$ is closed downwards under polynomial-time one-one reductions.) We also show that sets in $S(2)$ that are many-one reducible to their complements may already go beyond P, which contrasts with Selman's result that a set $A$ is in P if and only if $A \leq_m^p \overline{A}$ and $A$ is P-selective [Sel79]. It follows that the class P cannot be characterized by the auto-reducible sets (see [BvHT93]) in any of the higher classes in SH. It would be interesting to strengthen Corollary 5.2.7 to the case of the *self*-reducible sets, as that would contrast sharply with Buhrman, van Helden, and Torenvliet's characterization of P as those self-reducible sets that are in P-Sel [BvHT93].

**Theorem 5.2.6** 1. For each $k \geq 2$, $S(k) \subset \mathfrak{R}_m^p(S(k))$.

2. There exists a set $A$ in $S(2)$ such that $A \leq_m^p \overline{A}$ and yet $A \notin$ P.

**Corollary 5.2.7** There exists an auto-reducible set in $S(2)$ that is not in P.

**Proof of Theorem 5.2.6.** 1. In fact, we will define a set $L \in R^p_m(S(2)) - S(k)$. By Fact 5.2.4, the theorem follows. Choose $2k$ pairwise distinct strings $b_1, \ldots, b_{2k}$ of the same length. Define $L \stackrel{\mathrm{df}}{=} A_i \cup B_i$, where

$$A_i \stackrel{\mathrm{df}}{=} \begin{cases} \{b_1^{e(i)}, \ldots, b_k^{e(i)}\} & \text{if } f_i(b_1^{e(i)}, \ldots, b_{2k}^{e(i)}) \notin \{b_1^{e(i)}, \ldots, b_k^{e(i)}\} \\ \emptyset & \text{otherwise,} \end{cases}$$

$$B_i \stackrel{\mathrm{df}}{=} \begin{cases} \{b_{k+1}^{e(i)}, \ldots, b_{2k}^{e(i)}\} & \text{if } f_i(b_1^{e(i)}, \ldots, b_{2k}^{e(i)}) \notin \{b_{k+1}^{e(i)}, \ldots, b_{2k}^{e(i)}\} \\ \emptyset & \text{otherwise.} \end{cases}$$

Clearly, each potential $S(k)$-selector $f_i$, given $b_1^{e(i)}, \ldots, b_{2k}^{e(i)}$ as input strings, outputs an element not in $L$ though $\|L \cap \{b_1^{e(i)}, \ldots, b_{2k}^{e(i)}\}\| \geq k$. Thus, $L \notin S(k)$.

Now define a set

$$L' \stackrel{\mathrm{df}}{=} \{b_1^{e(i)} \mid b_1^{e(i)} \in L\} \cup \{b_{k+1}^{e(i)} \mid b_{k+1}^{e(i)} \in L\}$$

and an FP function $g$ by $g(b_j^{e(i)}) \stackrel{\mathrm{df}}{=} b_1^{e(i)}$ if $1 \leq j \leq k$, and $g(b_j^{e(i)}) \stackrel{\mathrm{df}}{=} b_{k+1}^{e(i)}$ if $k+1 \leq j \leq 2k$, and $g(x) = x$ for all $x$ not of the form $b_j^{e(i)}$ for any $i \geq 1$ and $j$, $1 \leq j \leq 2k$. Then, we have $x \in L$ if and only if $g(x) \in L'$ for each $x \in \Sigma^*$, that is, $L \leq^p_m L'$.

Now we show that $L' \in S(2)$. Given any distinct inputs $y_1, \ldots, y_n$ (each having, without loss of generality, the form $b_1^{e(i)}$ or $b_{k+1}^{e(i)}$ for some $i \geq 1$), define an $S(2)$-selector as follows:

**Case 1:** All inputs have the same length. Then, $\{y_1, \ldots, y_n\} \subseteq \{b_1^{e(i)}, b_{k+1}^{e(i)}\}$ for some $i \geq 1$. Define $f(y_1, \ldots, y_n)$ to be $b_1^{e(i)}$ if $b_1^{e(i)} \in \{y_1, \ldots, y_n\}$, and to be $b_{k+1}^{e(i)}$ otherwise. Hence, $f$ selects a string in $L'$ if $\|\{y_1, \ldots, y_n\} \cap L'\| \geq 2$.

**Case 2:** The input strings have different lengths. Let $\ell \stackrel{\mathrm{df}}{=} \max\{|y_1|, \ldots, |y_n|\}$. By brute force, we can decide in time polynomial in $\ell$ if there is some string with length smaller than $\ell$ in $L'$. If so, $f$ selects the first string found. Otherwise, by the argument of Case 1, we can show that $f$ selects a string (of maximum length) in $L'$ if $L'$ contains two of the inputs.

2. Let $\{M_i\}_{i \geq 1}$ be an enumeration of all deterministic polynomial-time Turing machines. Define

$$A \stackrel{\mathrm{df}}{=} \{0^{e(i)} \mid i \geq 1 \wedge 0^{e(i)} \notin L(M_i)\} \cup \{1^{e(i)} \mid i \geq 1 \wedge 0^{e(i)} \in L(M_i)\}.$$

Assume $A \in \mathrm{P}$ via $M_j$ for some $j \geq 1$. This contradicts that $0^{e(j)} \in A$ if and only if $0^{e(j)} \notin L(M_j)$. Hence, $A \notin \mathrm{P}$. Define an FP function $g$ by $g(0^{e(i)}) \stackrel{\mathrm{df}}{=} 1^{e(i)}$ and $g(1^{e(i)}) \stackrel{\mathrm{df}}{=} 0^{e(i)}$ for any $i \geq 1$, and for any $x \notin \{0^{e(i)}, 1^{e(i)}\}$, define $g(x) \stackrel{\mathrm{df}}{=} y$, where $y$ is a fixed string in $A$ (w.l.o.g., $A \neq \emptyset$). Clearly, $A \leq_m^p \overline{A}$ via $g$. $A \in S(2)$ follows as above. $\square$

**Definition 5.2.8** For sets $A$ and $B$, $A \leq_{m,\ell i}^p B$ if there is an FP function $f$ such that for all $x \in \Sigma^*$, (a) $x \in A \iff f(x) \in B$, and (b) $x <_{\mathrm{lex}} f(x)$.

Note that a similar kind of reduction was defined and was of use in [HHSY91], and that, intuitively, sets in $\{L \mid L \leq_{m,\ell i}^p L\}$ may be viewed as having a very weak type of padding functions.

**Theorem 5.2.9** If $L \in \mathrm{SH}$ and $L \leq_{m,\ell i}^p L$, then $L \in$ P-Sel.

**Proof.** Let $L \leq_{m,\ell i}^p L$ via $f$, and let $g$ be an $S(k)$-selector for $L$, for some $k$ for which $L \in S(k)$. A P-selector $h$ for $L$ is defined as follows: Given any inputs $x$ and $y$, generate two chains of $k$ lexicographically increasing strings by running the reduction $f$, i.e., $x = x_1 <_{\mathrm{lex}} x_2 <_{\mathrm{lex}} \cdots <_{\mathrm{lex}} x_k$ and $y = y_1 <_{\mathrm{lex}} y_2 <_{\mathrm{lex}} \cdots <_{\mathrm{lex}} y_k$, where $x_2 = f(x)$, $x_3 = f(f(x))$, etc., and similarly for the $y_i$. To ensure that $g$ will run on *distinct* inputs only (otherwise, $g$ is not obliged to meet requirements 1 and 2 of Definition 5.2.1), let $z_1, \ldots, z_l$ be all the $y_i$'s not in $\{x_1, \ldots, x_k\}$. Now run $g(x_1, \ldots, x_k, z_1, \ldots, z_l)$ and define $h(x, y)$ to output $x$ if $g$ outputs some string $x_i$, and to output $y$ if $g$ selects some string $y_i$ (recall our assumption that $S(k)$-selectors such as $g$ output exactly one string). Clearly, $h \in \mathrm{FP}$, and if $x$ or $y$ are in $L$, then at least $k$ inputs to $g$ are in $L$, so $h$ selects a string in $L$. $\square$

Ogihara [Ogi94] has recently introduced the polynomial-time membership comparable sets (see Definition 5.2.10 below) as another generalization of the P-selective sets. Since P-mc$(k)$ is closed under $\leq_{1\text{-tt}}^p$-reductions for each $k$ [Ogi94] but none of the $S(k)$ for $k \geq 2$ is closed under $\leq_m^p$-reductions (Theorem 5.2.6), it is clear that Ogihara's approach to generalized selectivity is different from ours, and in Theorem 5.2.12, we completely establish, in terms of incomparability and strict inclusion, the relations between his and our generalized selectivity classes (see Figure 5.2).

**Definition 5.2.10** [Ogi94] Let $g$ be a monotone non-decreasing and polynomially bounded FP function from $\mathbb{N}$ to $\mathbb{N}^+$.

1. A function $f$ is called a $g$-membership comparing function (a $g$-mc-function, for short) for $A$ if for every $x_1, \ldots, x_m$ with $m \geq g(\max\{|x_1|, \ldots, |x_m|\})$,
$$f(x_1, \ldots, x_m) \in \{0,1\}^m \quad \text{and} \quad (\chi_A(x_1), \ldots, \chi_A(x_m)) \neq f(x_1, \ldots, x_m).$$
2. A set $A$ is polynomial-time $g$-membership comparable if there exists a polynomial-time computable $g$-mc-function for $A$.
3. P-mc($g$) denotes the class of all polynomial-time $g$-membership comparable sets.
4. P-mc$(const) \stackrel{\text{df}}{=} \bigcup\{\text{P-mc}(k) \mid k \geq 1\}$, P-mc$(log) \stackrel{\text{df}}{=} \bigcup\{\text{P-mc}(f) \mid f \in \mathcal{O}(\log)\}$, and P-mc$(poly) \stackrel{\text{df}}{=} \bigcup\{\text{P-mc}(p) \mid p \in \mathbb{P}\text{ol}\}$.

**Remark 5.2.11** We can equivalently (i.e., without changing the class) require in the definition that $f(x_1, \ldots, x_m) \neq (\chi_A(x_1), \ldots, \chi_A(x_m))$ must hold only if the inputs $x_1, \ldots, x_m$ happen to be *distinct*. This is true because if there are $r$ and $t$ with $r \neq t$ and $x_r = x_t$, then $f$ simply outputs a length $m$ string having a "0" at position $r$ and a "1" at position $t$.

**Theorem 5.2.12** 1. P-mc$(2) \not\subseteq$ fair-S$(n-1, 1)$.

2. For any $k \geq 1$, S$(k) \subset$ P-mc$(k+1)$ and S$(k) \not\subseteq$ P-mc$(k)$.[2]
3. S$(n-1, 1) \subset$ P-mc$(2)$.
4. fair-S$(n-1, 1) \subset$ P-mc$(n)$ and fair-S$(n-1, 1) \not\subseteq$ P-mc$(n-1)$.

**Proof.** First recall that $\{f_i\}_{i \geq 1}$ is our enumeration of FP functions and that the set $W_{i,s} = \{w_{i,1}, \ldots, w_{i,s}\}$ collects the lexicographically smallest $s$ ($s \leq 2^{e(i)}$) strings in $\Sigma^{e(i)}$, where function $e$ is inductively defined to be $e(0) = 2$ and $e(i) = 2^{e(i-1)}$ for $i \geq 1$. Recall also our assumption that a selector for a set in SH outputs a single input string (if the promise is met), whereas S$(n-1, 1)$ and fair-S$(n-1, 1)$ are defined via selectors which output subsets of the given set of inputs.

1. We will construct a set $A$ in stages. Let $u_i$ be the smallest string in $W_{i,e(i)} \cap f_i(W_{i,e(i)})$ (if this set is non-empty; otherwise, $f_i$ immediately disqualifies for being a fair-S$(n-1, 1)$-selector and we may go to the next stage). Define $A \stackrel{\text{df}}{=} \bigcup_{i \geq 1}(W_{i,e(i)} - \{u_i\})$. Then,

[2] This generalizes to $k$ larger than 1 a result of Ogihara who proves that the P-selective sets are strictly contained in P-mc$(2)$ [Ogi94] as well as the known fact that P-Sel is strictly larger than P [Sel79].

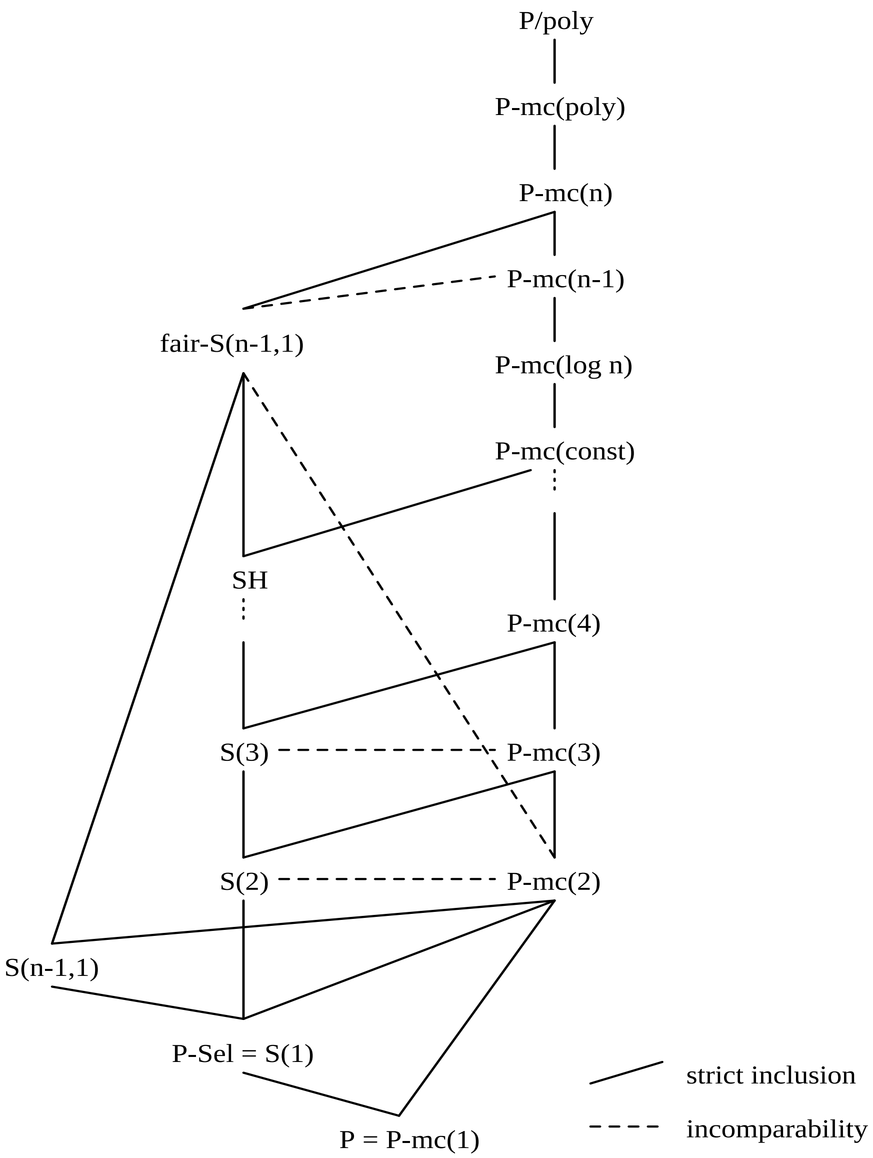


Figure 5.2: Inclusion relationships among S, fair-S, and P-mc classes.

$A \notin$ fair-S$(n-1,1)$, since for any $i$, $f_i(W_{i,e(i)})$ outputs a string not in $A$ although $e(i)-1$ of these inputs (each of length $e(i)$, i.e., the inputs satisfy the "fair condition") *are* in $A$.

For defining a P-mc(2) function $g$ for $A$, let any distinct inputs $x_1, \ldots, x_m$ with $m \geq 2$ be given. If there is some $x_j$ such that $x_j \notin W_{i,e(i)}$ for any $i$, then define $g(x_1, \ldots, x_m)$ to be $0^{j-1}10^{m-j}$. If there is some $x_j$ with $|x_j| < e(i_0)$, where $e(i_0) = \max\{|x_1|, \ldots, |x_m|\}$, then compute the bit $\chi_{\overline{A}}(x_j)$ by brute force in time polynomial in $e(i_0)$, and define $g(x_1, \ldots, x_m)$ to be $0^{j-1}\chi_{\overline{A}}(x_j)0^{m-j}$. Otherwise (i.e., if $\{x_1, \ldots, x_m\} \subseteq W_{i_0,e(i_0)}$), let $g(x_1, \ldots, x_m)$ be $0^m$. Since, by definition of $A$, there is at most one string in $W_{i_0,e(i_0)}$ that is not in $A$, but $m \geq 2$, we have $g(x_1, \ldots, x_m) \neq (\chi_A(x_1), \ldots, \chi_A(x_m))$. Thus, $A \in$ P-mc(2) via $g$.

2. For fixed $k \geq 1$, let $L \in S(k)$ via $f$. Define a P-mc$(k+1)$ function $g$ for $L$ that, given distinct inputs $x_1, \ldots, x_m$ with $m \geq k+1$, outputs the string $1^{j-1}01^{m-j}$ if $x_j$ is the string output by $f(x_1, \ldots, x_m)$. Clearly, $g(x_1, \ldots, x_m) \neq (\chi_L(x_1), \ldots, \chi_L(x_m))$, since there are at least $k$ 1's in $1^{j-1}01^{m-j}$, and $f(x_1, \ldots, x_m) = x_j$ is thus a string in $L$. Hence, $L \in$ P-mc$(k+1)$ via $g$, showing $S(k) \subseteq$ P-mc$(k+1)$. By Statement 1, this inclusion is strict, and so is any inclusion to be proven below.

To show that $S(k) \not\subseteq$ P-mc$(k)$, fix $k$ strings $b_1, \ldots, b_k$ of the same length. Define

$$A \stackrel{\mathrm{df}}{=} \left\{ b_j^{e(i)} \;\middle|\; \begin{array}{l} i \geq 1 \text{ and } f_i(b_1^{e(i)}, \ldots, b_k^{e(i)}) \in \{0,1\}^k \\ \text{and has a “1” at position } j,\ 1 \leq j \leq k \end{array} \right\}.$$

Clearly, since $f_i(b_1^{e(i)}, \ldots, b_k^{e(i)}) = (\chi_A(b_1^{e(i)}), \ldots, \chi_A(b_k^{e(i)}))$ for any $i$, no FP function $f_i$ can serve as a P-mc$(k)$ function for $A$. To define an $S(k)$-selector for $A$, let any inputs $y_1, \ldots, y_m$ (w.l.o.g., each of the form $b_j^{e(i)}$) be given, and let $\ell = \max\{|y_1|, \ldots, |y_m|\}$. As in the proofs of Theorem 5.2.5 and Theorem 5.2.6, it can be decided in time polynomial in $\ell$ whether there is some string of length smaller than $\ell$ in $A$. If so, the $S(k)$-selector $f$ for $A$ selects the first such string found. Otherwise, $f$ outputs an arbitrary string of maximum length. Since there are at most $k$ strings in $A$ at any length, either the output string is in $A$, or $\|A \cap \{y_1, \ldots, y_m\}\| < k$. Thus, $S(k) \not\subseteq$ P-mc$(k)$. Statement 1 implies that as well P-mc$(k) \not\subseteq S(k)$ for $k \geq 2$; the $k$th level of SH $= \bigcup_{i \geq 1} S(i)$ and of the hierarchy within P-mc$(const)$ are thus incomparable.

3. Let $L \in S(n-1,1)$ via selector $f$. Define a P-mc(2) function $g$ for $L$ as follows: Given distinct input strings $x_1, \ldots, x_n$ with $n \geq 2$, $g$ simulates $f(x_1, \ldots, x_n)$ and outputs the string $1^{j-1}01^{n-j}$ if $x_j$ is any (say the smallest) string in $f(x_1, \ldots, x_n)$. Again, we can

exclude one possibility for $(\chi_A(x_1), \ldots, \chi_A(x_n))$ via $g$ in polynomial time, because the $S(n-1,1)$-promise is met for the string $1^{j-1}01^{n-j}$, and thus $f$ must output a string in $L$.

4. Now we show that the proof of Statement 3 fails to some extent for the corresponding fair-class, i.e., we will show that fair-$S(n-1,1) \not\subseteq$ P-mc$(n-1)$.[3] $A \stackrel{\text{df}}{=} \bigcup_{i\geq 1} A_i$ is defined in stages so that in stage $i$, $f_i$ fails to be a P-mc$(n-1)$ function for $A_i$. This is ensured by defining $A_i$ as a subset of the $e(i)-1$ smallest strings of length $e(i)$, $W_{i,e(i)-1}$, such that $w_{i,j} \in A_i$ if and only if $f_i(W_{i,e(i)-1})$ outputs a string of length $e(i)-1$ and has a "1" at position $j$. Thus, $A \notin$ P-mc$(n-1)$, since $f_i(w_{i,1}, \ldots, w_{i,e(i)-1}) = (\chi_A(w_{i,1}), \ldots, \chi_A(w_{i,e(i)-1}))$ for any $i \geq 1$.

To see that $A \in$ fair-$S(n-1,1)$, let any distinct inputs $y_1, \ldots, y_n$ be given, each having, w.l.o.g., length $e(i)$ for some $i$, and let $e(i_0)$ be their maximum length. As before, if there exists a string of length smaller than $e(i_0)$, say $y_j$, then it can be decided by brute force in polynomial time whether or not $y_j$ belongs to $A$. Define a fair-$S(n-1,1)$-selecto $g$ to output $\{y_j\}$ if $y_j \in A$, and to output any input different from $y_j$ if $y_j \notin A$. Thus, either the string output by $g$ does belong to $A$, or $\|A \cap \{y_1, \ldots, y_n\}\| < n-1$. On the other hand, if all input strings are of the same length $e(i_0)$ and $\{y_1, \ldots, y_n\} \subseteq W_{i_0,e(i_0)-1}$, then the "fair condition" is not fulfilled, as $e(i_0) > n$, and $g$ is thus not obliged to output a string in $A$. If all inputs have length $e(i_0)$ and $\{y_1, \ldots, y_n\} \not\subseteq W_{i_0,e(i_0)-1}$, then by the above argument, $g$ can be defined such that either the string output by $g$ does belong to $A$, or $\|A \cap \{y_1, \ldots, y_n\}\| < n-1$. This completes the proof of $A \in$ fair-$S(n-1,1)$.

Finally, we show that fair-$S(n-1,1) \subseteq$ P-mc$(n)$. Let $L \in$ fair-$S(n-1,1)$ via selector $f$. Let $y_1, \ldots, y_n$ be any distinct input strings such that $n \geq \max\{|y_1|, \ldots, |y_n|\}$, i.e., the "fair condition" is now satisfied. Define a P-mc-function $g$ for $L$ which, on inputs $y_1, \ldots, y_n$, simulates $f(y_1, \ldots, y_n)$ and outputs the string $1^{j-1}01^{n-j}$ if $f$ selects $y_j$. Thus,

$$g(y_1, \ldots, y_n) \neq (\chi_L(y_1), \ldots, \chi_L(y_n)),$$

and we have $L \in$ P-mc$(n)$ via $g$. □

[3] This is similar as in Part 2 although the proof now rests also on the "fair condition" rather than merely on the $(n-1)$-promise. However, this "fair condition" can no longer "protect" fair-$S(n-1,1)$ from being contained in P-mc$(n)$.

### 5.2.2 Circuit, Lowness, and Collapse Results

This section demonstrates that the core results (i.e., small circuit, $\mathrm{Low}_2$-ness, and collapse results) holding for the P-selective sets, and proving them structurally simply, also hold for our generalized selectivity classes.

Since P-mc$(poly) \subseteq$ P/poly [Ogi94] and fair-S$(n-1,1)$ is (strictly) contained in P-mc$(n)$, it follows immediately that every set in fair-S$(n-1,1)$ has polynomial-size circuits and is thus in $\mathrm{EL}\Theta_3$ (by Köbler's result that P/poly $\subseteq \mathrm{EL}\Theta_3$ [Köb94]). Note that Ogihara refers to Amir, Beigel, and Gasarch, whose P/poly proof for "non-p-superterse" sets (see [ABG90, Theorem 10]) applies to Ogihara's class P-mc$(poly)$ as well. On the other hand, P-selective NP sets can even be shown to be in $\mathrm{Low}_2$ [KS85], the second level of the low hierarchy within NP. In contrast, the proof of [ABG90, Theorem 10] does not give a $\mathrm{Low}_2$-ness result for non-p-superterse NP sets, and thus also does not provide such a result for P-mc$(poly) \cap$ NP. By modifying the technique of Ko and Schöning, however, we generalize in Theorem 5.2.16 their result to our larger selectivity classes.[4] The proof of Theorem 5.2.16 explicitly constructs a family of non-uniform advice sets for any set in fair-S$(n-1,1)$, as merely stating the existence of those advice sets (which follows from Theorem 5.2.13) does not suffice for proving $\mathrm{Low}_2$-ness.

Note that some results of this section (e.g., Theorem 5.2.13) extend to the more general GC classes that will be defined in Section 5.4. We propose as an interesting task to explore whether all results of this section, in particular the $\mathrm{Low}_2$-ness result of Theorem 5.2.16, apply to the GC classes.

**Theorem 5.2.13** fair-S$(n-1,1) \subseteq$ P/poly.

**Corollary 5.2.14** SH $\subseteq$ P/poly.

**Corollary 5.2.15** fair-S$(n-1,1) \subseteq \mathrm{EL}\Theta_3$.

**Theorem 5.2.16** Any set in NP $\cap$ fair-S$(n-1,1)$ is $\mathrm{Low}_2$.

[4] Very recently, our generalization of Ko and Schöning's result that P-Sel $\cap$ NP $\subseteq \mathrm{Low}_2$ (and also other researchers' modifications or generalizations of their result such as "Any P-cheatable NP set is $\mathrm{Low}_2$" [ABG90], or "Any NPSV-selective NP set is $\mathrm{Low}_2$" [HNOS94]) has been further extended by Köbler [Köb95]. The most general currently known $\mathrm{Low}_2$-ness result for NP sets having selector functions (in any selectivity concept that has been considered in the literature) is stated in Köbler's paper as follows: "Any NP set that is *strongly* membership comparable by NPSV functions is $\mathrm{Low}_2$" [Köb95]. We refer to [Köb95, ABG90, HNOS94] for the notations not defined in this footnote.

**Proof.** Let $L$ be any NP set in fair-S$(n-1,1)$, and let $f$ be a selector for $L$ and $N$ be an NPM such that $L = L(N)$. First, for each length $m$, we shall construct a polynomially length-bounded advice $A_m$ that helps deciding membership of any string $x$, $|x| = m$, in $L$ in polynomial time. For $m < 4$, take $A_m \stackrel{\text{df}}{=} L^{=m}$ as advice. From now on let $m \geq 4$ be fixed, and let $n$ be such that $4 \leq 2n \leq m$.

Some notations are in order. A subset $G$ of $L^{=m}$ is called a *game* if $\|G\| = n$. Any output $w \in f(G)$ is called a *winner* of game $G$, and is said to be *yielded* by the *team* $G - \{w\}$. If $\|L^{=m}\| \leq 2(n+1)$, then simply take $A_m \stackrel{\text{df}}{=} L^{=m}$ as advice. Otherwise, $A_m$ is constructed in *rounds*. In round $i$, one team, $t_i$, is added to $A_m$, and all winners yielded by that team in any game are deleted from a set $B_{i-1}$. Initially, $B_0$ is set to be $L^{=m}$.

In more detail, in the first round, all games of $B_0 = L^{=m}$, one after the other, are fed into the selector $f$ for $L$ to determine all winners of each game, and, associated with each winner, the team yielding that winner. We will argue below that there must exist at least one team yielding at least $\frac{\binom{N}{n}}{\binom{N}{n-1}}$ winners if $N$ is the number of strings in $L^{=m}$. Choose the "smallest" (according to the ordering $\leq_{\text{lex}}$ on $L^{=m}$) such team, $t_1$, and add it to the advice $A_m$. Delete from $B_0$ all winners yielded by $t_1$ and set $B_1$ to be the remainder of $B_0$, i.e.,

$$B_1 \stackrel{\text{df}}{=} B_0 - \{w \mid \text{winner } w \text{ is yielded by team } t_1\},$$

and, entering the second round, repeat this procedure with all games of $B_1$ unless $B_1$ has $\leq 2(n+1)$ elements. In the second round, a second team $t_2$, and in later rounds, more teams $t_i$ are determined and are added to $A_m$. The construction of $A_m$ in rounds will terminate if $\|B_{k(m)}\| \leq 2(n+1)$ for some integer $k(m)$ depending on the given length $m$. In that case, add $B_{k(m)}$ to $A_m$. Formally, $A_m \stackrel{\text{df}}{=} \left(\bigcup_{i=1}^{k(m)} t_i\right) \cup B_{k(m)}$, where $B_{k(m)} \subseteq L^{=m}$ contains at most $2(n+1)$ elements, $t_i \subseteq L^{=m}$ is the team added to $A_m$ in round $i$, $1 \leq i \leq k(m)$, and the bound $k(m)$ on the number of rounds executed at length $m$ is specified below.

We now show that there is some polynomial in $m$ bounding the length of (the coding of) $A_m$ for any $m$. If $L^{=m}$ has $N > 2(n+1)$ strings, then there are $\binom{N}{n}$ games and $\binom{N}{n-1}$ teams in the first round. Since every game has at least one winner, there exists one team yielding at least

$$\frac{\binom{N}{n}}{\binom{N}{n-1}} = \frac{N-n+1}{n} > \frac{N}{2n} \geq \frac{N}{m}$$

winners to be deleted from $B_0$ in the first round. Thus, there remain in $B_1$ at most $N\left(1 - \frac{1}{m}\right)$

elements after the first round, and, successively applying this argument, $B_k$ contains at most $N\left(1-\frac{1}{m}\right)^k$ elements after $k$ rounds. Since $N \leq 2^m$ and the procedure terminates if $\|B_k\| \leq 2(n+1)$ for some integer $k$, it suffices to show that some polynomial $k(m)$ of fixed degree satisfies $\left(1-\frac{1}{m}\right)^{k(m)} \leq 2(n+1)2^{-m}$. This follows from the fact that $\lim_{m\to\infty}\left(\left(1-\frac{1}{m}\right)^{m^2}\right)^{m^{-1}} = e^{-1} < \frac{1}{2}$ implies that $\left(1-\frac{1}{m}\right)^{m^2}$ is in $\mathcal{O}(2^{-m})$. As in each round $n-1 < m$ strings of length $m$ are added to $A_m$, the length of (the coding of) $A_m$ is indeed bounded above by some polynomial of degree 4.

Note that the set

$$C \stackrel{\text{df}}{=} \left\{ \langle x, a_{|x|}\rangle \;\middle|\; \begin{array}{l} a_{|x|} \text{ is encoding of an advice } A_{|x|} \text{ and } x \in B_{k(|x|)}, \text{ or } (\exists t_j) \\ {[t_j \text{ is a team of } A_{|x|} \text{ and } x \text{ belongs to or is yielded by } t_j]} \end{array} \right\}$$

witnesses $L \in$ P/poly (Theorem 5.2.13), as clearly $C \in$ P and $L = \{x \mid \langle x, a_{|x|}\rangle \in C\}$.

Now we are ready to prove $L \in \mathrm{Low}_2$. Let $D \in \mathrm{NP}^{\mathrm{NP}^L}$ be witnessed by some NPOMs $N_1$ and $N_2$, that is, $D = L(N_1^{L(N_2^L)})$. Let $q(\ell)$ be a polynomial bound on the length of all queries that can be asked in this computation on an input of length $\ell$. We describe below an NPOM $M$ and an NP oracle set $E$ for which $D = L(M^E)$.

On input $x$, $M$ guesses for each length $m$, $1 \leq m \leq q(|x|)$, all possible polynomially length-bounded advice sets $A_m$ for $L^{=m}$, simultaneously guessing witnesses (that is, an accepting path of $N$ on input $z$) that each string $z$ in any guessed advice set is in $L^{=m}$. To check on each path whether the guessed sequence of advice sets is correct, $M$ queries its oracle $E$ whether it contains the string $\langle x, A_1, \ldots, A_{q(|x|)}\rangle$, where

$$E \stackrel{\text{df}}{=} \left\{ \langle x, A_1, \ldots, A_{q(|x|)}\rangle \;\middle|\; \begin{array}{l} (\exists m : 1 \leq m \leq q(|x|))\,(\exists y_m : |y_m| = m)\,(\exists w_m)\,[w_m \\ \text{is an accepting path of } N(y_m), \text{ and yet } y_m \text{ is neither a} \\ \text{string in } A_m \text{ nor is yielded by any team of } A_m] \end{array} \right\}$$

is clearly a set in NP. If the answer is "yes," then some guessed advice is incorrect, and $M$ rejects on that computation. If the answer is "no," then each guessed advice is correct for any possible query of the respective length. Thus, $M$ now can simulate the computation of $N_1^{L(N_2)}$ on input $x$ using the selector $f$ and the relevant advice $A_m$ to answer any question of $N_2$ correctly. Hence, $D \in \mathrm{NP}^{\mathrm{NP}}$. □

Ogihara has shown that if NP $\subseteq$ P-mc$(c \log n)$ for some $c < 1$, then P = NP [Ogi94].[5]

[5] In [Ogi94], this result is also established for certain complexity classes other than NP. In this thesis, we focus on the NP case only, however.

Since by the proof of Theorem 5.2.12, fair-S$(c \log n, 1) \subseteq$ P-mc$(c \log n)$, $c < 1$, we have immediately the following corollary to Ogihara's result.

**Corollary 5.2.17** If NP $\subseteq$ fair-S$(c \log n, 1)$ for some $c < 1$, then P $=$ NP.

## 5.3 Extended Lowness and the Join Operator

Essentially, the low hierarchy ([Sch83]; see Part 1 of Definition 2.3.5 on page 12) provides a yardstick to measure the complexity of sets that are known to be in NP but that are seemingly neither in P nor NP-complete.[6] In order to extend this classification beyond NP, the extended low hierarchy ([BBS86b]; see Definition 2.3.5.2 on page 12) has been introduced (see the surveys [Köb95, Hem93]). The intuition is that a set $A$ that is placed in the $k$th level of the low or the extended low hierarchy either contains no more information than the empty set relative to the computation of a $\Sigma_k^p$ oracle machine, or $A$ is so badly organized that a $\Sigma_k^p$ oracle machine is not able to extract useful information from $A$. These two hierarchies have been very thoroughly investigated in, e.g., [Sch83, KS85, BBS86b, Sch88, Sch89, Ko91, AH92, ABG90, Köb94, LS94, HNOS94]. One main motivation in these studies is to locate interesting problems (such as the graph isomorphism problem, which is known to be low) and classes of problems (known extended low classes include BPP, approximate polynomial time, the class of complements of sets having Arthur-Merlin games, the class of sparse and co-sparse sets, the P-selective sets, the class of sets having polynomial-size circuits (i.e., P/poly), etc.) in certain levels of the hierarchies and to prove lower bounds to certify the optimality of the location obtained. Another motivation is to explore and to better understand the structure of the hierarchies themselves and to relate their properties to other complexity-theoretic concepts. For instance, Schöning has shown that the existence of an NP-complete set (under any "reasonable" reducibility) in the low hierarchy implies a collapse of the polynomial hierarchy [Sch83], and Long and Sheu have proven that the extended low hierarchy is an infinite hierarchy [LS94]. This section contributes to this latter type of task.

[6] Very recently, Hemaspaandra, Wechsung, and this author have taken another approach to describe various degrees of "simplicity" of NP sets by studying the classes of NP sets for which all, or some, certificate schemes (i.e., NP machines) accepting the set have always, or have infinitely often, easy certificates (i.e., polynomial-time computable accepting paths) [HRW95].

The following result establishes a structural difference between Selman's P-selectivity and the generalized selectivity introduced here: Though $S(1) = \text{P-Sel} \subseteq \text{EL}_2$, we show that there are sets (indeed, sparse sets) in $S(2)$ that are not in $\text{EL}_2$. Previously, Allender and Hemachandra [AH92] have shown that P/poly (and indeed SPARSE and coSPARSE) is not contained in $\text{EL}_2$. Theorem 5.3.1 and Corollary 5.3.2, however, extend this result and give the first known (and optimal) $\text{EL}_2$ lower bound for generalized selectivity-like classes.

**Theorem 5.3.1** $\text{SPARSE} \cap S(2) \cap \text{P-mc}(2) \not\subseteq \text{EL}_2$.

**Proof.** For $i \geq 1$, define $t(i) \stackrel{\text{df}}{=} 2^{2^{2^{t(i-1)}}}$, where $t(0) \stackrel{\text{df}}{=} 2$, and let $T_k \stackrel{\text{df}}{=} \Sigma^{t(k)}$, for $k \geq 0$, and $T \stackrel{\text{df}}{=} \bigcup_{k \geq 0} T_k$. Let EE be defined as $\bigcup_{c \geq 0} \text{DTIME}[2^{c2^n}]$. We will construct a set $B$ such that (a) $B \subseteq T$, (b) $B \in \text{EE}$, (c) $\|B \cap T_k\| \leq 1$ for each $k \geq 0$, and (d) $B \notin \text{EL}_2$. Note that it follows from (a), (b), and (c) that $B$ is a sparse set in $S(2)$. Indeed, any input to the $S(2)$-selector that is not in $T$ is not in $B$ by (a). If all inputs that are in $T$ are in the same $T_k$ then, by (c), the $S(2)$-promise is never satisfied, and the selector may output an arbitrary input. If the inputs that are in $T$ fall in more than one $T_k$, then for all inputs of length smaller than the maximum length, it can be decided by brute force whether or not they belong to $B$—this is possible, as $B \in \text{EE}$ and the $T_k$ are triple-exponentially spaced. From these comments, the action of the $S(2)$-selector is clear.

Clearly, $B$ also is in $\text{P-mc}(k)$ for each $k \geq 3$ by Theorem 5.2.12. But since $S(2)$ and $\text{P-mc}(2)$ are incomparable, we still must argue that $B \in \text{P-mc}(2)$. Again, this follows from (a), (b), and (c), since for any fixed two inputs, $u$ and $v$, if they are of different lengths, then the smaller one can be solved by brute force; and if they have the same length, then it is impossible by (c) that $(\chi_B(u), \chi_B(v)) = (1, 1)$. In any case, one out of the four possibilities for the membership of $u$ and $v$ in $B$ can be excluded in polynomial time. Hence, $B \in \text{P-mc}(2)$.

For proving (d), we will construct $B$ such that $\text{NP}^B \not\subseteq \text{coNP}^{B \oplus \text{SAT}}$ (which clearly implies that $\text{NP}^{\text{NP}^B} \not\subseteq \text{NP}^{B \oplus \text{SAT}}$). Define

$$L_B \stackrel{\text{df}}{=} \{0^n \mid (\exists x : |x| = n)\, [x \in B]\}.$$

Clearly, $L_B \in \text{NP}^B$. Let $\{N_i\}_{i \geq 1}$ be a standard enumeration of all coNP oracle machines satisfying the condition that the runtime of each $N_i$ is independent of the oracle and each machine is repeated infinitely often in the enumeration. Let $p_i$ be the polynomial bound on the runtime of $N_i$. The set $B \stackrel{\text{df}}{=} \bigcup_{i \geq 0} B_i$ is constructed in stages. In stage $i$, at most one string of length $n_i$ will be added to $B$, and $B_{i-1}$ will have previously been set to the content of $B$ up to stage $i$. Initially, $B_0 = \emptyset$ and $n_0 = 0$. Stage $i > 0$ is as follows: Let $n_i$ be the smallest number such that $n_i > n_{i-1}$, $n_i = t(k)$ for some $k$, and $2^{n_i} > p_i(n_i)$. Simulate $N_i^{B_{i-1} \oplus \mathrm{SAT}}(0^{n_i})$.

**Case 1:** If it rejects (in the sense of coNP, i.e., if it has one or more rejecting computation paths), then fix some rejecting path and let $w_i$ be the smallest string of length $n_i$ that is not queried along this path (note that, by our choice of $n_i$, such a string $w_i$, if needed, must always exist), and set $B_i := B_{i-1} \cup \{w_i\}$.

**Case 2:** If $0^{n_i} \in L(N_i^{B_{i-1} \oplus \mathrm{SAT}})$, then set $B_i := B_{i-1}$.

**Case 3:** If the simulation of $N_i$ on input $0^{n_i}$ fails to be completed in double exponential (say, $2^{100 \cdot 2^{n_i}}$ steps) time (for example, because $N_i$ is huge in size relative to $n_i$), then abort the simulation and set $B_i := B_{i-1}$.

This completes the construction of stage $i$. Since we have chosen an enumeration such that the same machine as $N_i$ appears infinitely often and as the $n_i$ are strictly increasing, it is clear that for only a finite number of the $\{N_j\}_{j \geq 1}$ that are the same machine as $N_i$ can Case 3 occur (and thus $N_i$, either directly or via one of its clones, is diagonalized against eventually). Note that the construction meets requirements (a), (b), and (c) and shows $L_B \neq L(N_i^{B \oplus \mathrm{SAT}})$ for any $i \geq 1$. $\square$

**Corollary 5.3.2** coSPARSE $\cap$ coS$(2) \not\subseteq$ EL$_2$.

**Theorem 5.3.3** EL$_2$ is not closed under intersection, union, exclusive-or, or nxor.

**Proof (Sketch).** We sketch just the idea of the proof. Using the technique of [HJ] (to be applied also in some proofs of Section 5.4), it is not hard to prove that the set $B$ constructed in the above proof can be represented as $B = A_1 \cap A_2$ for P-selective sets $A_1$ and $A_2$. More precisely, let

$$\begin{aligned} A_1 &\stackrel{\text{df}}{=} \{x \mid (\exists w \in B)\, [|x| = |w| \wedge x \leq_{\text{lex}} w]\}, \\ A_2 &\stackrel{\text{df}}{=} \{x \mid (\exists w \in B)\, [|x| = |w| \wedge w \leq_{\text{lex}} x]\}. \end{aligned}$$

Since $B \in EE$ and is triple-exponentially spaced, we have from an argument similar to that in the proof of Lemma 5.4.5 (see [HJ]) that $A_1, A_2 \in$ P-Sel $\subseteq EL_2$. On the other hand, we have seen in the previous proof that $B = A_1 \cap A_2$ is not in $EL_2$. Similarly, if we define

$$C_1 \stackrel{\text{df}}{=} \{x \mid (\exists w \in B)\,[|x| = |w| \wedge x <_{\text{lex}} w]\},$$
$$C_2 \stackrel{\text{df}}{=} \{x \mid (\exists w \in B)\,[|x| = |w| \wedge x \leq_{\text{lex}} w]\},$$

we have $B = C_1 \,\Delta\, C_2$ and $C_1, C_2 \in$ P-Sel $\subseteq EL_2$. Thus, $EL_2$ is not closed under intersection or exclusive-or. Since $EL_2$ is closed under complementation, it must also fail to be closed under union and nxor. □

The proof of the above result also establishes the following corollary.

**Corollary 5.3.4** [HJ] P-Sel is not closed under intersection, union, exclusive-or, or nxor.

Theorem 5.3.5 below establishes that, *in terms of extended lowness, the join operator can lower complexity*. At first glance, this might seem paradoxical. After all, every set that reduces to a set $A$ or $B$ also reduces to $A \oplus B$, and thus, one might think that $A \oplus B$ must be at least as hard as $A$ and $B$, as most complexity lower bounds (e.g., NP-hardness) are defined in terms of reductions. However, extended lowness measures the complexity of a set's internal organization, and thus Theorem 5.3.5 is not paradoxical. Rather, Theorem 5.3.5 highlights the orthogonality of "complexity via reductions" and "complexity via non-extended-lowness." Indeed, note Corollary 5.3.6, which was first observed in [AH92]. Lemma 5.3.8 will be used in the upcoming proof of Theorem 5.3.5.

**Theorem 5.3.5** $(\exists A, B)\,[A \notin EL_2 \wedge B \notin EL_2 \wedge A \oplus B \in EL_2]$.

**Corollary 5.3.6** [AH92] $EL_2$ is not closed under $\leq_m^p$-reductions.

In contrast, every level of the low hierarchy within NP is clearly closed under $\leq_m^p$-reductions. Thus, the low hierarchy analog of Theorem 5.3.5 is false, and even the slightly stronger fact below can be proven.

**Fact 5.3.7** $(\forall k \geq 0)\,(\forall A, B)\,[(A \notin \text{Low}_k \vee B \notin \text{Low}_k) \implies A \oplus B \notin \text{Low}_k]$.

**Proof.** Assume $A \oplus B \in \text{Low}_k$. Since for all sets $A$ and $B$, $A \leq_m^p A \oplus B$ and $B \leq_m^p A \oplus B$, the closure of $\text{Low}_k$ under $\leq_m^p$-reductions implies that both $A$ and $B$ are in $\text{Low}_k$. □

**Lemma 5.3.8** If $F$ is a sparse set and $\mathrm{census}_F \in \mathrm{FP}^{F\oplus \mathrm{SAT}}$, then $F \in \mathrm{EL}_2$.

**Proof.** Let $L \in \mathrm{NP}^{\mathrm{NP}^F}$ via NPOMs $N_1$ and $N_2$, i.e., $L = L(N_1^{L(N_2^F)})$. Let $q(n)$ be a polynomial bounding the length of all queries that can be asked in the run of $N_1^{L(N_2^F)}$ on inputs of length $n$. Below we describe an NPOM $N$ with oracle $F \oplus \mathrm{SAT}$:

On input $x$, $|x| = n$, $N$ first computes $\mathrm{census}_F(0^i)$ for each relevant length $i \leq q(n)$, and then guesses all sparse sets up to length $q(n)$. Knowing the exact census of $F$, $N$ can use the $F$ part of its oracle to verify whether the guess for $F^{\leq q(n)}$ is correct, and rejects on all incorrect paths. On the correct path, $N$ uses itself, the SAT part of its oracle, and the correctly guessed set $F^{\leq q(n)}$ to simulate the computation of $N_1^{L(N_2^F)}$ on input $x$.

Clearly, $L(N^{F\oplus \mathrm{SAT}}) = L$. Thus, $\mathrm{NP}^{\mathrm{NP}^F} \subseteq \mathrm{NP}^{F\oplus\mathrm{SAT}}$, i.e., $F \in \mathrm{EL}_2$. $\square$

**Proof of Theorem 5.3.5.** $A \stackrel{\mathrm{df}}{=} \bigcup_{i\geq 0} A_i$ and $B \stackrel{\mathrm{df}}{=} \bigcup_{i\geq 0} B_i$ are constructed in stages. In order to show $A \notin \mathrm{EL}_2$ and $B \notin \mathrm{EL}_2$ it suffices to ensure in the construction that $\mathrm{NP}^A \not\subseteq \mathrm{coNP}^{A\oplus\mathrm{SAT}}$ and $\mathrm{NP}^B \not\subseteq \mathrm{coNP}^{B\oplus\mathrm{SAT}}$. As in the proof of Theorem 5.3.1, define function $t$ inductively by $t(0) \stackrel{\mathrm{df}}{=} 2$ and $t(i) \stackrel{\mathrm{df}}{=} 2^{2^{2^{t(i-1)}}}$ for $i \geq 1$, and let $\{N_i\}_{i\geq 1}$ be our enumeration of all coNP oracle machines having the property that the runtime of each $N_i$ is independent of the oracle and each machine appears infinitely often in the enumeration. Define

$$L_A \stackrel{\mathrm{df}}{=} \{0^{t(i)} \mid (\exists j \geq 1)\, [i = \langle 0, j\rangle \ \wedge\ \|A \cap \Sigma^{t(i)}\| \geq 1]\},$$

$$L_B \stackrel{\mathrm{df}}{=} \{0^{t(i)} \mid (\exists j \geq 1)\, [i = \langle 1, j\rangle \ \wedge\ \|B \cap \Sigma^{t(i)}\| \geq 1]\}.$$

Clearly, $L_A \in \mathrm{NP}^A$ and $L_B \in \mathrm{NP}^B$. In stage $i$ of the construction, at most one string of length $t(i)$ will be added to $A$ and at most one string of length $t(i)$ will be added to $B$ to

**(1)** ensure $L(N_j^{A_i\oplus\mathrm{SAT}}) \neq L_A$ if $i = \langle 0, j\rangle$ (or $L(N_j^{B_i\oplus\mathrm{SAT}}) \neq L_B$ if $i = \langle 1, j\rangle$), and to

**(2)** encode an easy to find string into $A$ if $i = \langle 1, j\rangle$ (or into $B$ if $i = \langle 0, j\rangle$) indicating whether or not some string has been added to $B$ (or to $A$) in (1).

Let $A_{i-1}$ and $B_{i-1}$ be the content of $A$ and $B$ prior to stage $i$. Initially, let $A_0 = B_0 = \emptyset$.

Stage $i$ is as follows: First assume $i = \langle 0, j\rangle$ for some $j \geq 1$. If it is the case that no path of $N_j^{A_{i-1}\oplus\mathrm{SAT}}(0^{t(i)})$ can query all strings in $\Sigma^{t(i)} - \{0^{t(i)}\}$ and $N_j^{A_{i-1}\oplus\mathrm{SAT}}(0^{t(i)})$ cannot query any string of length $t(i+1)$ (otherwise, just skip this stage—we will argue later that the diagonalization still works properly), then simulate $N_j^{A_{i-1}\oplus\mathrm{SAT}}$ on input $0^{t(i)}$. If it rejects (in the sense of coNP, i.e., if it has one or more rejecting computation paths), then fix some rejecting path and let $w_i$ be the smallest string in $\Sigma^{t(i)} - \{0^{t(i)}\}$ that is not queried along this path, and set $A_i := A_{i-1} \cup \{w_i\}$ and $B_i := B_{i-1} \cup \{0^{t(i)}\}$. Otherwise (i.e., if $0^{t(i)} \in L(N_j^{A_{i-1}\oplus\mathrm{SAT}})$), then set $A_i := A_{i-1}$ and $B_i := B_{i-1}$. The case of $i = \langle 1, j\rangle$ is analogous: just exchange $A$ and $B$. This completes the construction of stage $i$.

Since each machine $N_i$ appears infinitely often in our enumeration and as the $t(i)$ are strictly increasing, it is clear that for only a finite number of the $N_{i_1}, N_{i_2}, \ldots$ that are the same machine as $N_i$ can it happen that stage $i_k$ must be skipped (in order to ensure that $w_{i_k}$, if needed to diagonalize against $N_{i_k}$, indeed exists, or that the construction stages do not interfere with each other), and thus each machine $N_i$ is diagonalized against eventually. This proves that $A \notin \mathrm{EL}_2$ and $B \notin \mathrm{EL}_2$. Now observe that $A \oplus B$ is sparse and that $\mathit{census}_{A\oplus B} \in \mathrm{FP}^{A\oplus B}$. Indeed,

$$\mathit{census}_{A\oplus B}(0^n) = 2(\|A \cap \{0, 00, \ldots, 0^{n-1}\}\| + \|B \cap \{0, 00, \ldots, 0^{n-1}\}\|)$$

Thus, by Lemma 5.3.8, $A \oplus B \in \mathrm{EL}_2$. □

One of the most interesting open questions related to the topic of this section is whether the join operator also can *raise* complexity in terms of extended lowness (that is, whether there exist sets $A$ and $B$ such that $A \in \mathrm{EL}_k$ and $B \in \mathrm{EL}_k$, and yet $A \oplus B \notin \mathrm{EL}_k$ for, e.g., $k = 2$), or whether the second level of the extended low hierarchy is (and more generally, whether *all* levels of the hierarchy are) closed under join.

## 5.4 An Extended Selectivity Hierarchy Capturing Boolean Closures of P-selective Sets

Hemaspaandra and Jiang [HJ] noted that the class P-Sel is closed under exactly those Boolean connectives that are either completely degenerate or almost-completely degenerate. In particular, P-Sel is not closed under intersection or union, and is not even closed under marked union (join). This raises the question of how complex, e.g., the intersection of two P-selective sets is. Also, is the class of unions of two P-selective sets more or less complex than the class of intersections of two P-selective sets? Theorem 5.4.7 establishes that, in terms of P-mc classes, unions and intersections of sets in P-Sel are indistinguishable (though they both *are* different from exclusive-or). However, we will note as Theorem 5.4.8 that the GC hierarchy (defined below) does distinguish between these classes, thus capturing the closures of P-Sel under certain Boolean connectives more tightly.

**Definition 5.4.1** Let $g_1$, $g_2$, and $g_3$ be threshold functions. Define GC$(g_1(n), g_2(n), g_3(n))$ to be the class of all sets $L$ for which there exists a polynomial-time computable function $f$ such that for each $n \geq 1$ and any distinct input strings $y_1, \ldots, y_n$,

1. $f(y_1, \ldots, y_n) \subseteq \{y_1, \ldots, y_n\}$ and $\|f(y_1, \ldots, y_n)\| \leq g_2(n)$, and
2. $\|L \cap \{y_1, \ldots, y_n\}\| \geq g_1(n) \implies \|L \cap f(y_1, \ldots, y_n)\| \geq g_3(n)$.

**Remark 5.4.2** For constant thresholds $b$, $c$, $d$, we can equivalently (i.e., without changing the class) require in the definition that the selector $f$ for a set $L \in$ GC$(b, c, d)$ on all input sets of size at least $c$ must output *exactly* $c$ strings. This is true because if $f$ outputs fewer than $c$ strings, we can define a new selector $f'$ that outputs all strings output by $f$ and additionally $\|f\| - c$ arbitrary input strings not output by $f$, and $f'$ is still a GC$(b, c, d)$-selector for L. This will be useful in the proof of Lemma 5.4.13.

The GC classes generalize the S classes of Section 5.2, and as before, we also consider fair-GC classes by additionally requiring the "fair condition." Let GCH denote $\bigcup_{i,j,k \geq 1}$ GC$(i, j, k)$. The internal structure of GCH will be analyzed in Theorem 5.4.14 on page 87. First we note below that the largest nontrivial GC class, [7] fair-GC$(\lfloor \frac{n}{2} \rfloor, \lfloor \frac{n}{2} \rfloor, 1)$, and thus all of GCH, is contained in the P-mc hierarchy.

**Theorem 5.4.3** fair-GC$(\lfloor \frac{n}{2} \rfloor, \lfloor \frac{n}{2} \rfloor, 1) \subseteq$ P-mc$(\mathrm{poly})$.

**Proof.** Let $L \in$ fair-GC$(\lfloor \frac{n}{2} \rfloor, \lfloor \frac{n}{2} \rfloor, 1)$ via selector $f$. Fix any distinct inputs $x_1, \ldots, x_n$ such that $n \geq (\max\{|x_1|, \ldots, |x_n|\})^2$. Define a P-mc$(n^2)$ function $g$ as follows: $g$ simulates $f(x_1, \ldots, x_n)$ and outputs a 0 at each position corresponding to an output string of $f$, and outputs a "1" anywhere else. Note that if all the strings having a "1" in the output of $g$ indeed are in $L$, then so must be at least one of the outputs of $f$, as the "fair condition" is met and $\|\{x_1, \ldots, x_n\} \cap L\| \geq \frac{n}{2}$. Thus, $(\chi_L(x_1), \ldots, \chi_L(x_n)) \neq g(x_1, \ldots, x_n)$, and we have $L \in$ P-mc$(\mathrm{poly})$ via $g$. □

[7] In this chapter, the term "nontrivial" has a different meaning than in Chapter 3. Here, any class $\mathcal{C} \subseteq \mathfrak{P}(\Sigma^*)$ of sets is said to be *nontrivial* if $\mathcal{C}$ contains infinite sets, but not all sets of strings over $\Sigma$. For example, the class fair-GC$(\lceil \frac{n}{2} \rceil, \lceil \frac{n}{2} \rceil, 1)$ equals $\mathfrak{P}(\Sigma^*)$ if $n$ is odd, and is therefore called trivial.

**Lemma 5.4.4** [BvHT93] Let $A \in$ P-Sel and $V \subseteq \Sigma^*$. The P-selector $f$ for $A$ induces a total order $\preceq_f$ on $V$ such that $(\forall x, y \in V)\,[x \preceq_f y \iff (x \in A \implies y \in A)]$.[8]

The following lemma (proven in [HJ]) will be useful in some diagonalization proofs of this section. As in [HJ], define $\mu(0) \stackrel{\text{df}}{=} 2$ and $\mu(i+1) \stackrel{\text{df}}{=} 2^{2^{\mu(i)}}$ for each $i \geq 0$,

$$R_k \stackrel{\text{df}}{=} \{i \mid i \in \mathbb{N} \wedge \mu(k) \leq i < \mu(k+1)\},$$

and the following two classes of languages:[9]

$$\mathcal{C}_1 \stackrel{\text{df}}{=} \{A \subseteq \mathbb{N} \mid (\forall j \geq 0)\,[R_{2j} \cap A = \emptyset \wedge (\forall x, y \in R_{2j+1})\,[(x \leq y \wedge x \in A) \Rightarrow y \in A]]\};$$

$$\mathcal{C}_2 \stackrel{\text{df}}{=} \{A \subseteq \mathbb{N} \mid (\forall j \geq 0)\,[R_{2j} \cap A = \emptyset \wedge (\forall x, y \in R_{2j+1})\,[(x \leq y \wedge y \in A) \Rightarrow x \in A]]\}.$$

**Lemma 5.4.5** [HJ] $\mathcal{C}_1 \cap \mathrm{E} \subseteq$ P-Sel and $\mathcal{C}_2 \cap \mathrm{E} \subseteq$ P-Sel.

**Remark 5.4.6**

1. We will apply Lemma 5.4.5 in a slightly more general form in the proof of Theorem 5.4.7 below. That is, in the definition of $\mathcal{C}_1$ and $\mathcal{C}_2$, the underlying ordering of the elements in the regions $R_{2j+1}$ need not be the standard lexicographical order of strings. We may allow *any* ordering $\prec$ that respects the lengths of strings and such that, given two strings, $x$ and $y$, of the same length, it can be decided in polynomial time whether $x \prec y$. Also observe that in this technique (of constructing *widely-spaced* and *complexity-bounded* sets that thus are in P-Sel, since smaller strings can be solved by brute force), there is nothing special about spacing according to the $\mu$-function above and the complexity bound being E. One only needs the spacing to be at least as wide as $\nu(0) = 2$ and $\nu(i+1) = 2^{t(\nu(i))}$ for each $i \geq 0$, if the complexity bound is DTIME$[t(n)]$ (as in the proof of Theorem 5.3.3).

2. To accomplish the diagonalizations in this section, we need our enumeration of FP functions to satisfy a technical requirement. Fix an enumeration of all polynomial-time transducers $\{T_i\}_{i \geq 1}$ having the property that each transducer appears infinitely often in the list. That is, if $T = T_i$ (here, equality refers to the actual program) for some $i$, then there is an infinite set $J$ of distinct integers such that for each $j \in J$, we

[8] For any $x$ and $y$ in $V$, define $x \preceq_f y$ if and only if $(\exists u_1, \ldots, u_k)\,[x = u_1 \wedge y = u_k \wedge (\forall i : 1 \leq i \leq k-1)\,[f(u_i, u_{i+1}) = u_{i+1}]]$.

[9] We will implicitly use the standard correspondence between $\Sigma^*$ and $\mathbb{N}$.

have $T = T_j$. For each $k \geq 1$, let $f_k$ denote the function computed by $T_k$. In the diagonalizations below, it is enough to diagonalize for all $k$ against some $T_{k'}$ such that $T_k = T_{k'}$, i.e., both compute $f_k$. In particular, for keeping the sets $L_1$ and $L_2$ (to be defined in the upcoming proofs of Theorems 5.4.7 and 5.4.8) in E, we will construct $L_1$ and $L_2$ such that for all stages $j$ of the construction and for any set of inputs $X \subseteq R_{2j+1}$, the transducer computing $f_j(X)$ runs in time less than $2^{\max\{|x|\,:\,x\in X\}}$ (i.e., the simulation of $T_j$ on input $X$ is aborted if it fails to be completed in this time bound, and the construction of $L_1$ and $L_2$ proceeds to the next stage). The diagonalization is still correct, since for each $T_i$ there is a number $b_i$ (depending only on $T_i$) such that for each $k \geq b_i$, if $T_i = T_k$, then for $T_k$ we will properly diagonalize—and thus $T_i$ is implicitly diagonalized against.

3. For each $j \geq 0$ and $k < \|R_{2j+1}\|$, let $x_{j,0}, \ldots, x_{j,k}$ denote the strings corresponding to the first $k+1$ numbers in region $R_{2j+1}$ (in the standard correspondence between $\Sigma^*$ and $\mathbb{N}$). This notation is used in the diagonalization proofs of this section.

**Theorem 5.4.7** 1. P-Sel $\bigwedge$ P-Sel $\subseteq$ P-mc$(3)$, yet P-Sel $\bigwedge$ P-Sel $\not\subseteq$ P-mc$(2)$.

2. P-Sel $\bigvee$ P-Sel $\subseteq$ P-mc$(3)$, yet P-Sel $\bigvee$ P-Sel $\not\subseteq$ P-mc$(2)$.

3. P-Sel $\boldsymbol{\Delta}$ P-Sel $\not\subseteq$ P-mc$(3)$ and P-Sel $\overline{\boldsymbol{\Delta}}$ P-Sel $\not\subseteq$ P-mc$(3)$.

**Proof.** 1. & 2. Let $A \in$ P-Sel via $f$ and $B \in$ P-Sel via $g$, and let $\preceq_f$ and $\preceq_g$ be the orders induced by $f$ and $g$, respectively. Fix any inputs $x_1, x_2, x_3$ such that $x_1 \preceq_f x_2 \preceq_f x_3$. If $f$ and $g$ "agree" on any two of these strings, i.e., there exist $i, j \in \{1,2,3\}$ with $i < j$ and $x_i \preceq_g x_j$, then define a P-mc(3) function $h$ for $A \cap B$ to output a "1" at position $i$ and a 0 at position $j$. Otherwise (i.e., if $x_3 \preceq_g x_2 \preceq_g x_1$), define $h(x_1, x_2, x_3) \stackrel{\mathrm{df}}{=} 101$. In each case, we have $(\chi_{A\cap B}(x_1), \chi_{A\cap B}(x_2), \chi_{A\cap B}(x_3)) \neq h(x_1, x_2, x_3)$. A similar construction works for $A \cup B$ if we define $h(x_1, x_2, x_3) \stackrel{\mathrm{df}}{=} 010$ if $x_3 \preceq_g x_2 \preceq_g x_1$, and as above in the other cases. This proves P-Sel $\bigwedge$ P-Sel $\subseteq$ P-mc$(3)$ and P-Sel $\bigvee$ P-Sel $\subseteq$ P-mc$(3)$.

For proving the diagonalizations, recall from Remark 5.4.6 that $x_{j,0}, \ldots, x_{j,k}$ denote the smallest $k+1$ numbers in region $R_{2j+1}$. Define $L_1 \stackrel{\mathrm{df}}{=} \bigcup_{j\geq 0} L_{1,j}$ and $L_2 \stackrel{\mathrm{df}}{=} \bigcup_{j\geq 0} L_{2,j}$, where

$$L_{1,j} \stackrel{\mathrm{df}}{=} \left\{ i \in R_{2j+1} \;\middle|\; \begin{array}{l} (f_j(x_{j,0}, x_{j,1}) \in \{00, 01\} \wedge i \geq x_{j,1}) \vee \\ (f_j(x_{j,0}, x_{j,1}) \in \{10, 11\} \wedge i \geq x_{j,0}) \end{array} \right\};$$

$$L_{2,j} \stackrel{\text{df}}{=} \left\{ i \in R_{2j+1} \;\middle|\; \begin{array}{l} (f_j(x_{j,0}, x_{j,1}) \in \{00, 10\} \wedge i \leq x_{j,0}) \vee \\ (f_j(x_{j,0}, x_{j,1}) \in \{01, 11\} \wedge i \leq x_{j,1}) \end{array} \right\}.$$

Clearly, by the above remark about the construction of $L_1$ and $L_2$, we have $L_1 \in \mathcal{C}_1 \cap E$ and $L_2 \in \mathcal{C}_2 \cap E$. Thus, by Lemma 5.4.5, $L_1$ and $L_2$ are in P-Sel. Supposing $L_1 \cap L_2 \in$ P-mc$(2)$ via $f_{j_0}$ for some $j_0$, we have $f_{j_0}(x_{j_0,0}, x_{j_0,1}) \in \{0, 1\}^2$ such that

$$(\chi_{L_1 \cap L_2}(x_{j_0,0}), \chi_{L_1 \cap L_2}(x_{j_0,1})) \neq f_{j_0}(x_{j_0,0}, x_{j_0,1}).$$

However, in each of the four cases for the membership of $x_{j_0,0}$ and $x_{j_0,1}$ in $L_1 \cap L_2$, this is by definition of $L_1$ and $L_2$ exactly what $f_{j_0}$ claims is impossible. Therefore, P-Sel $\wedge$ P-Sel $\not\subseteq$ P-mc$(2)$. Furthermore, since P-Sel is closed under complementation, $\overline{L_1}, \overline{L_2} \in$ P-Sel. Now assume P-Sel $\vee$ P-Sel $\subseteq$ P-mc$(2)$. Then, $\overline{L_1} \cup \overline{L_2} = \overline{L_1 \cap L_2}$ is in P-mc$(2)$, and since P-mc$(2)$ is closed under complementation, we have $L_1 \cap L_2 \in$ P-mc$(2)$, a contradiction. Hence, P-Sel $\vee$ P-Sel $\not\subseteq$ P-mc$(2)$.

3. Let $L_1 \stackrel{\text{df}}{=} \bigcup_{j \geq 0} L_{1,j}$, where $L_{1,j}$ is the set of all $i \in R_{2j+1}$ such that

1. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) \in \{100, 101, 111\} \wedge i \geq x_{j,0})$ or
2. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) = 011 \wedge i \geq x_{j,1})$ or
3. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) \in \{001, 110\} \wedge i \geq x_{j,2})$.

Thus, $L_1 \in \mathcal{C}_1 \cap E$, and by Lemma 5.4.5, $L_1 \in$ P-Sel. For defining $L_2$, we assume the following re-ordering of the elements in $R_{2j+1}$ for each $j \geq 0$: $x_{j,1} \prec x_{j,2} \prec x_{j,0} \prec x_{j,3}$ and $x_{j,s} \prec x_{j,s+1}$ if and only if $x_{j,s} < x_{j,s+1}$ for $s \geq 3$. For any strings $x$ and $y$, we write $x \preceq y$ if $x \prec y$ or $x = y$. Now define $L_2 \stackrel{\text{df}}{=} \bigcup_{j \geq 0} L_{2,j}$, where $L_{2,j}$ is the set of all $i \in R_{2j+1}$ such that

1. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) = 110 \wedge i \preceq x_{j,0})$ or
2. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) \in \{010, 101\} \wedge i \preceq x_{j,1})$ or
3. $(f_j(x_{j,0}, x_{j,1}, x_{j,2}) = 100 \wedge i \preceq x_{j,2})$.

By Remark 5.4.6, $L_2 \in$ P-Sel. Note that for each $j \geq 0$, the set $L_1 \cap R_{2j+1}$ is empty if $f_j(x_{j,0}, x_{j,1}, x_{j,2}) \in \{000, 010\}$, and the set $L_2 \cap R_{2j+1}$ is empty if $f_j(x_{j,0}, x_{j,1}, x_{j,2})$

is in $\{000, 001, 011, 111\}$. Now suppose $L_1 \Delta L_2 \in$ P-mc$(3)$ via $f_{j_0}$ for some $j_0$, i.e., $f_{j_0}(x_{j_0,0}, x_{j_0,1}, x_{j_0,2}) \in \{0,1\}^3$ such that

$$(\chi_{L_1\Delta L_2}(x_{j_0,0}), \chi_{L_1\Delta L_2}(x_{j_0,1}), \chi_{L_1\Delta L_2}(x_{j_0,2})) \neq f_{j_0}(x_{j_0,0}, x_{j_0,1}, x_{j_0,2}).$$

However, in each of the eight cases for the membership of $x_{j_0,0}$, $x_{j_0,1}$, and $x_{j_0,2}$ in $L_1 \Delta L_2$, this is by definition of $L_1$ and $L_2$ exactly what $f_{j_0}$ claims is impossible. Therefore, P-Sel $\boldsymbol{\Delta}$ P-Sel $\not\subseteq$ P-mc$(3)$. Since $L_1 \overline{\Delta} \overline{L_2} = L_1 \Delta L_2$ and $\overline{L_2} \in$ P-Sel, this also implies that P-Sel $\overline{\boldsymbol{\Delta}}$ P-Sel $\not\subseteq$ P-mc$(3)$. □

Note that Theorem 5.4.7 does not contradict Ogihara's result in [Ogi94] that $\mathfrak{R}^p_{2\text{-tt}}(\text{P-Sel})$ is contained in P-mc$(2)$, since we consider the union and intersection of two possibly *different* sets in P-Sel, whereas the two queries in a $\leq^p_{2\text{-tt}}$-reduction are asked to the *same* set in P-Sel. Clearly, if P-Sel were closed under join, then we indeed would have a contradiction. However, P-Sel is not closed under join [HJ].

**Theorem 5.4.8** [10]

1. For each $k \geq 2$, $\bigoplus_k(\text{P-Sel}) \subseteq \text{GC}(1,k,1)$, but $\bigoplus_k(\text{P-Sel}) \not\subseteq \text{SH} \cup \text{GC}(1,k-1,1)$.
2. For each $k \geq 2$, $\bigvee_k(\text{P-Sel}) \subseteq \text{GC}(1,k,1)$, but $\bigvee_k(\text{P-Sel}) \not\subseteq \text{SH} \cup \text{GC}(1,k-1,1)$.
3. P-Sel $\bigwedge$ P-Sel $\not\subseteq$ GC$(1,2,1)$, but for each integer-valued FP function $k(0^n)$ satisfying $1 \leq k(0^n) \leq n$, P-Sel $\bigwedge$ P-Sel $\subseteq \text{GC}(\lceil \frac{n}{k(0^n)} \rceil, k(0^n), 1)$.[11]
4. P-Sel op P-Sel $\not\subseteq$ fair-GC$(1, n-1, 1)$ for op $\in \{\bigwedge, \boldsymbol{\Delta}, \overline{\boldsymbol{\Delta}}\}$.

**Proof.** 1. & 2. Let $L = A_1 \oplus \cdots \oplus A_k$, where $A_i \in$ P-Sel via selector functions $s_i$ for $i \in \{1, \ldots, k\}$. Let any inputs $x_1, \ldots, x_m$ be given, each having the form $\underline{i}a$ for some $i \in \{1, \ldots, k\}$ and $a \in \Sigma^*$. For each $i$, play a knock-out tournament among all strings $a$ for which $\underline{i}a$ belongs to the inputs, where we say $a_1$ beats $a_2$ if $a_2 \preceq_{s_i} a_1$. Let $w_1, \ldots, w_k$ be the winners of the $k$ tournaments. Define a GC$(1,k,1)$-selector for $L$ to output $\{\underline{1}w_1, \ldots, \underline{k}w_k\}$. Clearly, at least one of these strings must be in $L$ if at least one of the inputs is in $L$. The proof of $\bigvee_k(\text{P-Sel}) \subseteq \text{GC}(1,k,1)$ is similar.

[10] Note that some parts of this theorem extend Hemaspaandra and Jiang's results in [HJ], and also Rao's observation that P-Sel op P-Sel $\not\subseteq$ SH for any Boolean operation op chosen from $\{\wedge, \vee, \Delta\}$ [Rao94].

[11] Note that there is still a gap between the upper and the lower bound.

We only prove that P-Sel $\bigvee$ P-Sel $\not\subseteq$ SH by uniformly diagonalizing against all FP functions and all levels of SH. Define

$$L_1 \stackrel{\text{df}}{=} \bigcup_{\langle j,m\rangle : j\geq 0 \wedge m<\|R_{2j+1}\|} L_{1,\langle j,m\rangle} \quad \text{and} \quad L_2 \stackrel{\text{df}}{=} \bigcup_{\langle j,m\rangle : j\geq 0 \wedge m<\|R_{2j+1}\|} L_{2,\langle j,m\rangle},$$

where for each $j \geq 0$ and $m < \|R_{2j+1}\|$, the sets $L_{1,\langle j,m\rangle}$ and $L_{2,\langle j,m\rangle}$ are defined as follows:

$$\begin{aligned} L_{1,\langle j,m\rangle} &\stackrel{\text{df}}{=} \{i \in R_{2j+1} \mid i > f_j(x_{j,0},\dots,x_{j,m}) \wedge f_j(x_{j,0},\dots,x_{j,m}) \in \{x_{j,0},\dots,x_{j,m}\}\}; \\ L_{2,\langle j,m\rangle} &\stackrel{\text{df}}{=} \{i \in R_{2j+1} \mid i < f_j(x_{j,0},\dots,x_{j,m}) \wedge f_j(x_{j,0},\dots,x_{j,m}) \in \{x_{j,0},\dots,x_{j,m}\}\}. \end{aligned}$$

Clearly, $L_1 \in \mathcal{C}_1 \cap E$ and $L_2 \in \mathcal{C}_2 \cap E$. Thus, by Lemma 5.4.5, $L_1, L_2 \in$ P-Sel. Assume P-Sel $\bigvee$ P-Sel $\subseteq$ SH, and in particular, $L_1 \cup L_2 \in S(m_0)$ via $f_{j_0}$. If $m_0 < \|R_{2j_0+1}\|$, then this contradicts the fact that $f_{j_0}(x_{j_0,0},\dots,x_{j_0,m_0})$ selects a string not in $L_1 \cup L_2$ though $m_0$ of the inputs are in $L_1 \cup L_2$. If $m_0 \geq \|R_{2j_0+1}\|$, then by our assumption that each transducer $T_i$ appears infinitely often in the enumeration (see Remark 5.4.6), there is an index $j_1$ such that $m_0 < \|R_{2j_1+1}\|$ and $T_{j_1}$ computes $f_{j_0}$, and thus $f_{j_0}$ is implicitly diagonalized against.

3. Let $k(0^n)$ be a function as in the theorem. Let $L = A \cap B$ for sets $A$ and $B$, where $A \in$ P-Sel via $f$ and $B \in$ P-Sel via $g$. We will define a $\mathrm{GC}(\lceil \frac{n}{k(0^n)} \rceil, k(0^n), 1)$-selector $s$ for $L$. Given $n$ elements, rename them with respect to the linear order induced by $f$, i.e., we have $x_1 \preceq_f x_2 \preceq_f \cdots \preceq_f x_n$. Let $k \stackrel{\text{df}}{=} k(0^n)$. Now let $h$ be the unique permutation of $\{1,\dots,n\}$ such that for each $i, j \in \{1,\dots,n\}$, $h(i) = j$ if and only if $x_i$ is the $j$th element in the linear ordering of $\{x_1,\dots,x_n\}$ induced by $g$. Partition the set $\{1,\dots,n\}$ into $k$ regions of at most $\lceil \frac{n}{k} \rceil$ elements:

$$\begin{aligned} R(l) &\stackrel{\text{df}}{=} \left\{(l-1)\left\lceil\frac{n}{k}\right\rceil+1, (l-1)\left\lceil\frac{n}{k}\right\rceil+2, \dots, l\left\lceil\frac{n}{k}\right\rceil\right\} \quad \text{for } 1 \leq l \leq k-1, \text{ and} \\ R(k) &\stackrel{\text{df}}{=} \left\{(k-1)\left\lceil\frac{n}{k}\right\rceil+1, (k-1)\left\lceil\frac{n}{k}\right\rceil+2, \dots, n\right\}. \end{aligned}$$

Define $s(x_1,\dots,x_n) \stackrel{\text{df}}{=} \{a_1,\dots a_k\}$, where $a_l \stackrel{\text{df}}{=} x_{m(l)}$ and $m(l)$ is the $m \in R(l)$ such that $h(m)$ is maximum. Thus, for each region $R(l)$, $a_l$ is the "most likely" element of its region to belong to $B$. Consider the permutation matrix of $h$ with elements $(i, h(i))$, for $1 \leq i \leq n$. Let $c_A$ be the "cutpoint" for $A$ and let $c_B$ be the "cutpoint" for $B$, i.e.,

$$\begin{aligned} \{x_i \mid i < c_A\} \subseteq \overline{A} \quad &\text{and} \quad \{x_i \mid i \geq c_A\} \subseteq A; \\ \{x_{h(i)} \mid h(i) < c_B\} \subseteq \overline{B} \quad &\text{and} \quad \{x_{h(i)} \mid h(i) \geq c_B\} \subseteq B. \end{aligned}$$

Define

$$
\begin{array}{lll}
A_{\text{out}} \stackrel{\text{df}}{=} \{x_i \mid i < c_A\}; & \qquad & A_{\text{in}} \stackrel{\text{df}}{=} \{x_i \mid i \geq c_A\}; \\
B_{\text{out}} \stackrel{\text{df}}{=} \{x_{h(i)} \mid h(i) < c_B\}; & \qquad & B_{\text{in}} \stackrel{\text{df}}{=} \{x_{h(i)} \mid h(i) \geq c_B\}.
\end{array}
$$

Since $A_{\text{in}} \cap B_{\text{in}} \subseteq A \cap B$, it remains to show that if the promise $\|\{x_1, \ldots, x_n\} \cap L\| \geq \lceil \frac{n}{k} \rceil$ is met, then at least one of the outputs $a_l$ of $s$ is in $A_{\text{in}} \cap B_{\text{in}}$. First observe that for each $l$, if $i \geq c_A$ holds for each $i \in R(l)$ and $R(l)$ contains an index $i_0$ such that $h(i_0) \geq c_B$, then $a_l \in A_{\text{in}} \cap B_{\text{in}}$. On the other hand, if $c_A$ "cuts" a region $R(l_0)$, then in the worst case we have $a_{l_0} = (l_0 - 1)\lceil \frac{n}{k} \rceil + 1$ and $c_A = (l_0 - 1)\lceil \frac{n}{k} \rceil + 2$, and thus $a_{l_0} \notin A_{\text{in}}$ and at most $\lceil \frac{n}{k} \rceil - 1$ elements of $A_{\text{in}}$ can have an index in $R(l_0)$. However, if $\|\{x_1, \ldots, x_n\} \cap L\| \geq \lceil \frac{n}{k} \rceil$, then there must exist an $l_1$ with $l_1 > l_0$ such that for each $i \in R(l_1)$ it holds that $i \geq c_A$, and thus, $a_{l_1} \in A_{\text{in}} \cap B_{\text{in}}$. This proves $L \in \mathrm{GC}(\lceil \frac{n}{k} \rceil, k, 1)$ via $s$.

The proof of P-Sel $\wedge$ P-Sel $\not\subseteq \mathrm{GC}(1, 2, 1)$ is similar as in Part 4.

4. We only prove P-Sel $\wedge$ P-Sel $\not\subseteq$ fair-$\mathrm{GC}(1, n-1, 1)$. Define

$$
L_1 \stackrel{\text{df}}{=} \left\{ i \;\middle|\; \begin{array}{l} (\exists j \geq 0)\, [i \in R_{2j+1} \text{ and } i \geq w_j \text{ for the smallest string} \\ w_j \in R_{2j+1} \text{ such that } f_j(R_{2j+1}) \subseteq R_{2j+1} - \{w_j\}] \end{array} \right\};
$$

$$
L_2 \stackrel{\text{df}}{=} \left\{ i \;\middle|\; \begin{array}{l} (\exists j \geq 0)\, [i \in R_{2j+1} \text{ and } i \leq w_j \text{ for the smallest string} \\ w_j \in R_{2j+1} \text{ such that } f_j(R_{2j+1}) \subseteq R_{2j+1} - \{w_j\}] \end{array} \right\}.
$$

As before, $L_1, L_2 \in$ P-Sel. Assume there is a fair-$\mathrm{GC}(1, n-1, 1)$-selector $f_{j_0}$ for $L_1 \cap L_2$. First observe that the "fair condition" is satisfied if $f_{j_0}$ has all strings from $R_{2j_0+1}$ as inputs, since $\|R_{2j_0+1}\| = 2^{2^{\mu(2j_0+1)}} - \mu(2j_0 + 1)$ and the length of the largest string in $R_{2j_0+1}$ is at most $2^{\mu(2j_0+1)}$. For fair-$\mathrm{GC}(1, n-1, 1)$-selector $f_{j_0}$, there must exist a smallest string $w_{j_0} \in R_{2j_0+1}$ such that $f_{j_0}(R_{2j_0+1}) \subseteq R_{2j_0+1} - \{w_{j_0}\}$, and thus, $\{w_{j_0}\} = L_1 \cap L_2 \cap R_{2j_0+1}$. This would contradict $f_{j_0}(R_{2j_0+1})$ not selecting $w_{j_0}$. $\square$

Statement 3 of the above theorem immediately gives the first part of Corollary 5.4.9. Note that, even though this $\mathrm{GC}(\sqrt{n}, \sqrt{n}, 1)$ upper bound on P-Sel $\wedge$ P-Sel may not be strong enough to prove the second part of the corollary, the proof of this second part does easily follow from the P-Sel $\wedge$ P-Sel $\subseteq$ P-mc$(3)$ result of Theorem 5.4.7 via Ogihara's result that the assumption NP $\subseteq$ P-mc$(3)$ implies the collapse of P $=$ NP [Ogi94].

**Corollary 5.4.9** 1. P-Sel $\wedge$ P-Sel $\subseteq \mathrm{GC}(\sqrt{n}, \sqrt{n}, 1)$.

2. $\mathrm{NP} \subseteq \text{P-Sel} \wedge \text{P-Sel} \implies \mathrm{P} = \mathrm{NP}$.

The rest of this section studies the internal structure of GCH. We start with determining for which parameters $b$, $c$, and $d$ the class $\mathrm{GC}(b, c, d)$ is "nontrivial." Throughout this chapter, a class $\mathcal{C}$ of sets is said to be *nontrivial* if $\mathcal{C} \neq \mathfrak{P}(\Sigma^*)$ and $\mathcal{C}$ contains not only finite sets. Recall that $w_{i,1}, \ldots, w_{i,s}$ are the lexicographically smallest $s$ length $e(i)$ strings, for $i \geq 0$ and $s \leq 2^{e(i)}$ (function $e(i)$ is defined in Section 5.2). The proof of Lemma 5.4.10 below can be found in the appendix on page 93.

**Lemma 5.4.10** Let $b, c, d \in \mathbb{N}^+$ with $d \leq c$ and $d \leq b$. Then,

1. $(\exists A)\, [A \in \mathrm{GC}(b, c, d) \wedge \|A\| = \infty]$, and
2. $(\exists B)\, [B \notin \mathrm{GC}(b, c, d) \wedge \|B\| = \infty]$.

**Theorem 5.4.11** Let $b, c, d \in \mathbb{N}^+$.

1. Every set in $\mathrm{GC}(b, c, d)$ is finite if and only if $d > b$ or $d > c$.
2. If $d \leq b$ and $d \leq c$, then $\mathrm{GC}(b, c, d)$ is nontrivial.

**Proof.** If $d > c$ or $d > b$, then by Definition 5.4.1, every set in $\mathrm{GC}(b, c, d)$ is finite. On the other hand, if $d \leq b$ and $d \leq c$, then by Lemma 5.4.10.1, there is an infinite set in $\mathrm{GC}(b, c, d)$. Hence, every set in $\mathrm{GC}(b, c, d)$ is finite if and only if $d > b$ or $d > c$. Furthermore, if $d \leq b$ and $d \leq c$, then $\mathrm{GC}(b, c, d) \neq \mathfrak{P}(\Sigma^*)$ by Lemma 5.4.10.2. □

Now we turn to the relationships between the nontrivial classes within GCH. Given any parameters $b, c, d$ and $i, j, k$, we seek to determine which of $\mathrm{GC}(b, c, d)$ and $\mathrm{GC}(i, j, k)$ is contained in the other class (and if this inclusion is strict), or whether they are mutually incomparable. For classes $\mathcal{A}$ and $\mathcal{B}$, let $\mathcal{A} \bowtie \mathcal{B}$ denote that $\mathcal{A}$ and $\mathcal{B}$ are incomparable, i.e., $\mathcal{A} \not\subseteq \mathcal{B}$ and $\mathcal{B} \not\subseteq \mathcal{A}$. Theorem 5.4.14 will establish these relations for almost all the cases and is proven by making extensive use of the Inclusion Lemma and the Diagonalization Lemma below. The proofs of Lemmas 5.4.12 and 5.4.13 can be found in the appendix on pages 93 and 94, respectively.

**Lemma 5.4.12 (Inclusion Lemma)** Let $b, c, d \in \mathbb{N}^+$ and $l, m, n \in \mathbb{N}$ be given such that each GC class below is nontrivial. Then,

1. $\mathrm{GC}(b, c, c) = \mathrm{S}(b, c)$.

2. $\mathrm{GC}(b, c, d + n) \subseteq \mathrm{GC}(b + l, c + m, d)$.

3. If $l \geq n$ and $m \geq n$, then $\mathrm{GC}(b, c, c) \subseteq \mathrm{GC}(b + l, c + m, c + n)$.

4. If $l \leq n$ and $m \leq n$, then $\mathrm{GC}(b + l, c + m, d + n) \subseteq \mathrm{GC}(b, c, d)$.

**Lemma 5.4.13 (Diagonalization Lemma)** Let $b, c, d \in \mathbb{N}^+$ and $l, m, n, q \in \mathbb{N}$ be given such that each GC class below is nontrivial. Then,

1. If $l \geq n + 1$, then $(\exists L)\ [L \in \mathrm{GC}(b + l, c + m, d + n) - \mathrm{GC}(b, c + q, d)]$.

2. If $m \geq n + 1$, then $(\exists L)\ [L \in \mathrm{GC}(b + l, c + m, d + n) - \mathrm{GC}(b + q, c, d)]$.

3. If ($n \geq l + 1$ or $n \geq m + 1$), then $(\exists L)\ [L \in \mathrm{GC}(b, c, d) - \mathrm{GC}(b + l, c + m, d + n)]$.

**Theorem 5.4.14** Let $b, c, d \in \mathbb{N}^+$ and $i, j, k \in \mathbb{N}$ be given such that each GC class below is nontrivial. Then,

1. $\mathrm{GC}(b, c, d + k) \subset \mathrm{GC}(b + i, c + j, d)$ if $i \geq 1$ or $j \geq 1$ or $k \geq 1$.

2. $\mathrm{GC}(b, c + j, d + k) \subset \mathrm{GC}(b + i, c, d)$ if $1 \leq j \leq k$.

3. $\mathrm{GC}(b, c + j, d + k) \bowtie \mathrm{GC}(b + i, c, d)$ if $j > k \geq 1$.

4. $\mathrm{GC}(b + i, c, d + k) \subset \mathrm{GC}(b, c + j, d)$ if $1 \leq i \leq k$.

5. $\mathrm{GC}(b + i, c, d + k) \bowtie \mathrm{GC}(b, c + j, d)$ if $i > k \geq 1$.

6. $\mathrm{GC}(b + i, c, d) \bowtie \mathrm{GC}(b, c + j, d)$ if $i \geq 1$ and $j \geq 1$.

7. $\mathrm{GC}(b + i, c + j, d + k) \subset \mathrm{GC}(b, c, d)$ if ($1 \leq i < k$ and $1 \leq j \leq k$) or ($1 \leq j < k$ and $1 \leq i \leq k$).

8. $\mathrm{GC}(b + i, c + j, d + k) = \mathrm{GC}(b, c, d)$ if $i = j = k$ and $c = d$.

9. $\mathrm{GC}(b + i, c + j, d + k) \bowtie \mathrm{GC}(b, c, d)$ if $1 \leq i < k < j$ or $1 \leq j < k < i$.

**Proof.** The proof is done by repeatedly applying Lemma 5.4.12 and Lemma 5.4.13. Unless otherwise specified, $l$, $m$, and $n$ in the lemmas correspond to $i$, $j$, and $k$ in this proof.

1. The inclusion is clear (see Lemma 5.4.12.2). For the strictness of the inclusion, we have to consider three cases. If $i \geq 1$, then by Lemma 5.4.13.1 with $n = q = 0$, there exists a set $L \in GC(b+i, c+j, d) - GC(b, c, d)$. By Lemma 5.4.12.2 with $l = m = 0$, $L \notin GC(b, c, d+k)$. The case of $j \geq 1$ is treated similar, using Lemma 5.4.13.2 instead of Lemma 5.4.13.1. Finally, if $k \geq 1$, then by Lemma 5.4.13.3 with $l = m = 0$, we have $L \in GC(b, c, d) - GC(b, c, d+k)$. By Lemma 5.4.12.2 with $n = 0$, $L \in GC(b+i, c+j, d)$.

2. Applying Lemma 5.4.12.4 with $l = 0$ and then Lemma 5.4.12.2 with $m = n = 0$, we have $GC(b, c+j, d+k) \subseteq GC(b, c, d) \subseteq GC(b+i, c, d)$. By Lemma 5.4.13.3 with $l = 0$ (i.e., $n \geq 1$), there exists a set $L \in GC(b, c, d) - GC(b, c+j, d+k)$. By Lemma 5.4.12.2 with $m = n = 0$, $L \in GC(b+i, c, d)$.

3. "$\not\subseteq$" follows from Lemma 5.4.13.2 with $q = i$ and $l = 0$. "$\not\supseteq$" follows as in Part 2.

4. Applying Lemma 5.4.12.4 with $m = 0$ and then Lemma 5.4.12.2 with $l = n = 0$, we have $GC(b+i, c, d+k) \subseteq GC(b, c, d) \subseteq GC(b, c+m, d)$. The strictness of the inclusion follows as in Part 2, where Lemma 5.4.13.3 is applied with $m = 0$ instead of $l = 0$.

5. "$\not\subseteq$" follows from Lemma 5.4.13.1 with $q = j$ and $m = 0$. "$\not\supseteq$" holds by Lemma 5.4.13.3 with $m = 0$ (i.e., $n \geq 1$) and Lemma 5.4.12.2 with $l = n = 0$.

6. "$\not\subseteq$" holds, as by Lemma 5.4.13.1 with $q = j$ and $m = n = 0$, there exists a set $L \in GC(b+i, c, d) - GC(b, c+j, d)$. "$\not\supseteq$" similarly follows from Lemma 5.4.13.2 with $q = i$ and $l = n = 0$.

7. By Lemma 5.4.12.4, $GC(b+i, c+j, d+k) \subseteq GC(b, c, d)$. By Lemma 5.4.13.3, if $n > l$ or $n > m$, then there exists a set $L \in GC(b, c, d) - GC(b+i, c+j, d+k)$.

8. The equality follows from Lemma 5.4.12.3 and Lemma 5.4.12.4.

9. Let $i < k < j$. Then, by Lemma 5.4.13.2 with $q = 0$, there exists a set $L \in GC(b+i, c+j, d+k) - GC(b, c, d)$. Conversely, by Lemma 5.4.13.3, there exists a set $L \in GC(b, c, d) - GC(b+i, c+j, d+k)$. If $j < k < i$, the incomparability of $GC(b, c, d)$ and $GC(b+i, c+j, d+k)$ similarly follows from Lemma 5.4.13.1 and Lemma 5.4.13.3. □

Note that Theorem 5.4.14 does not settle all possible relations between the GC classes. That is, the relation between $GC(b, c, d)$ and $GC(b+i, c+j, d+k)$ is left open for

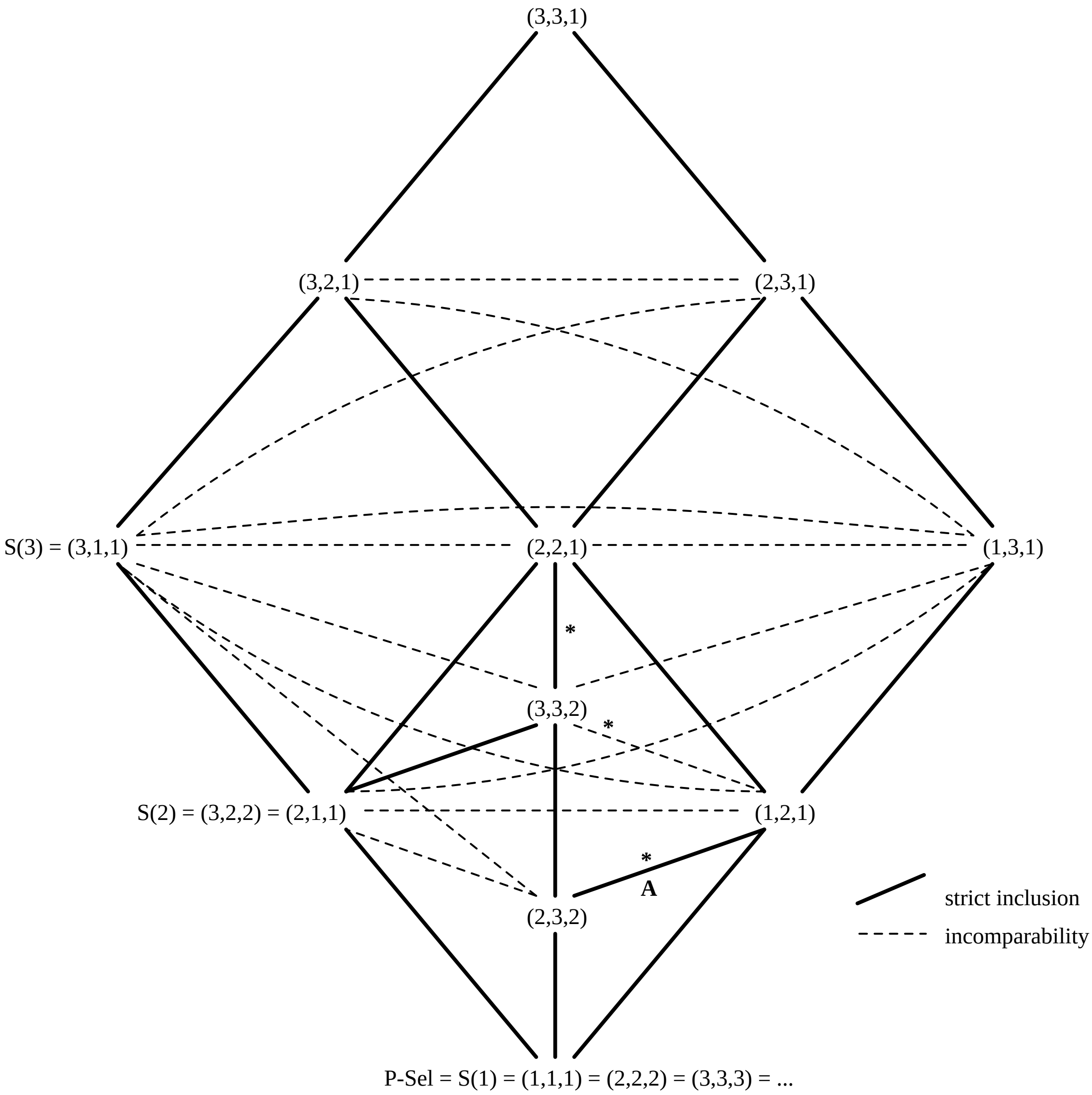


Figure 5.3: Relations between all nontrivial classes $\mathrm{GC}(b, c, d)$ with $1 \leq b, c, d \leq 3$.

the case of ($k \leq i$ and $k \leq j$ and $c \neq d$). Figure 5.3 shows the relations amongst all nontrivial classes $\mathrm{GC}(b, c, d)$ with $1 \leq b, c, d \leq 3$, as they are proven in Theorem 5.4.14. For instance, $\mathrm{S}(2) = \mathrm{GC}(3, 2, 2) \subset \mathrm{GC}(3, 3, 2)$ holds by the first part of the theorem with $b = 3, c = d = 2, i = k = 0$, and $j = 1$. Those relations not established by Theorem 5.4.14 are marked by "$*$" and are proven separately as Theorem 5.4.15 below. The "A" indicates that, while the inclusion holds by Lemma 5.4.12.4, the *strictness* of the inclusion for these cases has been observed by A. Nickelsen.

**Theorem 5.4.15** 1. [Nic94] $\mathrm{GC}(2, 3, 2) \subset \mathrm{GC}(1, 2, 1)$.

2. $\mathrm{GC}(3, 3, 2) \bowtie \mathrm{GC}(1, 2, 1)$.

3. $\mathrm{GC}(3, 3, 2) \subset \mathrm{GC}(2, 2, 1)$.

**Proof.** Both inclusions ($\mathrm{GC}(2, 3, 2) \subseteq \mathrm{GC}(1, 2, 1)$ and $\mathrm{GC}(3, 3, 2) \subseteq \mathrm{GC}(2, 2, 1)$) follow from Lemma 5.4.12.4 with $l = m = n = 1$. We now provide the diagonalizations.

1. For proving $\mathrm{GC}(1, 2, 1) \not\subseteq \mathrm{GC}(2, 3, 2)$, we will define a set $L = \bigcup_{i \geq 1} L_i$ such that for each $i$, $L_i \subseteq W_{i,4}$, and if $f_i(W_{i,4}) \subseteq W_{i,4}$ and $\|f_i(W_{i,4})\| = 3$, then we make sure that $\|L_i\| = 2$ and $\|L_i \cap f_i(W_{i,4})\| = 1$. This ensures that for no $i \geq 1$ can $f_i$ be a $\mathrm{GC}(2, 3, 2)$-selector for $L$. For example, this can be accomplished by defining $L_i$ as follows:

$$
\begin{aligned}
\chi_L(w_{i,1}, \ldots, w_{i,4}) &= 0101 \quad \text{if} \quad f_i(W_{i,4}) = \{w_{i,1}, w_{i,2}, w_{i,3}\}, \\
\chi_L(w_{i,1}, \ldots, w_{i,4}) &= 1010 \quad \text{if} \quad f_i(W_{i,4}) = \{w_{i,1}, w_{i,2}, w_{i,4}\}, \\
\chi_L(w_{i,1}, \ldots, w_{i,4}) &= 1100 \quad \text{if} \quad f_i(W_{i,4}) = \{w_{i,1}, w_{i,3}, w_{i,4}\}, \\
\chi_L(w_{i,1}, \ldots, w_{i,4}) &= 1100 \quad \text{if} \quad f_i(W_{i,4}) = \{w_{i,2}, w_{i,3}, w_{i,4}\}.
\end{aligned}
$$

Note that if $f_i(W_{i,4})$ outputs a string not in $W_{i,4}$ or the number of output strings is different from 3, then (by Definition 5.4.1 and Remark 5.4.2) $f_i$ immediately disqualifies for being a $\mathrm{GC}(2, 3, 2)$-selector for $L$ (and we set $L_i = \emptyset$ in this case). Thus, $L \notin \mathrm{GC}(2, 3, 2)$. On the other hand, $L \in \mathrm{GC}(1, 2, 1)$ can be seen as follows: Given any set of inputs $X$ with $\|X\| \geq 2$, we can w.l.o.g. assume that $X \subseteq \bigcup_{i \geq 1} W_{i,4}$; since smaller strings can be solved by brute force, we may even assume that $X \subseteq W_{j,4}$ for some $j$. Suppose further that $\|L \cap X\| \geq 1$. Define $g(X) \stackrel{\text{df}}{=} X$ if $\|X\| = 2$; and if $\|X\| > 2$, define $g(X)$ to output $\{w_{j,1}, w_{j,4}\}$ if $\{w_{j,1}, w_{j,4}\} \subseteq X$, and to output $\{w_{j,2}, w_{j,3}\}$ otherwise. Since $\|L \cap \{w_{j,1}, w_{j,4}\}\| = 1$ and

$\|L \cap \{w_{j,2}, w_{j,3}\}\| = 1$ holds in each of the four cases above, it follows that $\|L \cap g(X)\| \geq 1$. Hence, $L \in \mathrm{GC}(1, 2, 1)$ via $g$.

2. For proving $\mathrm{GC}(1, 2, 1) \not\subseteq \mathrm{GC}(3, 3, 2)$, $L$ is defined as $\bigcup_{i \geq 1} L_i$, where $L_i \subseteq W_{i,5}$, and if $f_i(W_{i,5}) \subseteq W_{i,5}$ and $\|f_i(W_{i,5})\| = 3$, then we make sure that $\|L_i\| = 3$ and $\|L_i \cap f_i(W_{i,5})\| = 1$. This ensures that for no $i \geq 1$ can $f_i$ be a $\mathrm{GC}(3, 3, 2)$-selector for $L$. For example, this can be achieved by defining $L_i$ as follows:

$$
\begin{aligned}
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 01011 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,2}, w_{i,3}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 10101 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,2}, w_{i,4}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 10110 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,2}, w_{i,5}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 01101 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,3}, w_{i,4}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 01011 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,3}, w_{i,5}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 01101 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,1}, w_{i,4}, w_{i,5}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 10101 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,2}, w_{i,3}, w_{i,4}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 11010 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,2}, w_{i,3}, w_{i,5}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 10110 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,2}, w_{i,4}, w_{i,5}\},\\
\chi_L(w_{i,1}, \ldots, w_{i,5}) &= 11010 \quad \text{if} \quad f_i(W_{i,5}) = \{w_{i,3}, w_{i,4}, w_{i,5}\}.
\end{aligned}
$$

As argued above, this shows that $L \notin \mathrm{GC}(3, 3, 2)$. For proving that $L \in \mathrm{GC}(1, 2, 1)$, let a set $X$ of inputs be given and suppose w.l.o.g. that $\|X\| \geq 3$ and $X \subseteq W_{j,5}$ for some $j$. Note that for each choice of $X$, at least one of $\{w_{j,1}, w_{j,2}\}$, $\{w_{j,2}, w_{j,3}\}$, $\{w_{j,3}, w_{j,4}\}$, $\{w_{j,4}, w_{j,5}\}$, or $\{w_{j,5}, w_{j,1}\}$ must be contained in $X$. On the other hand, each of $\{w_{j,1}, w_{j,2}\}$, $\{w_{j,2}, w_{j,3}\}$, $\{w_{j,3}, w_{j,4}\}$, $\{w_{j,4}, w_{j,5}\}$, and $\{w_{j,5}, w_{j,1}\}$ has (by construction of $L$) at least one string in common with $L_j$ if $L_j$ is not set to the empty set. From these comments the action of the $\mathrm{GC}(1, 2, 1)$-selector is clear.

For proving $\mathrm{GC}(3, 3, 2) \not\subseteq \mathrm{GC}(1, 2, 1)$, define a set $L \subseteq \bigcup_{i \geq 1} W_{i,3}$ as follows:

$$
\begin{aligned}
\chi_L(w_{i,1}, w_{i,2}, w_{i,3}) &= 100 \quad \text{if} \quad f_i(W_{i,3}) = \{w_{i,2}, w_{i,3}\},\\
\chi_L(w_{i,1}, w_{i,2}, w_{i,3}) &= 010 \quad \text{if} \quad f_i(W_{i,3}) = \{w_{i,1}, w_{i,3}\},\\
\chi_L(w_{i,1}, w_{i,2}, w_{i,3}) &= 001 \quad \text{if} \quad f_i(W_{i,3}) = \{w_{i,1}, w_{i,2}\}.
\end{aligned}
$$

Since in each case $\|L \cap W_{i,3}\| = 1$ but $L \cap f_i(W_{i,3}) = \emptyset$, we clearly have $L \notin \mathrm{GC}(1, 2, 1)$. On the other hand, $L$ is easily seen to be in $\mathrm{GC}(3, 3, 2)$ via a selector that first solves all "small" inputs (i.e., those strings not of maximum length) by brute force and then outputs two small members of $L$ (and one arbitrary input) if those can be found, or three arbitrary inputs if no more than one small member of $L$ is found by brute force. Note that the $\mathrm{GC}(3, 3, 2)$-promise is not satisfied in the latter case.

Part 3 follows from Part 2, as $\mathrm{GC}(1, 2, 1) \subset \mathrm{GC}(2, 2, 1)$. □

# Appendix A

# Some Proofs from Chapter 5

**Proof of Lemma 5.4.10.**

1. Let $A = \Sigma^*$. Given $n$ distinct strings $y_1, \ldots, y_n$, define

$$f(y_1, \ldots, y_n) \stackrel{\text{df}}{=} \begin{cases} \{y_1, \ldots, y_c\} & \text{if } n \geq c \\ \{y_1, \ldots, y_n\} & \text{if } n < c. \end{cases}$$

Clearly, $f \in \mathrm{FP}$, $f(y_1, \ldots, y_n) \subseteq A$, and $\|f(y_1, \ldots, y_n)\| \leq c$. If $\|\{y_1, \ldots, y_n\} \cap A\| \geq b$, then $n \geq b$, and thus we have $\|f(y_1, \ldots, y_n) \cap A\| = c \geq d$ if $n \geq c$, and we have $\|f(y_1, \ldots, y_n) \cap A\| = n \geq b \geq d$ if $n < c$. By Definition 5.4.1, $A \in \mathrm{GC}(b, c, d)$.

2. We will define $B \stackrel{\text{df}}{=} \bigcup_{i \geq 1} B_i$ such that for no $i$ with $b + c - d + 1 \leq 2^{e(i)}$ can $f_i$ be a $\mathrm{GC}(b, c, d)$-selector for $B$. By our assumption about the enumeration of FP functions (Remark 5.4.6), this suffices. For each $i$ with $b + c - d + 1 > 2^{e(i)}$, set $B_i \stackrel{\text{df}}{=} \emptyset$. For each $i$ such that $b + c - d + 1 \leq 2^{e(i)}$, let $F_i$ and $W_i$ be shorthands for the sets $f_i(w_{i,1}, \ldots, w_{i,b+c-d+1})$ and $\{w_{i,1}, \ldots, w_{i,b+c-d+1}\}$, respectively, and let $w_{i,j_1}, \ldots, w_{i,j_{d-1}}$ be the first $d - 1$ strings in $F_i$ (if $\|F_i\| \geq d$). W.l.o.g., assume $F_i \subseteq W_i$ and $\|F_i\| \leq c$ (if not, $f_i$ automatically disqualifies for being a $\mathrm{GC}(b, c, d)$-selector). If $d \leq \|F_i\|$, then set $B_i \stackrel{\text{df}}{=} \{w_{i,j_1}, \ldots, w_{i,j_{d-1}}\} \cup (W_i - F_i)$. If $d > \|F_i\|$, then set $B_i \stackrel{\text{df}}{=} W_i$. Thus, either we have $\|W_i \cap B\| \geq (d - 1) + ((b + c - d + 1) - c) = b$ and $\|F_i \cap B\| < d$, or we have $\|W_i \cap B\| = b + c - d + 1 > b$ and $\|F_i \cap B\| < d$. Hence, $B \notin \mathrm{GC}(b, c, d)$. □

**Proof of Lemma 5.4.12.**

1. & 2. Immediate from the definitions of GC and S classes.

3. Let $l \geq n$ and $m \geq n$. By Parts 1 and 2 of this lemma and by Theorem 5.2.3, we

have

$$\begin{aligned} \mathrm{GC}(b,c,c) &= S(b,c) = S(b+n,c+n) = \mathrm{GC}(b+n,c+n,c+n) \\ &\subseteq \mathrm{GC}(b+l,c+m,c+n). \end{aligned}$$

4. Suppose $m \le l \le n$ and $L \in \mathrm{GC}(b+l, c+m, d+n)$ via $f \in \mathrm{FP}$. As in the proof of Theorem 5.2.3, let finitely many strings $z_1, \dots, z_{b+2l-1}$, each belonging to L, be hardcoded into the transducer computing function $g$ defined below. Given inputs $Y = \{y_1, \dots, y_t\}$, choose (if possible) $l$ strings $z_{i_1}, \dots, z_{i_l} \notin Y$, and define

$$g(Y) \stackrel{\mathrm{df}}{=} \begin{cases} f(Y \cup \{z_{i_1}, \dots, z_{i_l}\}) - \{u_1, \dots, u_l\} & \text{if } z_{i_1}, \dots, z_{i_l} \notin Y \text{ exist} \\ f(Y) - \{v_1, \dots, v_m\} & \text{otherwise,} \end{cases}$$

where $\{u_1, \dots, u_l\}$ contains *all* $z$-strings output by $f$, say there are $h$ with $h \le l$, the remaining $l-h$ $u$-strings are arbitrary $y$-strings of the output of $f$, and similarly, $v_1, \dots, v_m$ are arbitrary output strings of $f$. Clearly, $g \in \mathrm{FP}$ and $g(Y) \subseteq Y$. Moreover, $\|g(Y)\| \le c+m-l \le c$ if $z_{i_1}, \dots, z_{i_l} \notin Y$ exist; otherwise, we trivially have $\|g(Y)\| \le c$. Note that if $z_{i_1}, \dots, z_{i_l} \notin Y$ do not exist, then $\|Y \cap \{z_1, \dots, z_{b+2l-1}\}\| \ge b+l$. Thus, if $\|L \cap Y\| \ge b$, then either $\|L \cap (Y \cup \{z_{i_1}, \dots, z_{i_l}\})\| \ge b+l$ implies $\|L \cap g(Y)\| \ge d+n-l \ge d$, or $\|L \cap Y\| \ge b+l$ implies $\|L \cap g(Y)\| \ge d+n-m \ge d$. This establishes that $m \le l \le n$ implies $\mathrm{GC}(b+l, c+m, d+n) \subseteq \mathrm{GC}(b,c,d)$. By symmetry, we similarly obtain that $l \le m \le n$ implies $\mathrm{GC}(b+l, c+m, d+n) \subseteq \mathrm{GC}(b,c,d)$ if we exchange $l$ and $m$ in the above argument. Since ($m \le l \le n$ or $l \le m \le n$) if and only if ($l \le n$ and $m \le n$), the proof is complete. □

**Proof of Lemma 5.4.13.**

1. The diagonalization part of the proof is analogous to the proof of Lemma 5.4.10.2, the only difference being that here we have $c+q$ instead of $c$. Also, it will be useful to require that any (potential) selector $f_i$ for some set in $\mathrm{GC}(b, c+q, d)$ has the property that for any set of inputs $W$ with $\|W\| \ge c+q$, $\|f_i(W)\|$ is *exactly* $c+q$. By Remark 5.4.2, this results in an equivalent definition of the GC class and can w.l.o.g. be assumed. The construction of set $L = \bigcup_{i \ge 1} L_i$ is as follows. For each $i$ with $2^{e(i)} < b+c+q-d+1$, set $L_i \stackrel{\mathrm{df}}{=} \emptyset$. For each $i$ such that $2^{e(i)} \ge b+c-d+1$, let $F_i$ and $W_i$ be shorthands for the sets $f_i(w_{i,1}, \dots, w_{i,b+c+q-d+1})$ and $\{w_{i,1}, \dots, w_{i,b+c+q-d+1}\}$, respectively, and let $w_{i,j_1}, \dots, w_{i,j_{d-1}}$ be the first $d-1$ strings in $F_i$ (if $\|F_i\| \ge d$). If $\|F_i\| = c+q$ ($\ge d$) and

$F_i \subseteq W_i$, then set $L_i \stackrel{\text{df}}{=} \{w_{i,j_1}, \ldots, w_{i,j_{d-1}}\} \cup (W_i - F_i)$; otherwise, set $L_i \stackrel{\text{df}}{=} W_i$. As before, $L \notin \mathrm{GC}(b, c+q, d)$.

Now we prove that $L \in \mathrm{GC}(b+l, c+m, d+n)$ if $l > n$. Given any distinct input strings $y_1, \ldots, y_t$, suppose they are lexicographically ordered (i.e., $y_1 <_{\text{lex}} \cdots <_{\text{lex}} y_t$), each $y_s$ is in $W_j$ for some $j$, and $y_k <_{\text{lex}} \cdots <_{\text{lex}} y_t$ are all strings of maximum length for some $k$ with $1 \leq k \leq t$. Define a $\mathrm{GC}(b+l, c+m, d+n)$-selector $f$ for $L$ as follows:

1. For $i \in \{1, \ldots, k-1\}$, decide by brute force whether $y_i$ is in $L$. Let $\nu$ denote $\|\{y_1, \ldots, y_{k-1}\} \cap L\|$. Output $\min\{\nu, d+n\}$ strings in $L$. If $\nu \geq d+n$ then halt, otherwise go to 2.

2. If $t \geq k + (d+n-\nu) - 1$, then output $y_k, \ldots, y_{k+(d+n-\nu)-1}$; otherwise, output $y_1, \ldots, y_t$.

Clearly, $f \in \mathrm{FP}$, $f(y_1, \ldots, y_t) \subseteq \{y_1, \ldots, y_t\}$, and since $\mathrm{GC}(b+l, c+m, d+n)$ is non-trivial, we have:

$$\|f(y_1, \ldots, y_t)\| \leq \nu + (d+n-\nu) \leq c+m.$$

Now we prove that if $\|\{y_1, \ldots, y_t\} \cap L\| \geq b+l$, then $\|g(y_1, \ldots, y_t) \cap L\| \geq d+n$. Let $i$ be such that $e(i)$ is the length of $y_k, \ldots, y_t$. Clearly, if $\|F_i\| \neq c+q$, then by construction of $L$ and $f$, either $f$ outputs $d+n$ strings in $L$, or $L \cap \{y_1, \ldots, y_t\} = f(y_1, \ldots, y_t)$. Similarly, if $f$ halts in step 1 because of $\nu \geq d+n$, then we are done. So suppose $\nu < d+n$, $\|\{y_1, \ldots, y_t\} \cap L\| \geq b+l$, and $\|F_i\| = c+q \geq d$. Recall that $w_{i,j_{d-1}}$ is the $(d-1)$st string in $F_i$. Define $D \stackrel{\text{df}}{=} \{y_k, \ldots, y_t\} \cap \{w_{i,1}, \ldots, w_{i,j_{d-1}}\}$. By construction of $L$, we have $\{w_{i,1}, \ldots, w_{i,j_{d-1}}\} \subseteq L$, so $D \subseteq L$. That is,

$$\{y_k, \ldots, y_{k+\|D\|-1}\} \quad \subseteq \quad L. \tag{A.1}$$

Since $\|\{y_k, \ldots, y_t\} \cap L\| \geq b+l-\nu$, we have $t-(k-1) \geq b+l-\nu \geq d+n-\nu$, and thus $t \geq k+(d+n-\nu)-1$. This implies:

$$\{y_k, \ldots, y_{k+(d+n-\nu)-1}\} \quad \subseteq \quad f(y_1, \ldots, y_t). \tag{A.2}$$

Thus, if $d+n-\nu \leq \|D\|$, we obtain from (A.1) that $\{y_k, \ldots, y_{k+(d+n-\nu)-1}\} \subseteq L$, which in turn implies with (A.2) that $\|L \cap f(y_1, \ldots, y_t)\| \geq \nu + (d+n-\nu) = d+n$. So it remains to show that $d+n-\nu \leq \|D\|$. Observe that $b+l \leq \|\{y_1, \ldots, y_t\} \cap L\| \leq \nu + \|D\| + b - d + 1$, since $\|W_i - F_i\| = (b + c + q - d + 1) - (c + q) = b - d + 1$ (here we need that $\|F_i\| = c + q$ rather than $\|F_i\| \leq c + q$ for $f_i$ to be a GC$(b, c + q, d)$-selector). Thus, $v + \|D\| + b - d + 1 \geq b + l$. By the assumption that $l \geq n + 1$, we obtain $d + n - v \leq \|D\|$.

Parts 2 and 3 can be proven by a similar technique. $\square$

# Index

# Bibliography

[ABG90] A. Amir, R. Beigel, and W. Gasarch. Some connections between bounded query classes and non-uniform complexity. *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 232–243. IEEE Computer Society Press, 1990.

[AH92] E. Allender and L. Hemachandra. Lower bounds for the low hierarchy. *Journal of the Association for Computing Machinery*, 39(1):234–251, 1992.

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

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

[All91] E. Allender. Limitations of the upward separation technique. *Mathematical Systems Theory*, 24(1):53–67, 1991.

[AR88] E. Allender and R. Rubinstein. P-printable sets. *SIAM Journal on Computing*, 17:1193–1202, 1988.

[AW90] E. Allender and C. Wilson. Downward translations of equality. *Theoretical Computer Science*, 75(3):335–346, 1990.

[BBJ$^+$89] A. Bertoni, D. Bruschi, D. Joseph, M. Sitharam, and P. Young. Generalized boolean hierarchies and boolean hierarchies over RP. In *Proceedings of the 7th Conference on Fundamentals of Computation Theory*, pages 35–46. Springer-Verlag *Lecture Notes in Computer Science #380*, August 1989.

[BBS86a] J. Balcázar, R. Book, and U. Schöning. The polynomial-time hierarchy and sparse oracles. *Journal of the ACM*, 33(3):603–617, 1986.

[BBS86b] J. Balcázar, R. Book, and U. Schöning. Sparse sets, lowness and highness. *SIAM Journal on Computing*, 15(3):739–746, 1986.

[BCO93] R. Beigel, R. Chang, and M. Ogiwara. A relationship between difference hierarchies and relativized polynomial hierarchies. *Mathematical Systems Theory*, 26:293–310, 1993.

[Bei89] R. Beigel. On the relativized power of additional accepting paths. In *Proceedings of the 4th Structure in Complexity Theory Conference*, pages 216–224. IEEE Computer Society Press, June 1989.

[Bei92] R. Beigel. Perceptrons, PP, and the polynomial hierarchy. In *Proceedings of the 7th IEEE Conference on Structure in Complexity Theory*, pages 14–19. IEEE Computer Society Press, 1992.

[BG82] A. Blass and Y. Gurevich. On the unique satisfiability problem. *Information and Control*, 55:80–88, 1982.

[BG92] R. Beigel and J. Gill. Counting classes: Thresholds, parity, mods, and fewness. *Theoretical Computer Science*, 103(1):3–23, 1992.

[BG94] R. Beigel and J. Goldsmith. Downward separation fails catastrophically for limited nondeterminism classes. In *Proceedings of the 9th Structure in Complexity Theory Conference*, pages 134–138. IEEE Computer Society Press, June/July 1994.

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

[BJY90] D. Bruschi, D. Joseph, and P. Young. Strong separations for the boolean hierarchy over RP. *International Journal of Foundations of Computer Science*, 1(3):201–218, 1990.

[BKS94] R. Beigel, M. Kummer, and F. Stephan. Approximable sets. In *Proceedings of the 9th IEEE Conference on Structure in Complexity Theory*, pages 12–23. IEEE Computer Society Press, 1994.

[BHL] H. Buhrman, E. Hemaspaandra, and L. Longpré. SPARSE reduces conjunctively to TALLY. *SIAM Journal on Computing*, in press. A preliminary version appeared as: H. Buhrman, L. Longpré, and E. Spaan. SPARSE reduces conjunctively to TALLY. In *Proceedings of the 8th IEEE Conference on Structure in Complexity Theory*, pages 208–214. IEEE Computer Society Press, 1993.

[Boo74] R. Book. Tally languages and complexity classes. *Information and Control*, 26:186–193, 1974.

[BvHT93] H. Buhrman, P. van Helden, and L. Torenvliet. P-selective self-reducible sets: A new characterization of P. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 44–51. IEEE Computer Society Press, May 1993.

[Cai87] J. Cai. Probability one separation of the Boolean hierarchy. In *Proceedings of the 4th Annual Symposium on Theoretical Aspects of Computer Science*, pages 148–158. Springer-Verlag *Lecture Notes in Computer Science #247*, 1987.

[CGH$^+$88] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung. The boolean hierarchy I: Structural properties. *SIAM Journal on Computing*, 17(6):1232–1252, 1988.

[CGH$^+$89] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung. The boolean hierarchy II: Applications. *SIAM Journal on Computing*, 18(1):95–111, 1989.

[CH85] J. Cai and L. Hemachandra. The boolean hierarchy: Hardware over NP. Technical Report 85-724, Cornell University, Department of Computer Science, Ithaca, NY, December 1985.

[CH90] J. Cai and L. Hemachandra. On the power of parity polynomial time. *Mathematical Systems Theory*, 23:95–106, 1990.

[Cha91] R. Chang. *On the Structure of NP Computations under Boolean Operators*. PhD thesis, Cornell University, Ithaca, NY, 1991.

[CHV93] J. Cai, L. Hemachandra, and J. Vyskoč. Promises and fault-tolerant database access. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory*, pages 101–146. Cambridge University Press, 1993.

[CK90a] R. Chang and J. Kadin. The Boolean hierarchy and the polynomial hierarchy: A closer connection. In *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 169–178. IEEE Computer Society Press, July 1990.

[CK90b] R. Chang and J. Kadin. On computing Boolean connectives of characteristic functions. Technical Report TR 90-1118, Department of Computer Science, Cornell University, Ithaca, NY, May 1990.

[CKS81] A. Chandra, D. Kozen, and L. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[CM87] J. Cai and G. Meyer. Graph minimal uncolorability is DP-complete. *SIAM Journal on Computing*, 16(2):259–277, 1987.

[Cob64] A. Cobham. The intrinsic computational difficulty of functions. In *Proceedings of the 1964 International Congress for Logic, Methodology, and Philosophy of Science*, pages 24–30, North Holland, 1964.

[Coo71] S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[Cro94] K. Cronauer. A criterion to separate complexity classes by oracles. Technical Report 76, Universität Würzburg, Institut für Informatik, Würzburg, Germany, January 1994.

[DHHT94] D. Derek-Brown, Y. Han, L. Hemaspaandra, and L. Torenvliet. Semi-membership algorithms: Some recent advances. *SIGACT News*, 25(3):12–23, 1994.

[Edm65] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[ESY84] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.

[EY80] S. Even and Y. Yacobi. Cryptocomplexity and NP-completeness. In *Proceedings of the 7th International Colloquium on Automata, Languages, and Programming*, pages 195–207. Springer-Verlag *Lecture Notes in Computer Science*, 1980.

[FFK91] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. In *Proceedings of the 6th IEEE Conference on Structure in Complexity Theory*, pages 30–42. IEEE Computer Society Press, 1991.

[FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48:116–148, 1994.

[FFL93] S. Fenner, L. Fortnow, and L. Li. Gap-definability as a closure property. In *Proceedings of the 10th Annual Symposium on Theoretical Aspects of Computer Science*, pages 484–493. Springer-Verlag *Lecture Notes in Computer Science*, 1993.

[FR91] L. Fortnow and N. Reingold. PP is closed under truth-table reductions. In *Proceedings of the 6th IEEE Conference on Structure in Complexity Theory*, pages 13–15. IEEE Computer Society Press, 1991.

[Gil77] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6(4):675–695, 1977.

[GNW90] T. Gundermann, N. Nasser, and G. Wechsung. A survey on counting classes. In *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 140–153. IEEE Computer Society Press, July 1990.

[GP86] L. Goldschlager and I. Parberry. On the construction of parallel computers from various bases of boolean functions. *Theoretical Computer Science*, 43:43–58, 1986.

[GS88] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.

[Gup91] S. Gupta. The power of witness reduction. In *Proceedings of the 6th IEEE Conference on Structure in Complexity Theory*, pages 43–59. IEEE Computer Society Press, 1991.

[Gup93] S. Gupta. On bounded-probability operators and $\mathrm{C}_{=}\mathrm{P}$. *Information Processing Letters*, 48:93–98, 1993.

[GW86] T. Gundermann and G. Wechsung. Nondeterministic Turing machines with modified acceptance. In *Proceedings of the 11th International Symposium on Mathematical Foundations of Computer Sciences*, pages 396–404. Springer-Verlag *Lecture Notes in Computer Science #233*, 1986.

[GW87] T. Gundermann and G. Wechsung. Counting classes with finite acceptance types. *Computers and Artificial Intelligence*, 6(5):395–409, 1987.

[Har83] J. Hartmanis. On sparse sets in NP $-$ P. *Information Processing Letters*, 16:55–60, 1983.

[Hau14] F. Hausdorff. *Grundzüge der Mengenlehre*. Leipzig, 1914.

[Hem87] L. A. Hemachandra. Counting in Structural Complexity Theory. *Ph.D. Thesis*, Cornell University, Ithaca, NY, 1987.

[Hem93] L. Hemaspaandra. Lowness: a yardstick for NP $-$ P. *SIGACT News*, 24(2):10–14, 1993.

[Hem94] L. A. Hemachandra. Personal communication, November 27, 1994.

[Her90] U. Hertrampf. Relations among Mod-classes. *Theoretical Computer Science*, 74(3):325–328, 1990.

[HH88] J. Hartmanis and L. Hemachandra. Complexity classes without machines: On complete languages for UP. *Theoretical Computer Science*, 58:129–142, 1988.

[HH90] J. Hartmanis and L. Hemachandra. Robust machines accept easy sets. *Theoretical Computer Science*, 74(2):217–226, 1990.

[HH91] J. Hartmanis and L. Hemachandra. One-way functions and the nonisomorphism of NP-complete sets. *Theoretical Computer Science*, 81:155–163, 1991.

[HH94] E. Hemaspaandra and L. Hemaspaandra. Quasi-injective reductions. *Theoretical Computer Science*, 123:407–413, 1994.

[HHO$^+$] L. Hemachandra, A. Hoene, M. Ogiwara, A. Selman, T. Thierauf, and J. Wang. Selectivity. In *Proceedings of the 5th International Conference on Computing and Information*, pages 55–59, IEEE Computer Society Press, 1993.

[HHSY91] L. Hemachandra, A. Hoene, D. Siefkes, P. Young. On sets polynomially enumerable by iteration. *Theoretical Computer Science*, 80:203–225, 1991.

[HHT93] Y. Han, L. Hemachandra, and T. Thierauf. Threshold computation and cryptographic security. In *Proceedings of the 4th International Symposium on Algorithms and Computation*, pages 230–239. Springer-Verlag *Lecture Notes in Computer Science #762*, December 1993.

[HIS85] J. Hartmanis, N. Immerman, and V. Sewelson. Sparse sets in NP$-$P: EXPTIME versus NEXPTIME. *Information and Control*, 65(2/3):159–181, 1985.

[HJ93] L. Hemachandra and S. Jha. Defying upward and downward separation. In *Proceedings of the 10th Annual Symposium on Theoretical Aspects of Computer Science*, pages 185–195. Springer-Verlag *Lecture Notes in Computer Science #665*, February 1993. To appear in *Information and Computation*.

[HJ] L. Hemaspaandra and Z. Jiang. P-Selectivity: Intersections and Indices. *Theoretical Computer Science*, to appear. Available as Univ. of Rochester Department of Computer Science Technical Report TR-482.

[HJRW95] L. Hemaspaandra, Z. Jiang, J. Rothe, and O. Watanabe. Multi-selectivity and complexity-lowering joins. Technical Report TR 568, University of Rochester, Rochester, NY, January 1995.

[HJV93] L. Hemaspaandra, S. Jain, and N. Vereshchagin. Banishing robust Turing completeness. *International Journal of Foundations of Computer Science*, 4(3):245–265, 1993.

[HNOS94] L. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. Computing solutions uniquely collapses the polynomial hierarchy. *SIAM Journal on Computing*, to appear. A preliminary version appeared in *Proc. 5th ISAAC*, pages 56–64, 1994.

[Hop81] J. Hopcroft. Recent directions in algorithmic research. In *Proceedings 5th GI Conference on Theoretical Computer Science*, pages 123–134. Springer-Verlag *Lecture Notes in Computer Science #104*, 1981.

[HOW92] L. Hemachandra, M. Ogiwara, O. Watanabe. How hard are sparse sets? In *Proceedings of the 7th Structure in Complexity Theory Conference*, pages 222–238. IEEE Computer Society Press, June 1992.

[HR92] L. Hemachandra and R. Rubinstein. Separating complexity classes with tally oracles. *Theoretical Computer Science*, 92(2):309–318, 1992.

[HR95] L. Hemaspaandra and J. Rothe. Intersection suffices for Boolean hierarchy equivalence. In *Proceedings of the First Annual International Computing and Combinatorics Conference*, pages 430–435. Springer-Verlag *Lecture Notes in Computer Science #959*, August 1995.

[HR] L. Hemaspaandra and J. Rothe. Unambiguous computation: Boolean hierarchies and sparse Turing-complete sets. *SIAM Journal on Computing*, to appear. Available as: Technical Report TR 483, University of Rochester, Rochester, NY, January 1994.

[HRW95] L. Hemaspaandra, J. Rothe, and G. Wechsung. Easy sets and hard certificate schemes. Technical Report Math/95/5, Institut für Informatik, Friedrich-Schiller-Universität Jena, Jena, Germany, May 1995. Also available as: Technical Report TR 585, University of Rochester, Rochester, NY, May 1995.

[HS65] J. Hartmanis and R. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[HY84] J. Hartmanis and Y. Yesha. Computation times of NP sets of different densities. *Theoretical Computer Science*, 34:17–32, 1984.

[HZ93] L. Hemaspaandra and M. Zimand. Strong forms of balanced immunity. *Mathematical Systems Theory*, to appear. Available as: Technical Report TR 480, Department of Computer Science, University of Rochester, Rochester, NY, December 1993.

[IT89] R. Impagliazzo and G. Tardos. Decision versus search problems in super-polynomial time. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 222–227. IEEE Computer Society Press, October/November 1989.

[Joc68] C. Jockusch. Semirecursive sets and positive reducibility. *Transactions of the AMS*, 131(2):420–436, 1968.

[Kad88] J. Kadin. The polynomial time hierarchy collapses if the Boolean hierarchy collapses. *SIAM Journal on Computing*, 17(6):1263–1282, 1988. Erratum appears in the same journal, 20(2):404.

[Kad89] J. Kadin. $\mathrm{P}^{\mathrm{NP}[\log n]}$ and sparse Turing-complete sets for NP. *Journal of Computer and System Sciences*, 39(3):282–298, 1989.

[KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th ACM Symposium on Theory of Computing*, pages 302–309, April 1980. An extended version has also appeared as: Turing machines that take advice. *L'Enseignement Mathématique*, 2nd series, 28:191–209, 1982.

[Ko83] K. Ko. On self-reducibility and weak P-selectivity. *Journal of Computer and System Sciences*, 26:209–221, 1983.

[Ko91] K. Ko. Separating the low and the high hierarchies by oracles. *Information and Computation*, 90:156–177, 1991.

[Köb94] J. Köbler. Locating P/poly optimally in the extended low hierarchy. *Theoretical Computer Science*, 134:263–285, 1994.

[Köb95] J. Köbler. On the structure of low sets. In *Proceedings of the 10th Structure in Complexity Theory Conference*, pages 246–261. IEEE Computer Society Press, 1995.

[KS85] K. Ko and U. Schöning. On circuit-size complexity and the low hierarchy in NP. *SIAM Journal on Computing*, 14(1):41–51, 1985.

[KST92] J. Köbler, U. Schöning, and J. Torán. Graph isomorphism is low for PP. *Computational Complexity*, 2:301–330, 1992.

[KST⁺93] J. Köbler, U. Schöning, J. Torán, and S. Toda. Turing machines with few accepting computations and low sets for PP. *Journal of Computer and System Sciences*, 44:272–286, 1992.

[KSW87] J. Köbler, U. Schöning, and K. Wagner. The difference and truth-table hierarchies for NP. *R.A.I.R.O. Informatique théorique et Applications*, 21:419–435, 1987.

[KW95] J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. In *Proceedings of the 22nd International Colloquium on Automata, Languages and Programming*, 1995, to appear. Available as: Technical Report 94-11, Institut für Informatik, Universität Ulm, Ulm, Germany, November 1994.

[Lev73] L. Levin. Universal sorting problems. *Problems of Information Transmission*, 9:265–266, 1973.

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

[LR94] K.-J. Lange and P. Rossmanith. Unambiguous polynomial hierarchies and exponential size. In *Proceedings of the 9th Structure in Complexity Theory Conference*, pages 106–115. IEEE Computer Society Press, June/July 1994.

[LS86] T. Long and A. Selman. Relativizing complexity classes with sparse oracles. *Journal of the ACM*, 33(3):618–627, 1986.

[LS94] T. Long and M. Sheu. A refinement of the low and high hierarchies. *SIAM Journal on Computing*, 23(3):488–509, 1994.

[MP79] A. Meyer and M. Paterson. With what frequency are apparently intractable problems difficult? Technical Report MIT/LCS/TM-126, MIT Laboratory for Computer Science, Cambridge, MA, 1979.

[MS72] A. Meyer and L. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129, 1972.

[Nic94] A. Nickelsen, 1994. Personal communication.

[NR93] R. Niedermeier and P. Rossmanith. Extended locally definable acceptance types. In *Proceedings of the 10th Annual Symposium on Theoretical Aspects of Computer Science*, pages 473–483. Springer-Verlag *Lecture Notes in Computer Science #665*, February 1993, 473–483.

[Ogi94] M. Ogihara. Polynomial-time membership comparable sets. In *Proceedings of the 9th Structure in Complexity Theory Conference*, pages 2–11. IEEE Computer Society Press, 1994.

[OH90] M. Ogiwara and L. Hemachandra. A complexity theory for feasible closure properties. Technical Report C-99, Tokyo Institut of Technology, Department of Information Sciences, Tokyo, Japan, October 1990; also: *Journal of Computer and System Sciences*, 46(3):295–325, 1993.

[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.

[PY84] C. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.

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

[Rac82] C Rackoff. Relativized questions involving probabilistic algorithms. *Journal of the ACM*, 29(1):261–268, 1982.

[Rao94] R. Rao, 1994. Personal communication.

[Reg89] K. Regan. Provable complexity properties and constructive reasoning. Manuscript, April 1989.

[Rot95] J. Rothe. A promise class at least as hard as the polynomial hierarchy. *Journal on Computing and Information*, 1(1):92–107, 1995. Special Issue: *Proceedings of the 6th International Conference on Computing and Information*, May 1994.

[RR91] K. Regan and J. Royer. On closure properties of bounded 2-sided error complexity classes. Manuscript, 1991, to appear in *Mathematical Systems Theory*.

[RRW94] R. Rao, J. Rothe, and O. Watanabe. Upward separation for FewP and related classes. *Information Processing Letters*, 52(4):175–180, 1994.

[RV92] J. Rothe and J. Vogel. A note on the polynomial-time hierarchy and probabilistic operators. *Computers and Artificial Intelligence*, 13(1):2–12, 1994.

[Sal93] S. Saluja. Relativized limitations of left set technique and closure classes of sparse sets. In *Proceedings of the 8th IEEE Conference on Structure in Complexity Theory*, pages 215–222. IEEE Computer Society Press, 1993.

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

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

[Sch88] U. Schöning. Graph isomorphism is in the low hierarchy. *Journal of Computer and System Sciences*, 37:312–323, 1988.

[Sch89] U. Schöning. Probabilistic complexity classes and lowness. *Journal of Computer and System Sciences*, 39(1):84–100, 1989.

[Sch91] U. Schöning. Recent highlights in structural complexity theory. Technical Report TR 6-91, Universität Ulm, Ulm, Germany, 1991.

[Sch93] U. Schöning. On random reductions from sparse sets to tally sets. *Information Processing Letters*, 46:239–241, 1993.

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP. *Mathematical Systems Theory*, 13:55–65, 1979.

[Sel88] A. Selman. Natural self-reducible sets. *SIAM Journal on Computing*, 17(5):989–996, 1988.

[Sel88] A. Selman. Promise problems complete for complexity classes. *Information and Computation*, 78:87–98, 1988.

[Sim75] J. Simon. On Some Central Problems in Computational Complexity. *Ph.D. Thesis*, Cornell University, Ithaca, NY, 1975.

[SL92] M. Sheu and T. Long. UP and the low and high hierarchies: a relativized separation. *Mathematical Systems Theory*, to appear. Ohio State University Technical Report OSU-CISRC-3/92-TR-9, 1992.

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

[Tar91] J. Tarui. Randomized polynomials, threshold circuits, and the polynomial hierarchy. In *Proceedings of the 8th Symposium on Theoretical Aspects of Computer Science*, pages 238–250. Springer-Verlag *Lecture Notes in Computer Science #480*, 1991.

[TO92] S. Toda and M. Ogiwara. Counting classes are at least as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 21(2):316–328, 1992.

[Tod91] S. Toda. PP is as hard as the polynomial time hierarchy. *SIAM Journal on Computing*, 20:865–877, 1991.

[Tod91] S. Toda. Computational Complexity of Counting Complexity Classes. *Ph.D. Thesis*, Tokyo Institute of Technology, Tokyo, Japan, June 1991.

[Tod91] S. Toda. On polynomial-time truth-table reducibilities of intractable sets to P-selective sets. *Mathematical Systems Theory*, 24:69–82, 1991.

[Tor88] J. Torán. Structural Properties of Counting Hierarchies. *Ph.D. 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.

[Val79] L. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

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

[Wag86] K. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23:325–356, 1986.

[Wat88] O. Watanabe. On hardness of one-way functions. *Information Processing Letters*, 27:151–157, 1988.

[Wat91] O. Watanabe. On intractability of the class UP. *Mathematical Systems Theory*, 24:1–10, 1991.

[Wat92] O. Watanabe. On polynomial-time one-truth-table reducibility to a sparse set. *Journal of Computer and System Sciences*, 44(3):500–516, 1992.

[Wec85] G. Wechsung. On the Boolean closure of NP. In *Proceedings of the 1985 International on Conference on Fundamentals of Computation Theory*, pages 485–493. Springer-Verlag *Lecture Notes in Computer Science*, 1985. (An unpublished precursor of this paper was coauthored by K. W. Wagner.)

[You92] P. Young. How reductions to sparse sets collapse the polynomial-time hierarchy: A primer. *SIGACT News*, 1992. Part I (#3, pages 107–117), Part II (#4, pages 83–94), and Corrigendum to Part I (#4, page 94).

## Biographical Sketch

| | |
|---|---|
| | Jörg Rothe |
| 11/1/1966 | born in Erfurt, Germany |
| 1973–1981 | comprehensive school in Erfurt, Germany |
| 1981–1985 | extended highschool in Ilmenau, Germany, with advanced specialization in mathematics and physics |
| 1985 | Abitur |
| 1985–1986 | employment as a male nurse at the County Mental Hospital Hildburghausen |
| 1986–1991 | studies in mathematics and computer science at the Friedrich-Schiller-Universität Jena |
| 1991 | Diploma in mathematics |
| since 1991 | graduate student, employed at the Institut für Informatik of the Friedrich-Schiller-Universität Jena |
| 1993–1994 | fellowship of the German Academic Exchange Service (DAAD) for a year-long research visit at the University of Rochester Department of Computer Science |

Jena, Germany
July, 1995