Structural properties of oracle classes

Stanislav Živný

Computing Laboratory, University of Oxford, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

Abstract

Denote by $\mathcal{C}$ the class of oracles relative to which $\mathsf{P} = \mathsf{NP}$ (collapsing oracles), and by $\mathcal{S}$ the class of oracles relative to which $\mathsf{P} \neq \mathsf{NP}$ (separating oracles). We present structural results on $\mathcal{C}$ and $\mathcal{S}$. Using a diagonalization argument, we show that neither $\mathcal{C}$ nor $\mathcal{S}$ is closed under disjoint union, also known as join. We show that this implies that neither $\mathcal{C}$ nor $\mathcal{S}$ is closed under union, intersection, or symmetric difference. Consequently EL_1, the first level of the extended low hierarchy, is not closed under join.

Key words:
Computational complexity, Diagonalization, Extended hierarchies, Relativization

1. Introduction

In the seminal paper on relativization [2], Baker et al. showed the existence of an oracle relative to which $\mathsf{P} = \mathsf{NP}$ and an oracle relative to which $\mathsf{P} \neq \mathsf{NP}$. For several decades the relativization method has been used in many different areas of computational complexity (see for example [8]).

This paper follows the idea of Balcázar [3] and Hartmanis and Hemachandra [10] to investigate properties of oracles relative to the P vs. NP problem. We denote by $\mathcal{C}$ the class of sets relative to which $\mathsf{P} = \mathsf{NP}$ (collapsing oracles), and by $\mathcal{S}$ the class of sets relative to which $\mathsf{P} \neq \mathsf{NP}$ (separating oracles). The most interesting open question is whether the empty set lies in $\mathcal{C}$ or $\mathcal{S}$, which is equivalent to the P vs. NP problem. In this paper we investigate structural properties of $\mathcal{C}$ and $\mathcal{S}$ such as closedness under set operations.

In Section 2, we recall standard notation from complexity theory, present known results about $\mathcal{C}$ and $\mathcal{S}$, and observe a simple characterisation of $\mathcal{C}$ in terms of low levels of the extended low and high hierarchies. We also show that it is sufficient to show that both $\mathcal{C}$ and $\mathcal{S}$ are not closed under join in order to get the same results for other operations. In Section 3, using a diagonalization argument, we show that $\mathcal{S}$ is not closed under join. In Section 4, using a diagonalization argument, we prove that $\mathcal{C}$ is not closed under join. Finally, in Section 5, we conclude with some open problems.

Email address: `stanislav.zivny@comlab.ox.ac.uk` (Stanislav Živný).

2. Notation and basic properties of $\mathcal{C}$ and $\mathcal{S}$

In this section, we first recall some standard notation.

An alphabet is any non-empty finite set, in our case $\Sigma = \{0, 1\}$. The set of all words over Σ of length n is denoted Σ^n. The set of all finite words over Σ is denoted Σ^*. All the sets, languages and oracles in this paper are subsets of Σ^*. We denote by $\leq_{lex}$ the standard lexicographical order over words. The i-th bit of a word x is denoted $x[i]$. For any set B, we denote by $\overline{B}$ the *complement* of B, defined as $\overline{B} = \Sigma^* \setminus B$. For any set B, we define $I(B) = \{0^n \mid (\exists x \in B)[|x| = n]\}$. For any two sets A and B, we denote by $A \triangle B$ the *symmetric difference* of A and B, defined as $A \triangle B = (A \setminus B) \cup (B \setminus A)$. We denote by $A \oplus B$ the *disjoint union* of A and B (also known as *join*), defined as $A \oplus B = \{0w \mid w \in A\} \cup \{1w \mid w \in B\}$.

Throughout this paper we use standard notation from complexity theory (see for example [5,6,13]). For a Turing machine M and an oracle A, we denote by $L(M, A)$ the language accepted by M with the oracle A. We will work with complexity classes P, NP, PH (polynomial-time hierarchy), PSPACE, E (deterministic exponential time with linear exponent), NE (nondeterministic E) and their relativized versions.

We denote by $\leq_m^p$ the standard polynomial-time many-one reducibility. We denote by $\leq_T^p$ the standard polynomial-time Turing reducibility. If $A \leq_m^p B$ and $B \leq_m^p A$, we say that A and B are polynomial-time many-one equivalent, denoted $\mathsf{A} \equiv_m^p B$, and similarly for $\leq_T^p$-reducibility. We say that a class $\mathcal{C}$ is closed under $\leq_m^p$

downwards, if $A \leq_m^p B$ for some $B \in \mathcal{C}$ implies $A \in \mathcal{C}$. We say that a class $\mathcal{C}$ is closed under $\leq_m^p$ *upwards*, if $A \leq_m^p B$ for some $A \in \mathcal{C}$ implies $B \in \mathcal{C}$. Similarly, we define the notion of $\mathcal{C}$ being closed downwards and upwards for $\leq_T^p$-reducibility.

We say that A is *hard* for a class $\mathcal{C}$ if $C \leq_m^p A$ for every $C \in \mathcal{C}$. We say that A is *complete* for $\mathcal{C}$ if A is hard for $\mathcal{C}$ and $A \in \mathcal{C}$. Sometimes we also use the notion of completeness and hardness for $\leq_T^p$-reducibility but in such cases we will mention that explicitly. Recall the standard complete problem for PSPACE^A, defined as $KS(A) = \{\langle M, x, 1^s\rangle \mid M$ is a deterministic Turing machine that accepts x with the oracle A using an amount of space bounded by $s\}$.

Recall from [4] that for every $n \geq 1$, we define $\mathsf{EL}_n = \{A \mid \Sigma_n^{p,A} \subseteq \Sigma_{n-1}^{p,A\oplus SAT}\}$, and for every $n \geq 0$, we define $\mathsf{EH}_n = \{A \mid \Sigma_n^{p,A\oplus SAT} \subseteq \Sigma_n^{p,A}\}$, where $\Sigma_n^{p,A}$ is the n-th level of the polynomial-time hierarchy relative to an oracle A. The *extended low hierarchy* is then defined as $\mathsf{ELH} = \cup_{n\geq 1}\mathsf{EL}_n$, and the *extended high hierarchy* is defined as $\mathsf{EHH} = \cup_{n\geq 0}\mathsf{EH}_n$.

There is an effective enumeration $\{M_i\}_{i\geq 1}$ of deterministic oracle Turing machines, where M_i runs in time $p_i(n) = n^i + i$, and M_i can be found in polynomial time in i, such that that for every A, $\mathsf{P}^A = \{L(M_i, A) \mid i \geq 1\}$. Note that for every i, p_i is non-decreasing, and for every i and n, $p_i(n) \leq p_{i+1}(n)$. We refer to this enumeration as *enumeration* $\{M_i\}_{i\geq 1}$. For the existence of $\{M_i\}_{i\geq 1}$, see [5].

Now we define the classes of collapsing ($\mathcal{C}$ for collapsing) and separating ($\mathcal{S}$ for separating) oracles relative to the P vs. NP problem.

Definition 1

(i) $\mathcal{C} = \{A \mid \mathsf{P}^A = \mathsf{NP}^A\}$.

(ii) $\mathcal{S} = \{B \mid \mathsf{P}^B \neq \mathsf{NP}^B\}$.

Next we sum up some known properties of $\mathcal{C}$ and $\mathcal{S}$.

Theorem 2.1 ([2,5,11,9,15])

(i) $\mathcal{C} \neq \emptyset$.

(ii) $\mathcal{S} \neq \emptyset$.

(iii) $(\exists A \in \mathsf{PH}$ *such that* $A \in \mathcal{C}) \Leftrightarrow \mathsf{PH}$ *collapses.*

(iv) A *is* PSPACE*-complete* $\Rightarrow A \in \mathcal{C}$.

(v) A *is* E*-complete* $\Rightarrow A \in \mathcal{C}$.

(vi) A *is* NE*-complete* $\Rightarrow A \in \mathcal{C}$.

(vii) $(\forall C)[\exists A, B \mid C \oplus A \in \mathcal{C} \wedge C \oplus B \in \mathcal{S}]$.

(viii) *Both* $\mathcal{C}$ *and* $\mathcal{S}$ *are closed under* $\equiv_T^p$*, and therefore under complement.*

(ix) *Neither* $\mathcal{C}$ *nor* $\mathcal{S}$ *is closed under* $\leq_m^p$ *(neither downwards nor upwards).*

PROOF. Statements (i), (ii), and (iv) are from [2]. The set B from statement (ii) is constructed by a diagonalization argument in such a way that $I(B) \in \mathsf{NP}^B \setminus \mathsf{P}^B$. B can be constructed in a way that $B \in \mathsf{E}$ [15]. Statement (iii) is Exercise 16 in Chapter 8 of [5], and a proof can be found for example in [15]. Statement (v) is the standard simulation of nondeterministic computation [15]. Statement (vi) was proved by Hemachandra [11]. Statement (vii) is from [9]. Statements (viii) and (ix) are easy [15].

Some other properties of $\mathcal{C}$ and $\mathcal{S}$ concerning complete and hard problems for various complexity classes can be found in [15].

We observe a close correspondence between $\mathcal{C}$ and low levels of the extended low and high hierarchies.

Proposition 2.2 $A \in \mathcal{C} \Leftrightarrow (A \in \mathsf{EL}_1) \wedge (SAT \leq_T^p A)$.

PROOF. Let $A \in \mathcal{C}$, that is, $\mathsf{P}^A = \mathsf{NP}^A$. Clearly, $\mathsf{NP}^A \subseteq \mathsf{P}^A \subseteq \mathsf{P}^{A\oplus SAT}$. Therefore, $A \in \mathsf{EL}_1$. By $SAT \in \mathsf{NP}^A = \mathsf{P}^A$, it follows that $SAT \leq_T^p A$.

Let $A \in \mathsf{EL}_1$ and $SAT \leq_T^p A$. The first property means that $\mathsf{NP}^A \leq_T^p A \oplus SAT$. Since $SAT \leq_T^p A$, we get $A \oplus SAT \leq_T^p A$. Together, $\mathsf{NP}^A \leq_T^p A$ and hence $\mathsf{P}^A = \mathsf{NP}^A$.

Proposition 2.3 ([4]) $A \in \mathsf{EH}_0 \Leftrightarrow SAT \leq_T^p A$.

Corollary 2.4 $A \in \mathcal{C} \Leftrightarrow A \in (\mathsf{EL}_1 \cap \mathsf{EH}_0)$.

Now we show that in order to show that $\mathcal{C}$ is not closed under union, intersection, symmetric difference, and join it is sufficient to show that $\mathcal{C}$ is not closed under join, and similarly for $\mathcal{S}$. A similar argument has been used in [12].

Proposition 2.5 *If* $\mathcal{C}$ *is not closed under* $\oplus$*, then* $\mathcal{C}$ *is also not closed under* $\cup$*,* $\cap$ *and* $\triangle$*. Analogously for* $\mathcal{S}$*.*

PROOF.

Let $A, B \in \mathcal{C}$ and $A \oplus B \notin X$. Clearly, $0A \equiv_T^p A$ and $1B \equiv_T^p B$. By Theorem 2.1 (viii), $0A \in \mathcal{C}$ and $1B \in \mathcal{C}$. However, $0A \cup 1B = 0A \triangle 1B = A \oplus B \notin \mathcal{C}$. By Theorem 2.1 (viii), $\overline{0A} \in \mathcal{C}$ and $\overline{1B} \in \mathcal{C}$. However, $\overline{0A} \cap \overline{1B} = \overline{A} \oplus \overline{B}$ and clearly, $\overline{A} \oplus \overline{B} \equiv_T^p A \oplus B$. Again, by Theorem 2.1 (viii), $\overline{0A} \cap \overline{1B} \notin \mathcal{C}$. Note that we only used that $\mathcal{C}$ is closed under $\equiv_T^p$ and complement. As the same holds for $\mathcal{S}$, the same proof goes through for $\mathcal{S}$.

3. $\mathcal{S}$ is not closed under $\oplus$

In this section, we prove that $\mathcal{S}$ is not closed under join, and consequently, neither under union, intersection, nor symmetric difference. Note first that we could prove that $\mathcal{S}$ is not closed under union and intersection much easier: take $A = Q \oplus C$ and $B = Q \oplus \overline{C}$, where Q is some PSPACE-complete problem and C satisfies $Q \oplus C \in \mathcal{S}$ (Theorem 2.1 (vii)). It can be easily shown that $A \cup B = Q \oplus \Sigma^* \notin \mathcal{S}$ and $A \cap B = Q \oplus \emptyset \notin \mathcal{S}$. In order to get the same result for symmetric difference and, more importantly, for join, we use a diagonalization argument in a more involved way.

Theorem 3.1 $\mathcal{S}$ *is not closed under* $\oplus$.

PROOF.

We construct A and B such that $A, B \in \mathcal{S}$, but $A \oplus B \notin \mathcal{S}$. Before giving the formal proof, we show the idea of the proof. To ensure that $A, B \in \mathcal{S}$, that is, $\mathsf{P}^A \neq \mathsf{NP}^A$ and $\mathsf{P}^B \neq \mathsf{NP}^B$, we diagonalize against deterministic polynomial-time Turing machines with oracles in the standard way as in the

stage 0:
$k(0) = l(0) = 0$
$A_0 = B_0 = \{xx \mid x \in Q\}$
stage n:
Let $k(n)$ be the smallest odd natural number s.t. $k(n) > \max\{p_{n-1}(k(n-1)), p_{n-1}(l(n-1))\}$ and $p_n(k(n)) < 2^{k(n)}$.
If $0^{k(n)} \in L(M_n, A_{n-1})$ then $A'_n = A_{n-1}$ and $B'_n = B_{n-1}$.
If $0^{k(n)} \notin L(M_n, A_{n-1})$ then $A'_n = A_{n-1} \cup \{y_1(n)\}$ and $B'_n = B_{n-1} \cup S_1$,
where $y_1(n)$ is the first word, in lexicographical order, of length $k(n)$ s.t. $y_1(n)$ is not queried in the computation of M_n on input $0^{k(n)}$ with the oracle A_{n-1}, and $S_1 = \{x \mid |x| = k(n) \wedge x \leq_{lex} y_1(n)\}$.
Let $l(n)$ be the smallest odd natural number s.t. $l(n) > \max\{p_{n-1}(l(n-1)), p_n(k(n))\}$ and $p_n(l(n)) < 2^{l(n)}$.
If $0^{l(n)} \in L(M_n, B'_n)$ then $B_n = B'_n$ and $A_n = A'_n$.
If $0^{l(n)} \notin L(M_n, B'_n)$ then $B_n = B'_n \cup \{y_2(n)\}$ and $A_n = A'_n \cup S_2$,
where $y_2(n)$ is the first word, in lexicographical order, of length $l(n)$ s.t. $y_2(n)$ is not queried in the computation of M_n on input $0^{l(n)}$ with the oracle B'_n, and $S_2 = \{x \mid |x| = l(n) \wedge x \leq_{lex} y_2(n)\}$.

Figure 1. Construction of A, B such that $I(A) \notin \mathsf{P}^A$ and $I(B) \notin \mathsf{P}^B$.

proof of Theorem 2.1 (ii) (see [5]). Besides the diagonalization, we also encode a PSPACE-complete set Q into A and B. This ensures that A and B are PSPACE-hard. Of course, neither A nor B can belong to PSPACE (in that case, they would be PSPACE-complete and therefore belong to $\mathcal{C}$ by Theorem 2.1 (iv)). The trick is to encode some information about A into B and vice versa, so that $A \oplus B$ is powerful enough to recognize strings which we use in A and B for the diagonalization, and hence $A \oplus B \in \mathcal{C}$.

Recall the enumeration $\{M_i\}_{i\geq 1}$ from Section 2. We construct A and B in stages. Denote by $k(n)$ and $l(n)$ increasing sequences of natural numbers: $k(n)$ is the length of the word which is used in the n-th stage to ensure that M_n does not accept $I(A)$ and $l(n)$ is defined similarly for M_n and $I(B)$. After $n-1$ stages, we denote the so far constructed oracles A_{n-1} and B_{n-1}. In the n-th stage, we add at most one word of length $k(n)$ to A and at most one word of length $l(n)$ to B to ensure that $I(A) \neq L(M_n, A)$ and $I(B) \neq L(M_n, B)$. If a word w is added to A, then some words are added to B in order to ensure that $A \oplus B \in \mathcal{C}$. Similarly, for B and A the other way around. We show that either $0^{k(n)} \in L(M_n, A)$ and $A \cap \Sigma^{=k(n)} = \emptyset$, or $0^{k(n)} \notin L(M_n, A)$ and $A \cap \Sigma^{=k(n)} \neq \emptyset$. Similarly, we show that either $0^{l(n)} \in L(M_n, B)$ and $B \cap \Sigma^{=l(n)} = \emptyset$, or $0^{l(n)} \notin L(M_n, B)$ and $B \cap \Sigma^{=l(n)} \neq \emptyset$.

Figure 1 describes the n-th stage in the construction of A and B. The final oracles are $A = \cup_n A_n$ and $B = \cup_n B_n$.

By the conditions on $k(n)$ and $l(n)$, appropriate words $y_1(n)$ of length $k(n)$ and $y_2(n)$ of length $l(n)$ always exist if needed. There are $2^{k(n)}$ words of length $k(n)$ but $p_n(k(n)) < 2^{k(n)}$ and there are $2^{l(n)}$ words of length $l(n)$ but $p_n(l(n)) < 2^{l(n)}$.

The conditions on $k(n)$ and $l(n)$ also ensure that $k(n)$ and $l(n)$ are long enough not to disturb, by the possible addition of some words to A and B, any computation from the previous stages. The condition on $l(n)$, that $l(n) > p_n(k(n))$, ensures that the second part of the n-th stage does not disturb, by the possible addition of some words to A and B, the first part of the n-th stage. This means that $0^{k(n)} \in L(M_n, A_{n-1}) \Leftrightarrow 0^{k(n)} \in L(M_n, A)$. Similarly, $0^{l(n)} \in L(M_n, B_{n-1}) \Leftrightarrow 0^{l(n)} \in L(M_n, B)$. This holds for every n. Therefore, $0^{k(n)} \in I(A) \Leftrightarrow 0^{k(n)} \notin L(M_n, A)$ for every n. Similarly, $0^{l(n)} \in I(B) \Leftrightarrow 0^{l(n)} \notin L(M_n, B)$ for every n. Therefore, $I(A) \notin \mathsf{P}^A$ and $I(B) \notin \mathsf{P}^B$. Since $I(A) \in \mathsf{NP}^A$ and $I(B) \in \mathsf{NP}^A$, we obtain $\mathsf{P}^A \neq \mathsf{NP}^A$ and $\mathsf{P}^B \neq \mathsf{NP}^B$, that is, $A \in \mathcal{S}$ and $B \in \mathcal{S}$.

Note the structure of $A \oplus B$ which follows from the construction of A and B. If w is a word of even length from $A \oplus B$, then either w is the only word of length n in $0A$ (we call this word the *unique word* of length n in $A \oplus B$) and $1B$ consists of all words of length n lexicographically smaller than or equal to w, or the other way around (interchange A and B). Hence there is a deterministic polynomial-time algorithm (with the oracle $A \oplus B$) that, for a given even n, determines whether there are no words of length n in $A \oplus B$, or finds the unique word w of length n and also finds out whether $w \in A$ or $w \in B$. The algorithm uses binary search and needs only $O(n)$ queries to the oracle.

Now we show that $A \oplus B \notin \mathcal{S}$. Take an arbitrary $L \in \mathsf{NP}^{A\oplus B}$, then there is a nondeterministic polynomial-time Turing machine M_1 with an oracle such that $L = L(M_1, A \oplus B)$. Let polynomial $p(n)$ bound the running time of M_1.

We define a Turing machine M_2 which has on its input tape, besides the input word x of length n, the description of words of even length in $A \oplus B$. Note that as shown above, this description can be computed in polynomial time in n, and hence has polynomial length in n: for every even number k between 2 and $p(n)$ (or $p(n) - 1$, if $p(n)$ is odd), it consists of the unique word w of length k (if it exists) and a single bit indicating whether $w \in A$ or $w \in B$. Define a Turing machine M_2 such that M_2 on input x (and its extended input) works as follows:

– M_2 simulates M_1 on input x until M_1 asks an oracle a word w, or accepts, or rejects. M_2 accepts whenever M_1 accepts, and rejects whenever M_1 rejects.
– If M_1 asks w of odd length, then the answer is yes if and only if the answer for M_2 is yes, and M_2 resumes the simulation of M_1.

– If M_1 asks w of even length, then M_2 uses the extended input for the answer, and then M_2 resumes the simulation of M_1.

M_2 is nondeterministic because M_1 is nondeterministic and uses polynomial time because M_1 works in polynomial time. Note that M_2 does not query words of even length. If the oracle of M_2 is $A \oplus B$ and its extended input encodes the structure of words of even length in $A \oplus B$, then M_2 accepts L, that is, $L = L(M_2, A \oplus B)$.

Because M_2 does not query words of even length, the only information that M_2 can obtain from the oracle $A \oplus B$ on odd words are words of even length from A and B, which encode the PSPACE-complete problem Q. By Theorem 2.1 (iv), there is a deterministic polynomial-time Turing machine M_3 with an oracle such that $L = L(M_3, A \oplus B)$ with an extended input. Let polynomial $q(n)$ bound the running time of M_3.

Define a Turing machine M_4 such that M_4 on input x of length n works as follows:

– By binary search with the help of the oracle, M_4 first computes in polynomial time the description of words of even length in the oracle set.
– M_4 simulates M_3 on input x until M_3 asks a word w, or accepts, or rejects. M_4 accepts whenever M_3 accepts, and rejects whenever M_3 rejects.
– If M_3 asks w of odd length, then the answer is yes if and only if the answer for M_4 is yes, and M_4 resumes the simulation of M_3.
– If M_3 asks w of even length to its extended input, M_4 uses the extended input, and M_4 resumes the simulation of M_3.

M_4 is deterministic because M_3 is deterministic and uses polynomial time because M_3 works in polynomial time and Step 1 takes polynomial time (Claim 1). If the oracle of M_4 is $A \oplus B$, then M_4 accepts L, that is, $L = L(M_4, A \oplus B)$. Therefore, $L \in \mathsf{P}^{A\oplus B}$. Since L is arbitrary, $\mathsf{NP}^{A\oplus B} \subseteq \mathsf{P}^{A\oplus B}$, that is, $A \oplus B \notin \mathcal{S}$.

By Proposition 2.5, we get an immediate

Corollary 3.2 *$\mathcal{S}$ is not closed under $\cup, \cap, \triangle$.*

Corollary 3.3 *There exist sets $A \notin \mathsf{EL}_1$ and $B \notin \mathsf{EL}_1$ such that $A \oplus B \in \mathsf{EL}_1$.*

PROOF. By Corollary 2.4, $A \in \mathcal{C} \Leftrightarrow (A \in \mathsf{EL}_1 \cap \mathsf{EH}_0)$. In the proof of Theorem 3.1, we constructed $A, B \in \mathcal{S}$ such that $A \oplus B \notin \mathcal{S}$. This means that $A, B \notin \mathcal{C}$ and $A \oplus B \in \mathcal{C}$. In other words, $(A \notin \mathsf{EL}_1 \vee A \notin \mathsf{EH}_0)$, $(B \notin \mathsf{EL}_1 \vee B \notin \mathsf{EH}_0)$ and $A \oplus B \in \mathsf{EL}_1$. Since $Q \leq_T^p A$ and $Q \leq_T^p B$, where Q is the PSPACE-complete problem encoded into A and B, it follows that $SAT \leq_T^p A$ and $SAT \leq_T^p B$. By Proposition 2.3, $A \notin \mathsf{EL}_1$ and $B \notin \mathsf{EL}_1$.

4. $\mathcal{C}$ is not closed under $\oplus$

In this section, we prove that $\mathcal{C}$ is not closed under join.

Theorem 4.1 *$\mathcal{C}$ is not closed under $\oplus$.*

PROOF. For any two sets R and S, we define

$$I(R, S) = \{0^n \mid R \cap \Sigma^n \neq S \cap \Sigma^n\}.$$

Let $A = KS(R)$ and $B = KS(S)$. From the definition of A and B, $\mathsf{NP}^A \subseteq \mathsf{PSPACE}^A \subseteq \mathsf{P}^A$. Hence $A, B \in \mathcal{C}$.

Next we show that for any R and S, $I(R, S) \in \mathsf{NP}^{A\oplus B}$: on input 0^n, the machine guesses y of length n, and accepts if either $y \in R$ and $y \notin S$, or $y \notin R$ and $y \in S$.

In order to show that $A \oplus B \notin \mathcal{C}$, it remains to construct R and S such that $I(R, S) \notin \mathsf{P}^{A\oplus B}$. For this we use a diagonalization argument similar to the argument used in the proof of Theorem 3.1. We construct R and S in stages. Denote by $k(n)$ an increasing sequence of natural numbers: $k(n)$ is the length of the word which is used in the n-th stage to ensure that M_n does not accept $I(R, S)$. After $n-1$ stages, we denote the so far constructed oracles R_{n-1} and S_{n-1}. We show that either $0^{k(n)} \in L(M_n, KS(R_n) \oplus KS(S_n))$ and $0^{k(n)} \notin I(R, S)$, or $0^{k(n)} \notin L(M_n, KS(R_n) \oplus KS(S_n))$ and $0^{k(n)} \in I(R, S)$.

Figure 2 describes the n-th stage in the construction of R and S. It is clear from the construction that in the n-th stage the answers of the oracle $KS(R_{n-1} \cup T_1) \oplus KS(S_{n-1} \cup T_2)$ to M_n's queries are the same for any $T_1, T_2 \in \mathcal{Q}$, and hence M_n's output is the same. The final oracles are $R = \cup_n R_n$ and $S = \cup_n S_n$.

Observe that in the simulation of M_n each query q_i will decrease the size of the set $\mathcal{Q}$ by at most half. Thus we will result with a set $\mathcal{Q}$ of size at least $2^{2^{k(n)}-p_n(k(n))}$. From the condition on $k(n)$, $\mathcal{Q}$ has at least two different sets T_1 and T_2.

Similarly to the proof of Theorem 3.1, the conditions on $k(n)$ ensure that each $k(n)$ is long enough no to disturb, by the addition of some words to R and S, any computation from the previous stages. This means that $0^{k(n)} \in L(M_n, KS(R_n) \oplus KS(S_n)) \Leftrightarrow 0^{k(n)} \in L(M_n, A \oplus B)$. This holds for every n. Hence $0^{k(n)} \in I(R, S) \Leftrightarrow 0^{k(n)} \notin L(M_n, A \oplus B)$. Therefore, $I(R, S) \notin \mathsf{P}^{A\oplus B}$.

By Proposition 2.5, we get an immediate

Corollary 4.2 *$\mathcal{C}$ is not closed under $\cup, \cap, \triangle$.*

Theorem 4.1 together with Corollary 2.4 give the following

Corollary 4.3 *EL_1 is not closed under $\oplus$.*

5. Conclusion

We have studied structural properties of the classes of collapsing and separating oracles relative to the P vs. NP problem. Although we have not proved any separation of complexity classes, we believe that studying structural properties of different classes might lead towards new separation results. We proved that both $\mathcal{C}$ and its complement $\mathcal{S}$ are not closed under union, intersection, symmetric difference, and join. One consequence is that EL_1, the first level of the extended low hierarchy, is not closed under join.

stage 0:
$k(0) = 0$
$R_0 = S_0 = \emptyset$
stage n:
Let $k(n)$ be the smallest natural number s.t. $k(n) > p_{n-1}(k(n-1))$ and $2^{2^{k(n)-p_n(k(n))}} \geq 2$.
Let $\mathcal{Q}$ be the set of all possible subsets of $\Sigma^{k(n)}$.
Simulate M_n on input $0^{k(n)}$.
If M_n queries q_1 to A, then for every $T \in \mathcal{Q}$, consider the set $R_{n-1} \cup T$ and the oracle answer $KS(R_{n-1} \cup T)(q_1)$.
Choose the more popular answer b_1 (which is answered for at least half of the sets in $\mathcal{Q}$),
keep in $\mathcal{Q}$ only sets T for which $KS(R_{n-1} \cup T)(q_1) = b_1$, and resume the simulation of M_n.
If M_n queries q_2 to B, then for every $T \in \mathcal{Q}$, consider the set $S_{n-1} \cup T$ and the oracle answer $KS(S_{n-1} \cup T)(q_2)$.
Choose the more popular answer b_2 (which is answered for at least half of the sets in $\mathcal{Q}$),
keep in $\mathcal{Q}$ only sets T for which $KS(S_{n-1} \cup T)(q_2) = b_2$, and resume the simulation of M_n.
Let $T_1, T_2 \in \mathcal{Q}$ such that $T_1 \neq T_2$.
If M_n accepts $0^{k(n)}$, set $R_n = R_{n-1} \cup T_1$ and $S_n = S_{n-1} \cup T_1$.
If M_n rejects $0^{k(n)}$, set $R_n = R_{n-1} \cup T_1$ and $S_n = S_{n-1} \cup T_2$.

Figure 2. Construction of R and S such that $I(R, S) \notin \mathsf{P}^{A \oplus B}$.

Hemaspaandra et al. have shown that EL_2 is not closed under join [12]. Their results seems to be incomparable to ours, that is, one does not imply the other. Allender and Hemachandra have observed that EL_1 is not closed under $\leq_m^p$ [1]. We now know that neither of the first two levels of EL is closed under join. It is an interesting question whether any level of EL can be closed under join. (Note that Allender and Hemachandra have shown that if PH is infinite, none of the levels of EL is closed under $\leq_m^p$ [1]).

It is not known whether there is an $A \in \mathcal{C}$ such that $A \in \mathsf{PSPACE}$ and A is not PSPACE-complete. (And Theorem 2.1 (iii) shows that such an A cannot be in PH unless PH, the polynomial-time hierarchy, collapses.) However, this problem is very hard. Even the simpler question of whether there is an $A \in \mathsf{PSPACE}$ such that A is not PSPACE-complete is open, and an affirmative solution would imply $\mathsf{P} \neq \mathsf{PSPACE}$. Toda has shown that PP is polynomial-time Turing hard for PH, that is, $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}}$ [14], where PP is the class of languages accepted by probabilistic polynomial-time Turing machines, and $\mathsf{PP} \subseteq \mathsf{PSPACE}$ (see [13]). We do not know whether there is an $A \in \mathcal{C}$ such that A is PP-complete. It is not known whether there is an A such that $\mathsf{P}^{\mathsf{PP}^A} \neq \mathsf{NP}^{\mathsf{PP}^A}$, which would mean, in some sense, a difficulty in proving $A \in \mathcal{C}$ for an A which is PP-complete. In fact, it is not known whether there is an A such that $\mathsf{P}^{\mathsf{PP}^A} \neq \mathsf{PSPACE}$, and an affirmative answer would certainly give rise to interesting circuit lower bounds.

Acknowledgements

The author would like to thank his former MSc supervisor Václav Koubek from Charles University in Prague, and his current PhD supervisor Peter Jeavons from Oxford University for many helpful discussions and ongoing support. The author is grateful to anonymous referees for many useful comments on the presentation of the paper. Moreover, one of the referees spotted an error in the original proof of Theorem 4.1 (based on random oracles [7]), and provided a simpler proof using a diagonalization argument. This work was supported by EPSRC grant EP/F01161X/1.

References

[1] Allender, E., Hemachandra, L.A.: Lower Bounds for the Low Hierarchy. Journal of the ACM **39**(1) (1992) 234–251
[2] Baker, T.P., Gill, J., Solovay, R.: Relativizations of the P =? NP Question. SIAM Journal on Computing **4**(4) (1975) 431–442
[3] Balcázar, J.L.: Separating, Strongly Separating, and Collapsing Relativized Complexity Classes. In: Proceedings of the 11th Mathematical Foundations of Computer Science (MFCS'84). (1984) 1–16
[4] Balcázar, J.L., Book, R.V., Schöning, U.: Sparse Sets, Lowness and Highness. SIAM Journal on Computing **15**(3) (1986) 739–747
[5] Balcázar, J.L., Díaz, J., Gabarró, J.: Structural complexity I. Springer-Verlag New York, Inc. (1988)
[6] Balcázar, J.L., Díaz, J., Gabarró, J.: Structural complexity II. Springer-Verlag New York, Inc. (1990)
[7] Bennett, C.H., Gill, J.: Relative to a Random Oracle A, $\mathrm{P}^A \neq \mathrm{NP}^A \neq co\mathrm{NP}^A$ with Probability 1. SIAM Journal on Computing **10**(1) (1981) 96–113
[8] Fortnow, L.: The Role of Relativization in Complexity Theory. Bulletin of the EATCS **52** (1994) 229–243
[9] Hartmanis, J.: Solvable problems with conflicting relativizations. Bulletin of the EATCS **27** (1985) 40–48
[10] Hartmanis, J., Hemachandra, L.A.: On Sparse Oracles Separating Feasible Complexity Classes. Information Processing Letters **28**(6) (1988) 291–295
[11] Hemachandra, L.A.: The Strong Exponential Hierarchy Collapses. Journal of Computer and System Sciences **39**(3) (1989) 299–322
[12] Hemaspaandra, L.A., Jiang, Z., Rothe, J., Watanabe, O.: Boolean Operations, Joins, and the Extended Low Hierarchy. Theoretical Computer Science **205**(1-2) (1998) 317–327
[13] Hemaspaandra, L.A., Ogihara, M.: The complexity theory companion. Springer-Verlag New York, Inc. (2002)
[14] Toda, S.: PP is as Hard as the Polynomial-Time Hierarchy. SIAM Journal on Computing **20**(5) (1991) 865–877

[15] Živný, S.: Properties of oracle classes that collapse or separate complexity classes. Master's thesis, Vrije Universiteit in Amsterdam, The Netherlands (July 2005)