SIAM J. COMPUT.
Vol. 26, No. 3, pp. 634–653, June 1997

002

UNAMBIGUOUS COMPUTATION: BOOLEAN HIERARCHIES AND SPARSE TURING-COMPLETE SETS*

LANE A. HEMASPAANDRA† AND JÖRG ROTHE‡

Abstract. It is known that for any class $\mathcal{C}$ closed under *union and intersection*, the Boolean closure of $\mathcal{C}$, the Boolean hierarchy over $\mathcal{C}$, and the symmetric difference hierarchy over $\mathcal{C}$ all are equal. We prove that these equalities hold for any complexity class closed under *intersection*; in particular, they thus hold for unambiguous polynomial time (UP). In contrast to the NP case, we prove that the Hausdorff hierarchy and the nested difference hierarchy over UP both fail to capture the Boolean closure of UP in some relativized worlds.

Karp and Lipton proved that if *nondeterministic* polynomial time has sparse Turing-complete sets, then the polynomial hierarchy collapses. We establish the first consequences from the assumption that *unambiguous* polynomial time has sparse Turing-complete sets: (a) UP $\subseteq \mathrm{Low}_2$, where Low_2 is the second level of the low hierarchy, and (b) each level of the unambiguous polynomial hierarchy is contained one level lower in the promise unambiguous polynomial hierarchy than is otherwise known to be the case.

Key words. unambiguous computation, Boolean hierarchy, sparse Turing-complete sets

AMS subject classifications. 68Q15, 68Q10, 03D15

PII. S0097539794261970

1. Introduction.

NP and NP-based hierarchies—such as the polynomial hierarchy [47, 57] and the Boolean hierarchy over NP [9, 10, 41]—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, we have the following: though the polynomial hierarchy possesses downward separation (that is, if its low levels collapse, then all its levels collapse) [47, 57], downward separation does not hold "robustly" (i.e., in every relativized world) for the exponential time hierarchy [24, 36] or for limited-nondeterminism hierarchies [32] (see also [4]). As examples regarding UP, we have the following: NP has $\leq_m^p$-complete sets, but UP does not robustly possess $\leq_m^p$-complete sets [22] or even $\leq_T^p$-complete sets [31]; NP positively relativizes, in the sense that it collapses to P if and only if it does so with respect to every tally oracle [45] (see also [1]), but UP does not robustly positively relativize [29]; NP has "constructive programming systems," but UP does not robustly have such systems [52]; NP (actually, nondeterministic computation) admits time hierarchy theorems [25], but it is an open question whether unambiguous computation has nontrivial time hierarchy theorems; NP displays upward separation (that is, NP − P contains sparse sets if and

* Received by the editors January 24, 1994; accepted for publication June 7, 1995.
http://www.siam.org/journals/sicomp/26-3/26197.html

† Department of Computer Science, University of Rochester, Rochester, NY 14627 (lane@cs.rochester.edu). The research of this author was supported in part by NSF grants CCR-8957604, INT-9116781/JSPS-ENGR-207, CCR-9322513, and INT-9513368/DAAD-315-PRO-of-ab and an NAS/NRC COBASE grant.

‡ Institut für Informatik, Friedrich-Schiller-Universität Jena, 07743 Jena, Germany (rothe@informatik.uni-jena.de). The research of this author was supported in part by a grant from the DAAD and NSF grants CCR-8957604 and INT-9513368/DAAD-315-PRO-of-ab and was done in part while visiting the University of Rochester.

only if NE $\neq$ E) [24], but it is not known whether UP does (see [32], which shows that R and BPP do not robustly display upward separation, and [51], which shows that FewP does 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 this paper, 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, which has generated quite a bit of interest and the collapse of which is known to imply the collapse of the polynomial hierarchy [37, 16, 3], a large number of definitions are known to be equivalent. For example, for NP, all the following coincide [9]: 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 [41]. In fact, these equalities are known to hold for all classes that contain Σ^* and $\emptyset$ and are closed under union and intersection [26, 9, 41, 5, 20, 15, 14]. In section 3, 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 Σ^* and $\emptyset$ and closed under intersection (e.g., UP, US, and DP, first defined, respectively, in [59], [6], and [50] and each of which is not currently known to be closed under union): $\mathrm{SDH}(\mathcal{K}) = \mathrm{CH}(\mathcal{K}) = \mathrm{BC}(\mathcal{K})$. However, for the remaining two hierarchies, we show that not all classes containing Σ^* 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 robustly capture the Boolean closure of UP. 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.

It is well known, thanks to the work of Karp and Lipton [39] (see also the related references given in section 4), that if NP has sparse Turing-hard sets, then the polynomial hierarchy collapses. 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 (which was introduced recently by Niedermeier and Rossmanith [48]). Section 4 studies this issue and shows that if UP has sparse Turing-complete sets, then the levels of the unambiguous polynomial hierarchy "slip down" slightly in terms of their location within the promise unambiguous polynomial hierarchy (a version of the unambiguous polynomial hierarchy that requires only that computations *actually executed* be unambiguous), i.e., the kth level of the unambiguous polynomial hierarchy is contained in the $(k-1)$st level of the promise unambiguous polynomial hierarchy. Various related results are also established. For example, if UP has Turing-hard sparse sets, then (a) UP $\subseteq \mathrm{Low}_2$, where Low_2 is the second level of the low hierarchy [53], and (b) the kth level of the unambiguous polynomial hierarchy can be accepted via a deterministic polynomial-time Turing transducer given access to both a Σ_2^p set and the $(k-1)$st level of the promise unambiguous polynomial hierarchy.

2. Notation. In general, we adopt the standard notations of Hopcroft and Ullman [35]. Fix the alphabet $\Sigma = \{0, 1\}$. Σ^* is the set of all strings over Σ. For each

string $u \in \Sigma^*$, $|u|$ denotes the length of u. The empty string is denoted by ϵ. For each set $L \subseteq \Sigma^*$, $\|L\|$ denotes the cardinality of L and $\overline{L} = \Sigma^* - L$ denotes the complement of L. $L^{=n}$ ($L^{\leq n}$) is the set of all strings in L having length n (less than or equal to n). Let Σ^n and $\Sigma^{\leq n}$ be shorthands for $(\Sigma^*)^{=n}$ and $(\Sigma^*)^{\leq n}$, respectively. A set S is said to be *sparse* if there is a polynomial q such that for every $m \geq 0$, $\|S^{\leq m}\| \leq q(m)$. To encode a pair of strings, we use a polynomial-time computable pairing function, $\langle \cdot, \cdot \rangle : \Sigma^* \times \Sigma^* \to \Sigma^*$, that has polynomial-time computable inverses; this notion is extended to encode every k-tuple of strings, in the standard way. Let $\leq_{\text{lex}}$ denote the standard quasi-lexicographical ordering on Σ^*, that is, for strings x and y, $x \leq_{\text{lex}} y$ if either $x = y$, or $|x| < |y|$, or ($|x| = |y|$ and there exists some $z \in \Sigma^*$ such that $x = z0u$ and $y = z1v$). $x <_{\text{lex}} y$ indicates that $x \leq_{\text{lex}} y$ but $x \neq y$.

For sets A and B, their join, $A \oplus B$, is $\{0x \,|\, x \in A\} \cup \{1x \,|\, x \in B\}$, and their symmetric difference, $A \,\Delta\, B$, is $(A - B) \cup (B - A)$. For any class $\mathcal{C}$, define $\text{co}\mathcal{C} \stackrel{\text{df}}{=} \{L \,|\, \overline{L} \in \mathcal{C}\}$, and let $\text{BC}(\mathcal{C})$ denote the Boolean algebra generated by $\mathcal{C}$, i.e., the smallest class containing $\mathcal{C}$ and closed under all Boolean operations. For any classes $\mathcal{A}$ and $\mathcal{B}$, let $\mathcal{A} \oplus \mathcal{B}$ denote the class $\{A \oplus B \,|\, A \in \mathcal{A} \wedge B \in \mathcal{B}\}$. Similarly, for classes $\mathcal{C}$ and $\mathcal{D}$ of sets, define

$$\mathcal{C} \wedge \mathcal{D} \stackrel{\text{df}}{=} \{A \cap B \,|\, A \in \mathcal{C} \wedge B \in \mathcal{D}\}, \qquad \mathcal{C} \;\boldsymbol{\Delta}\; \mathcal{D} \stackrel{\text{df}}{=} \{A \,\Delta\, B \,|\, A \in \mathcal{C} \wedge B \in \mathcal{D}\},$$
$$\mathcal{C} \vee \mathcal{D} \stackrel{\text{df}}{=} \{A \cup B \,|\, A \in \mathcal{C} \wedge B \in \mathcal{D}\}, \qquad \mathcal{C} - \mathcal{D} \stackrel{\text{df}}{=} \{A - B \,|\, A \in \mathcal{C} \wedge B \in \mathcal{D}\}.$$

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 [59]. UP is the class of all languages that are accepted by some UPM [59]. 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 [23] and thus create a polynomial hierarchy analogue that is completely contained in a low level of the polynomial hierarchy (Allender and Hemachandra as cited in [29]). So, 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 Turing machine M, $L(M)$ denotes the set of strings accepted by M, and the notation $M(x)$ means "M on input x." For any oracle Turing machine 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 ℓ_n such that, for every x of length n, every path of $M(x)$ is of length ℓ_n, and furthermore, in the case of oracle machines, that ℓ_n is independent of the oracle. Let A and B be sets. We say A is *Turing reducible* to B (denoted by $A \leq_T^p B$ or $A \in \text{P}^B$) if there is a DPOM M such that $A = L(M^B)$. 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 $B \in \mathcal{C}$.

3. Boolean hierarchies over classes closed under intersection. The Boolean hierarchy is a natural extension of the classes NP [17, 44] and DP $\stackrel{\text{df}}{=}$ NP $\wedge$ coNP [50]. 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 [13]. Note

that DP clearly is closed under intersection but is not closed under union unless the polynomial hierarchy collapses (due to [37]; see also [15, 14]).

DEFINITION 3.1. [9, 41, 26] *Let $\mathcal{K}$ be any class of sets.*

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

$$\mathrm{C}_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K}, \quad \mathrm{C}_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \begin{cases} \mathrm{C}_{k-1}(\mathcal{K}) \vee \mathcal{K} & \text{if } k \text{ is odd,} \\ \mathrm{C}_{k-1}(\mathcal{K}) \wedge \mathrm{co}\mathcal{K} & \text{if } k \text{ is even,} \end{cases} \quad k \geq 2, \quad \mathrm{CH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k \geq 1} \mathrm{C}_k(\mathcal{K}).$$

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

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

3. *The* Hausdorff ("union of differences") hierarchy over $\mathcal{K}$:[1]

$$\mathrm{E}_1(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K}, \ \mathrm{E}_2(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathcal{K} - \mathcal{K}, \ \mathrm{E}_k(\mathcal{K}) \stackrel{\mathrm{df}}{=} \mathrm{E}_2(\mathcal{K}) \vee \mathrm{E}_{k-2}(\mathcal{K}), \ k > 2, \ \mathrm{EH}(\mathcal{K}) \stackrel{\mathrm{df}}{=} \bigcup_{k \geq 1} \mathrm{E}_k(\mathcal{K}).$$

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

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

It is easily seen that for any X chosen from {C, D, E, SD}, if $\mathcal{K}$ contains $\emptyset$ and Σ^*, 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. *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. $\mathrm{D}_{2n}(\mathcal{K}) = \mathrm{C}_{2n}(\mathrm{co}\mathcal{K})$.

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

$$\begin{aligned} \mathrm{D}_{2n+1}(\mathcal{K}) &= \mathcal{K} \wedge (\mathrm{co}\mathcal{K} \vee \mathrm{D}_{2n-1}(\mathcal{K})) \stackrel{\text{hyp.}}{=} \mathcal{K} \wedge (\mathrm{co}\mathcal{K} \vee \mathrm{coC}_{2n-1}(\mathrm{co}\mathcal{K})) \\ &= \mathcal{K} \wedge \mathrm{co}(\mathcal{K} \wedge \mathrm{C}_{2n-1}(\mathrm{co}\mathcal{K})) = \mathcal{K} \wedge \mathrm{coC}_{2n}(\mathrm{co}\mathcal{K}) \\ &= \mathrm{co}(\mathrm{co}\mathcal{K} \vee \mathrm{C}_{2n}(\mathrm{co}\mathcal{K})) = \mathrm{coC}_{2n+1}(\mathrm{co}\mathcal{K}) \end{aligned}$$

shows part 1 for $n+1$, and

$$\mathrm{D}_{2n+2}(\mathcal{K}) = \mathcal{K} - (\mathcal{K} - \mathrm{D}_{2n}(\mathcal{K})) \stackrel{\text{hyp.}}{=} \mathcal{K} \wedge (\mathrm{co}\mathcal{K} \vee \mathrm{C}_{2n}(\mathrm{co}\mathcal{K})) = \mathrm{C}_{2n+2}(\mathrm{co}\mathcal{K})$$

shows part 2 for $n+1$. □

COROLLARY 3.3. $\mathrm{CH(UP)} = \mathrm{coCH(UP)} = \mathrm{DH(coUP)}$ *and* $\mathrm{CH(coUP)} = \mathrm{coCH(coUP)} = \mathrm{DH(UP)}$.

[1] Hausdorff hierarchies [26] (see [9, 5, 20], respectively, for applications to NP, R, and ~~G~~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 [26] (see also [9]). In this paper, since UP, for example, is not known to be closed under union, the distinction is nontrivial.

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 in this paper, UP. However, many apply to any nontrivial class (i.e., any class containing Σ^* and $\emptyset$) closed under intersection (see Theorem 3.10). Although it has been proven in [9] and [41] that all the standard normal forms of Definition 3.1 coincide for NP,[2] the situation for UP seems to be different since 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 [41] 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 3.4. SDH(UP) = BC(UP).

Proof. The inclusion from left to right is clear. For the converse inclusion, it is sufficient to show that SDH(UP) is closed under all Boolean operations since BC(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 SDH(UP). Then for some $k, \ell \geq 1$, there are sets $A_1, \ldots, A_k, B_1, \ldots, B_\ell$ in UP representing L and L':

$$L = A_1 \,\Delta\, \cdots \,\Delta\, A_k \quad \text{and} \quad L' = B_1 \,\Delta\, \cdots \,\Delta\, B_\ell.$$

So

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

and since UP is closed under intersection and SDH(UP) is (trivially) closed under symmetric difference, we clearly have that $L \cap L' \in$ SDH(UP). Furthermore, since $\overline{L} = \Sigma^* \,\Delta\, L$ implies that $\overline{L} \in$ SDH(UP), SDH(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 3.5. CH(UP) = BC(UP).

Proof. We will prove that SDH(UP) $\subseteq$ CH(UP). By Theorem 3.4, this will suffice.

Let L be any set in SDH(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,

$$L = \left(\cdots \left(\left((U_1 \cup U_2 \cup \cdots \cup U_k) \cap \left(\overline{\bigcup_{j_1 < j_2} (U_{j_1} \cap U_{j_2})} \right) \right) \right.\right.$$

[2] 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. [5] (see also [26, 9, 41, 15, 14]).

$$\cup \left(\bigcup_{j_1<j_2<j_3} (U_{j_1} \cap U_{j_2} \cap U_{j_3}) \right) \Bigg) \cap \cdots \cup \left(\bigcup_{j_1<\cdots<j_k} (U_{j_1} \cap \cdots \cap U_{j_k}) \right) \Bigg),$$

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

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

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

$$\Big(\cdots \Big(\Big(\Big(U_1 \cap \overline{\emptyset} \Big) \cup U_2 \Big) \cap \overline{\emptyset} \Big) \cup \cdots \cup U_k \Big);$$

call this C_1. Clearly, $C_1 \in \mathrm{C}_{2k-1}(\mathrm{UP})$. 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 $C_3 = (A_1 \cap \overline{A_2}) \cup A_3$ 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 3.6. SDH(UP) *and* CH(UP) *are both closed under all Boolean operations.*

Note that the proofs of Theorems 3.5 and 3.4 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 SDH(UP) and CH(UP) are indeed equal, the above proof embeds $\mathrm{SD}_k(\mathrm{UP})$ in an exponentially higher level of the C hierarchy over UP. Similarly, the proof of Theorem 3.4 embeds $\mathrm{C}_k(\mathrm{UP})$ in an exponentially higher level of SDH(UP).

COROLLARY 3.7.

1. *For each* $k \ge 1$, $\mathrm{SD}_k(\mathrm{UP}) \subseteq \mathrm{C}_{2^{k+1}-k-2}(\mathrm{UP})$.
2. *For each* $k \ge 1$, $\mathrm{C}_k(\mathrm{UP}) \subseteq \mathrm{SD}_{T(k)}(\mathrm{UP})$, *where* $T(k) = 2^k - 1$ *if* k *is odd, and* $T(k) = 2^k - 2$ *if* k *is even.*

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 3.5) 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 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 3.4 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:

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

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

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

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

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

Remark 3.8. 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, since it still embeds $\mathrm{C}_k(\mathrm{UP})$ in an exponentially higher level of $\mathrm{SDH}(\mathrm{UP})$. We propose as an interesting task the establishment of *tight* level-wise containments, at least up to the limits of relativizing techniques, between the hierarchies $\mathrm{SDH}(\mathrm{UP})$ and $\mathrm{CH}(\mathrm{UP})$, both of which capture the Boolean closure of UP.

We conjecture that there is some relativized world in which an exponential increase (though less dramatic than the particular exponential increase of Corollary 3.7) indeed is necessary.

Theorem 3.9 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 3.13 below) that, relative to a recursive oracle, even the fourth level of CH(UP) and the third level of SDH(UP) are not subsumed by any level of the EH(UP) hierarchy. Consequently, neither the D nor the E normal forms of Definition 3.1 capture the Boolean closure of UP.

THEOREM 3.9. *For every* $k \geq 1$, $\mathrm{D}_k(\mathrm{UP}) \subseteq \mathrm{C}_k(\mathrm{UP}) \cap \mathrm{E}_k(\mathrm{UP})$.

Proof. For the first inclusion, by [11, Proposition 2.1.2], each set L in $\mathrm{D}_k(\mathrm{UP})$ 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 [11, Proposition 2.1.2] only uses intersection, the sets A_i are in UP. A special case of [11, Proposition 2.1.3] says that sets in $\mathrm{D}_k(\mathrm{UP})$ via decreasing chains such as the A_i are in $\mathrm{C}_k(\mathrm{UP})$, and so $L \in \mathrm{C}_k(\mathrm{UP})$.

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 $\mathrm{D}_{2n-1}(\mathrm{UP}) \subseteq \mathrm{E}_{2n-1}(\mathrm{UP})$ to be valid, and let L be any set in $\mathrm{D}_{2n+1}(\mathrm{UP}) = \mathrm{UP} - (\mathrm{UP} - \mathrm{D}_{2n-1}(\mathrm{UP}))$. By our inductive hypothesis, L can be represented as

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

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

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

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 $L \in \mathrm{E}_{2n+1}(\mathrm{UP})$. 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 Σ^* and $\emptyset$.

THEOREM 3.10. *Theorems* 3.4, 3.5, *and* 3.9 *and Corollaries* 3.6 *and* 3.7 *apply to all classes that contain* Σ^* *and* $\emptyset$ *and are closed under intersection.*

Remark 3.11. Although DP is closed under intersection but seems to lack closure under union (unless the polynomial hierarchy collapses to DP [37, 15, 14]) and thus

Theorem 3.10 in particular applies to DP, we note that the known results about Boolean hierarchies over NP [9, 41] in fact even for the DP case imply stronger results than those given by our Theorem 3.10, 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 3.8).

The following combinatorial lemma will be useful in proving Theorem 3.13.

LEMMA 3.12. [12] *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 $S' \supseteq \{s \in S \mid (\exists t \in T)\,[\langle s, t\rangle \in E]\}$, and $T' \supseteq \{t \in T \mid (\exists s \in S)\,[\langle t, s\rangle \in E]\}$. Then either*

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

For papers concerned with oracles separating internal levels of Boolean hierarchies over classes other than those of this paper, we refer the reader to [9, 8, 20, 7, 18] (see also [21]). Theorem 3.13 is optimal since 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 3.13. *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 3.14. *There is a recursive oracle A such that*

1. $\mathrm{EH}(\mathrm{UP}^A) \neq \mathrm{BC}(\mathrm{UP}^A)$ *and* $\mathrm{DH}(\mathrm{UP}^A) \neq \mathrm{BC}(\mathrm{UP}^A)$, [3] *and*
2. $\mathrm{EH}(\mathrm{UP}^A)$ *and* $\mathrm{DH}(\mathrm{UP}^A)$ *are not closed under all Boolean operations.*

Proof of Theorem 3.13. Although the theorem claims that 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 $D^{(k)}$, as a notational shorthand we will 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 $S^n_v \stackrel{\mathrm{df}}{=} \{vw \mid vw \in \Sigma^n\}$. The sets S^n_v are used to distinguish between different segments of Σ^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 $\mathrm{EH}(\mathrm{UP})$ since 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

[3] Since both Corollary 3.3 (establishing $\mathrm{DH}(\mathrm{UP}) = \mathrm{CH}(\mathrm{coUP})$) and Theorem 3.5 ($\mathrm{BC}(\mathrm{UP}) = \mathrm{CH}(\mathrm{UP})$) relativize, 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 BC(UP).

(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 jth 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 henceforth 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 $\mathrm{E}_{2k}(\mathrm{UP})$ machine relative to X), or $L(H^X) \neq L_X$, where L_X is as defined below.

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

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

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 every $n \geq 2$, then L_A is even in $\mathrm{UP}^A \wedge \mathrm{coUP}^A \wedge \mathrm{coUP}^A$ and thus in $\mathrm{C}_4(\mathrm{UP}^A)$. 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: $(\exists z \in (S_{10}^n \cup S_{11}^n) - A_j')\,[0^n \notin L(N_{i,2}^{B_j \cup \{z\}})]$.
Choose any such z. Set $A_j := B_j \cup \{z\}$. We have $0^n \in L(H^A)$ but $0^n \notin L_A$.

Case 2.2.2: $(\forall z \in (S_{10}^n \cup S_{11}^n) - A_j')\,[0^n \in L(N_{i,2}^{B_j \cup \{z\}})]$.
To apply Lemma 3.12, 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 $t \in T$, $\langle s, t\rangle \in E$ if and only if $N_{i,2}^{B_j \cup \{s\}}$ queries t along its lexicographically first accepting path, and $\langle t, s\rangle \in E$ 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 3.12 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 string 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:

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

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: $(\exists w \in (S_{10}^n \cup S_{11}^n) - D_j')\,[0^n \notin L(N_{i,2}^{E_j \cup \{w\}})]$.

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

Case 2.2.2: $(\forall w \in (S_{10}^n \cup S_{11}^n) - D_j')\,[0^n \in L(N_{i,2}^{E_j \cup \{w\}})]$.

As before, Lemma 3.12 yields two strings $s \in S_{10}^n - D_j'$ and $t \in S_{11}^n - D_j'$ such that $N_{i,2}^{E_j \cup \{s,t\}}(0^n)$ 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 3.13) for the case of US [6] (which is denoted 1NP in [21, 18]).

4. 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 [39, 34, 40, 1, 45, 54, 38].[4] Kadin showed that if there is a sparse $\leq_T^p$-complete set in NP, then the polynomial hierarchy collapses to $\mathrm{P}^{\mathrm{NP[log]}}$ [38]. Due to the promise nature of UP (in particular, UP probably lacks complete sets [22]), 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 [48])?[5]

Cai, Hemachandra, and Vyskoč [12] 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 $\mathrm{P}^{\mathcal{UP}}$, 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 of NP $\cap$ coNP by Grollmann and Selman [19]—in which *only computations actually executed need be unambiguous.* Lange, Niedermeier, and Rossmanith [43], [48, p. 482] generalize this approach to build up an entire hierarchy of unambiguous computations in which the oracle levels are guardedly accessed (Definition 4.1, part 3)—the *promise unambiguous polynomial hierarchy.*

Definition 4.1.

1. *The* polynomial hierarchy [47, 57] *is defined as follows:*
$\Sigma_0^p \stackrel{\mathrm{df}}{=} \mathrm{P}$, $\Delta_0^p \stackrel{\mathrm{df}}{=} \mathrm{P}$, $\Sigma_k^p \stackrel{\mathrm{df}}{=} \mathrm{NP}^{\Sigma_{k-1}^p}$, $\Pi_k^p \stackrel{\mathrm{df}}{=} \mathrm{co}\Sigma_k^p$, $\Delta_k^p \stackrel{\mathrm{df}}{=} \mathrm{P}^{\Sigma_{k-1}^p}$, $k \geq 1$, *and* $\mathrm{PH} \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 0} \Sigma_k^p$.

2. *The* unambiguous polynomial hierarchy [48] *is defined as follows:*
$\mathrm{U}\Sigma_0^p \stackrel{\mathrm{df}}{=} \mathrm{P}$, $\mathrm{U}\Delta_0^p \stackrel{\mathrm{df}}{=} \mathrm{P}$, $\mathrm{U}\Sigma_k^p \stackrel{\mathrm{df}}{=} \mathrm{UP}^{\mathrm{U}\Sigma_{k-1}^p}$, $\mathrm{U}\Pi_k^p \stackrel{\mathrm{df}}{=} \mathrm{coU}\Sigma_k^p$, $\mathrm{U}\Delta_k^p \stackrel{\mathrm{df}}{=} \mathrm{P}^{\mathrm{U}\Sigma_{k-1}^p}$, $k \geq 1$, *and* $\mathrm{UPH} \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 0} \mathrm{U}\Sigma_k^p$.

3. *The* promise unambiguous polynomial hierarchy [43], [48, p. 482] *is defined as follows:* $\mathcal{U}\Sigma_0^p \stackrel{\mathrm{df}}{=} \mathrm{P}$, $\mathcal{U}\Sigma_1^p \stackrel{\mathrm{df}}{=} \mathrm{UP}$, *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* $L(N_{i+2}^{L(N_{i+3}^{\cdot^{\cdot^{\cdot^{L(N_k)}}}})})$ *has at most one accepting path.* $\mathcal{UPH} \stackrel{\mathrm{df}}{=} \bigcup_{k\geq 0} \mathcal{U}\Sigma_k^p$. *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*

[4] For reductions less flexible than Turing reductions (e.g., $\leq_m^p$, $\leq_{btt}^p$, etc.), this issue has been studied even more intensely (see, e.g., the surveys [61, 28]).

[5] Note that it is not known whether such a collapse implies a collapse of PH. Note also that Toda's [58] result on whether P-selective sets can be truth-table hard for UP does not imply such a collapse since truth-table reductions are less flexible than Turing reductions.

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* Σ_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* Π *and* Δ *classes of the hierarchies.*

The following facts follow from the definition (see also [48]) or can easily be shown.

FACT 4.2. *For every* $k \geq 1$, *the following hold:*

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

The classes "$\mathrm{UP}_{\leq k}$," the analogues of UP in which up to k accepting paths are allowed, have been studied in various contexts [60, 27, 2, 12, 30, 33]. One motivation for $\mathrm{U}\Sigma_k^p$ is that, for each k, $\mathrm{UP}_{\leq k} \subseteq \mathrm{U}\Sigma_k^p$ [48].

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 kth level of UPH is contained in the $(k-1)$st level of $\mathcal{UPH}$.

THEOREM 4.3. *If there exists a sparse Turing-complete set for* UP, *then*

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

Proof. For the first statement, let L be any set in $\mathrm{UP}^{\mathrm{UP}}$. By assumption, L is in $\mathrm{UP}^{\mathrm{P}^S} = \mathrm{UP}^S$ for some sparse set $S \in \mathrm{UP}$. 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 $L \in \mathrm{UP}^S$, 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 $r(n) \stackrel{\mathrm{df}}{=} q(p(n))$ 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 $L \in \mathrm{P}^{\mathcal{UP}}$, 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 henceforth use $\mathcal{UP}$ and other $\mathcal{U}\cdots$ notations in this informal manner). Before formally describing machine M (Figure 4.1), 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 4.1), 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):

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

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. jth string of $S^{\leq p(n)}$ has no ith 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.

FIG. 4.1. *DPOM M guardedly unambiguously accessing an* NP *oracle to accept a set in* $\mathrm{UP}^{\mathrm{UP}}$.

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

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

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 is in $\mathrm{U}\Sigma_{k-1}^{p,\mathrm{P}^S} = \mathrm{U}\Sigma_{k-1}^{p,S}$; let $N_1, N_2, \ldots, N_{k-1}$ be the UPOMs that witness this fact, that is, $L = L(N_1^{L(N_2^{\cdot^{\cdot^{\cdot^{L(N_{k-1}^S)}}}})})$.

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 $N_1^{L(N_2^{\cdot^{\cdot^{\cdot^{L(N_{k-1}^S)}}}})}(x)$ 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, $L(N_2'^{L(N_3'^{\cdot^{\cdot^{\cdot^{L(N_{k-1}')}}}})})$, 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) since the above-described computation depends on the advice table T, and so, for some bad advice T, the unambiguity of the modified 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 $L \in \mathcal{U}\Sigma_{k-1}^p$. □

In the above proof, the assumption that the sparse set S is in UP is needed to determine the exact census of S 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 [39].[6] Hopcroft [34] dramatically simplified their proof, and Balcázar, Book, and Schöning [1, 54] generalized, as Theorem 4.6, the Karp–Lipton result; the general approach of Hopcroft and Balcázar, Book, and Schöning will be central to our upcoming proof of Theorem 4.7. Schöning's low hierarchy [53] gives a way of classifying the complexity of NP sets that seem to be neither in P nor NP-complete. 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 4.6 below says that $\mathrm{Low}_2 \supseteq \mathrm{NP} \cap \mathrm{P/poly} \cap \{L \mid L \text{ is self-reducible}\}$.

DEFINITION 4.4. [46]

1. *A partial order* $<_{\mathrm{pwl}}$ *on* Σ^* *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) $(\exists q : q \text{ polynomial})(\forall x, y \in \Sigma^*)\,[x <_{\mathrm{pwl}} y \implies |x| \leq q(|y|)]$.

2. *A set* A *is* self-reducible *if and only if there exist a polynomially well-founded and length-related order* $<_{\mathrm{pwl}}$ *on* Σ^* *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 4.5. [1] *Let* A *be a self-reducible set and let* M *witness* A*'s self-reducibility. For any set* B *and any* n, *if* $(L(M^B))^{\leq n} = B^{\leq n}$, *then* $A^{\leq n} = B^{\leq n}$.[7]

THEOREM 4.6. [1] *If* A *is a self-reducible set and there is a* $k \geq 0$ *and a sparse set* S *such that* $A \in \Sigma_k^{p,S}$, *then* $\Sigma_2^{p,A} \subseteq \Sigma_{k+2}^p$.

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

THEOREM 4.7. *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* j, $0 \leq j \leq k-3$.

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" [55, 49] of any UP set is self-reducible and is in UP. More precisely, to apply Theorem 4.6 we would need A to be self-reducible. Although that cannot be assumed in general of an arbitrary UP set, the left set of A, i.e., the set of prefixes of

[6] Very recently, Köbler and Watanabe [42] have improved this collapse to $\mathrm{ZPP}^{\mathrm{NP}}$, and they 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 $\mathrm{UP} \subseteq \mathrm{P/poly}$ (see [42] for the notations not defined in this footnote).

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

Description of self-reducer M_{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_{self} for B.

FIG. 4.2. *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.

FIG. 4.3. *A Turing reduction from a* UP *set A to its left set B via prefix search.*

witnesses for elements in A defined by

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

does have this property and is also in UP. A self-reducing machine M_{self} for B is given in Figure 4.2. 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 $<_{\text{pwl}}$ defined by the following: For fixed x and all strings $y_1, y_2 \in \Sigma^{\leq p(|x|)}$, $\langle x, y_1\rangle <_{\text{pwl}} \langle x, y_2\rangle$ if and only if y_2 is a prefix of y_1.

By assumption, since B is a UP set, $B \in \mathrm{P}^S$ for some sparse set S, so Theorem 4.6 with $k = 0$ applies to B. Furthermore, A is in P^B, via prefix search by DPOM M_A (Figure 4.3). Thus $L \in \Sigma_2^{p,\mathrm{P}^B} \subseteq \Sigma_2^{p,B} \subseteq \Sigma_2^p$, which shows that $A \in \text{Low}_2$.

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 4.3), B is self-reducible via M_{self} (see Figure 4.2), 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 $r(n) \stackrel{\text{df}}{=} m(p(n))$ and $s(n) \stackrel{\text{df}}{=} q(r(n))$.

To show that $L \in \mathrm{P}^{\mathcal{U}\Sigma_{k-1}^p \oplus \Sigma_2^p}$, we will describe a DPOM M that on input x, $|x| = n$, using the Σ_2^p 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 $\mathcal{U}\Sigma_{k-1}^p$ part of its oracle to simulate the $\mathrm{U}\Sigma_{k-1}^{p,A}$ computation of $N_1^{L(N_2^{\cdot^{L(N_{k-1}^A)}})}(x)$ except with N_1, N_2, $\ldots, N_{k-1}$ modified in the same way as was described in the proof of Theorem 4.3. In more detail, 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, if N_{k-1} consults its original oracle A about some query y, then the modified machine N_{k-1}' runs the P computation $M_A^{L(M_B^T)}$ 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 $L(M_B^T)$ is a fixed point of B's self-reducer M_{self} up to length $p(n)$, that is, $(L(M_{\text{self}}^{L(M_B^T)}))^{\leq p(n)} = (L(M_B^T))^{\leq p(n)}$, and thus $B^{\leq p(n)} = (L(M_B^T))^{\leq p(n)}$ by Lemma 4.5. This property is checked for each guessed T in the Σ_2^p part of the oracle. Formally,

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

The prefix search of M is similar to the one performed in the proof of Theorem 4.3 (see Figure 4.1); 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 $L \in \mathrm{U}\Sigma_j^{p,\Sigma_2^{p,\,\mathcal{U}\Sigma_{k-j-3}^p}}$. On input x, N simulates the $\mathrm{U}\Sigma_j^p$ computation of the first j UPOMs $N_1, \ldots, N_j$. In the subsequent Σ_2^p 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 Σ_2^p computation guesses all possible advice sets and the top machine checks if the guessed advice is good (that is, if $L(M_B^T)$ 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 4.3) and is used to correctly answer all queries of N_{k-1} without consulting an oracle. This proves the theorem. □

Since Theorem 4.7 relativizes and there are relativized worlds in which UP^A is not Low_2^A [56], we have the following corollary.

COROLLARY 4.8. *There is a relativized world in which* (*relativized*) UP *has no sparse Turing-hard sets.*

Acknowledgments. We are very grateful to Gerd Wechsung for his help in bringing about this collaboration, and for his kind and insightful advice over many years. We thank Marius Zimand for proofreading and Nikolai Vereshchagin for helpful discussions during his visit to Rochester. We thank Osamu Watanabe for discussing

with us his results joint with Johannes Köbler, and we thank Osamu Watanabe and Johannes Köbler for providing us with copies of their paper [42].

REFERENCES

[1] J. Balcázar, R. Book, and U. Schöning, *The polynomial-time hierarchy and sparse oracles*, J. Assoc. Comput. Mach., 33 (1986), pp. 603–617.

[2] R. Beigel, *On the relativized power of additional accepting paths*, in Proc. 4th Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamitos, CA, 1989, pp. 216–224.

[3] R. Beigel, R. Chang, and M. Ogiwara, *A relationship between difference hierarchies and relativized polynomial hierarchies*, Math. Systems Theory, 26 (1993), pp. 293–310.

[4] R. Beigel and J. Goldsmith, *Downward separation fails catastrophically for limited nondeterminism classes*, in Proc. 9th Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamitos, CA, 1994, pp. 134–138.

[5] A. Bertoni, D. Bruschi, D. Joseph, M. Sitharam, and P. Young, *Generalized Boolean hierarchies and Boolean hierarchies over RP*, in Proc. 7th Conference on Fundamentals of Computation Theory, Lecture Notes in Comput. Sci. 380, Springer-Verlag, Berlin, 1989, pp. 35–46.

[6] A. Blass and Y. Gurevich, *On the unique satisfiability problem*, Inform. and Control, 55 (1982), pp. 80–88.

[7] D. Bruschi, D. Joseph, and P. Young, *Strong separations for the Boolean hierarchy over RP*, Internat. J. Found. Comput. Sci., 1 (1990), pp. 201–218.

[8] J. Cai, *Probability one separation of the Boolean hierarchy*, in Proc. 4th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 247, Springer-Verlag, Berlin, 1987, pp. 148–158.

[9] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung, *The Boolean hierarchy* I: *Structural properties*, SIAM J. Comput., 17 (1988), pp. 1232–1252.

[10] J. Cai, T. Gundermann, J. Hartmanis, L. Hemachandra, V. Sewelson, K. Wagner, and G. Wechsung, *The Boolean hierarchy* II: *Applications*, SIAM J. Comput., 18 (1989), pp. 95–111.

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

[12] J. Cai, L. Hemachandra, and J. Vyskoč, *Promises and fault-tolerant database access*, in Complexity Theory, K. Ambos-Spies, S. Homer, and U. Schöning, eds., Cambridge University Press, Cambridge, UK, 1993, pp. 101–146.

[13] J. Cai and G. Meyer, *Graph minimal uncolorability is* D^{P}*-complete*, SIAM J. Comput., 16 (1987), pp. 259–277.

[14] R. Chang, *On the structure of NP computations under Boolean operators*, Ph.D. thesis, Cornell University, Ithaca, NY, 1991.

[15] 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, 1990.

[16] R. Chang and J. Kadin, *The Boolean hierarchy and the polynomial hierarchy: A closer connection*, SIAM J. Comput., 25 (1996), pp. 340–354.

[17] S. Cook, *The complexity of theorem-proving procedures*, in Proc. 3rd ACM Symposium on Theory of Computing, ACM, New York, 1971, pp. 151–158.

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

[19] J. Grollmann and A. Selman, *Complexity measures for public-key cryptosystems*, SIAM J. Comput., 17 (1988), pp. 309–335.

[20] T. Gundermann, N. Nasser, and G. Wechsung, *A survey on counting classes*, in Proc. 5th Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamitos, CA, 1990, pp. 140–153.

[21] T. Gundermann and G. Wechsung, *Counting classes with finite acceptance types*, Comput. Artificial Intelligence, 6 (1987), pp. 395–409.

[22] J. Hartmanis and L. Hemachandra, *Complexity classes without machines: On complete languages for UP*, Theoret. Comput. Sci., 58 (1988), pp. 129–142.

[23] J. Hartmanis and L. Hemachandra, *Robust machines accept easy sets*, Theoret. Comput. Sci., 74 (1990), pp. 217–226.

[24] J. Hartmanis, N. Immerman, and V. Sewelson, *Sparse sets in NP−P: EXPTIME versus NEXPTIME*, Inform. and Control, 65 (1985), pp. 159–181.
[25] J. Hartmanis and R. Stearns, *On the computational complexity of algorithms*, Trans. Amer. Math. Soc., 117 (1965), pp. 285–306.
[26] F. Hausdorff, *Grundzüge der Mengenlehre*, Walter De Gruyten & Co., Berlin, Leipzig, 1927.
[27] L. Hemachandra, *Counting in Structural Complexity Theory*, Ph.D. thesis, Technical Report TR87-840, Department of Computer Science, Cornell University, Ithaca, NY, 1987.
[28] L. Hemachandra, M. Ogiwara, and O. Watanabe, *How hard are sparse sets*?, in Proc. 7th Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamitos, CA, 1992, pp. 222–238.
[29] L. Hemachandra and R. Rubinstein, *Separating complexity classes with tally oracles*, Theoret. Comput. Sci., 92 (1992), pp. 309–318.
[30] E. Hemaspaandra and L. Hemaspaandra, *Quasi-injective reductions*, Theoret. Comput. Sci., 123 (1994), pp. 407–413.
[31] L. Hemaspaandra, S. Jain, and N. Vereshchagin, *Banishing robust Turing completeness*, Internat. J. Found. Comput. Sci., 4 (1993), pp. 245–265.
[32] L. Hemaspaandra and S. Jha, *Defying upward and downward separation*, Inform. and Computation, 121 (1995), pp. 1–13.
[33] L. Hemaspaandra and M. Zimand, *Strong self-reducibility precludes strong immunity*, Math. Systems Theory, 29 (1996), pp. 535–548.
[34] J. Hopcroft, *Recent directions in algorithmic research*, in Proc. 5th GI Conference on Theoretical Computer Science, Lecture Notes in Comput. Sci. 104, Springer-Verlag, Berlin, 1981, pp. 123–134.
[35] J. Hopcroft and J. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison–Wesley, Reading, MA, 1979.
[36] R. Impagliazzo and G. Tardos, *Decision versus search problems in super-polynomial time*, in Proc. 30th IEEE Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1989, pp. 222–227.
[37] J. Kadin, *The polynomial time hierarchy collapses if the Boolean hierarchy collapses*, SIAM J. Comput., 17 (1988), pp. 1263–1282; erratum, SIAM J. Comput., 20 (1991), p. 404.
[38] J. Kadin, $P^{NP[\log n]}$ *and sparse Turing-complete sets for NP*, J. Comput. System Sci., 39 (1989), pp. 282–298.
[39] R. Karp and R. Lipton, *Some connections between nonuniform and uniform complexity classes*, in Proc. 12th ACM Symposium on Theory of Computing, ACM, New York, 1980, pp. 302–309; an extended version has also appeared as *Turing machines that take advice*, Enseign. Math. (2), 28 (1982), pp. 191–209.
[40] K. Ko and U. Schöning, *On circuit-size complexity and the low hierarchy in NP*, SIAM J. Comput., 14 (1985), pp. 41–51.
[41] J. Köbler, U. Schöning, and K. Wagner, *The difference and truth-table hierarchies for NP*, RAIRO Inform. Théor. Appl., 21 (1987), pp. 419–435.
[42] J. Köbler and O. Watanabe, *New collapse consequences of NP having small circuits*, in Proc. 22nd International Colloquium on Automata, Languages, and Programming, Lecture Notes in Comput. Sci. 944, Springer-Verlag, Berlin, 1995, pp. 196–207.
[43] K.-J. Lange and P. Rossmanith, *Unambiguous polynomial hierarchies and exponential size*, in Proc. 9th Structure in Complexity Theory Conference, IEEE Computer Society Press, Los Alamitos, CA, 1994, pp. 106–115.
[44] L. Levin, *Universal sorting problems*, Problems Inform. Transmission, 9 (1973), pp. 265–266.
[45] T. Long and A. Selman, *Relativizing complexity classes with sparse oracles*, J. Assoc. Comput. Mach., 33 (1986), pp. 618–627.
[46] A. Meyer and M. Paterson, *With what frequency are apparently intractable problems difficult*?, Technical Report MIT/LCS/TM-126, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1979.
[47] A. Meyer and L. Stockmeyer, *The equivalence problem for regular expressions with squaring requires exponential space*, in Proc. 13th IEEE Symposium on Switching and Automata Theory, IEEE Computer Society Press, Los Alamitos, CA, 1972, pp. 125–129.
[48] R. Niedermeier and P. Rossmanith, *Extended locally definable acceptance types*, in Proc. 10th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 665, Springer-Verlag, Berlin, 1993, pp. 473–483.
[49] M. Ogiwara and O. Watanabe, *On polynomial-time bounded truth-table reducibility of NP sets to sparse sets*, SIAM J. Comput., 20 (1991), pp. 471–483.
[50] C. Papadimitriou and M. Yannakakis, *The complexity of facets (and some facets of complexity)*, J. Comput. System Sci., 28 (1984), pp. 244–259.

[51] R. Rao, J. Rothe, and O. Watanabe, *Upward separation for FewP and related classes*, Inform. Process. Lett., 52 (1994), pp. 175–180.
[52] K. Regan, *Provable complexity properties and constructive reasoning*, manuscript, 1989.
[53] U. Schöning, *A low and a high hierarchy within NP*, J. Comput. System Sci., 27 (1983), pp. 14–28.
[54] U. Schöning, *Complexity and Structure*, Lecture Notes in Comput. Sci. 211, Springer-Verlag, 1986.
[55] L. Selman, *Natural self-reducible sets*, SIAM J. Comput., 17 (1988), pp. 989–996.
[56] M. Sheu and T. Long, *UP and the low and high hierarchies: A relativized separation*, Math. Systems Theory, 29 (1996), pp. 423–450.
[57] L. Stockmeyer, *The polynomial-time hierarchy*, Theoret. Comput. Sci., 3 (1977), pp. 1–22.
[58] S. Toda, *On polynomial-time truth-table reducibilities of intractable sets to P-selective sets*, Math. Systems Theory, 24 (1991), pp. 69–82.
[59] L. Valiant, *The relative complexity of checking and evaluating*, Inform. Process. Lett., 5 (1976), pp. 20–23.
[60] O. Watanabe, *On hardness of one-way functions*, Inform. Process. Lett., 27 (1988), pp. 151–157.
[61] P. Young, *How reductions to sparse sets collapse the polynomial-time hierarchy: A primer*, SIGACT News, 1992, #3, pp. 107–117 (part I), #4, pp. 83–94 (part II), and #4, p. 94 (corrigendum to part I).