Disjoint NP-Pairs

(Extended Abstract)

Christian Glaßer [1] Alan L. Selman Samik Sengupta
Liyu Zhang

Department of Computer Science and Engineering,
University at Buffalo, Buffalo, NY 14260

Email: {cglasser,selman,samik,lzhang7}@cse.buffalo.edu

January 12, 2003

[1]Supported by a postdoctoral grant from the German Academic Exchange Service (Deutscher Akademischer Austauschdienst – DAAD).

Abstract

We study the question of whether the class $\mathcal{D}$ of disjoint pairs (A, B) of NP-sets contains a complete pair. The question relates to the question of whether optimal proof systems exist, and we relate it to the previously studied question of whether there exists a disjoint pair of NP-sets that is NP-hard. We show under reasonable hypotheses that nonsymmetric disjoint NP-pairs exist, which provides additional evidence for the existence of P-inseparable disjoint NP-pairs.

We construct an oracle relative to which the class of disjoint NP-pairs does not have a complete pair and an oracle relative to which complete pairs exist, but no pair is NP-hard. Both oracles satisfy additional interesting properties.

1 Introduction

We study the class $\mathcal{D}$ of disjoint pairs (A, B), where A and B are nonempty, disjoint sets belonging to NP. Such disjoint NP-pairs are interesting for at least two reasons. First, Grollmann and Selman [GS88] showed that the question of whether $\mathcal{D}$ contains P-inseparable disjoint NP-pairs is related to the existence of public-key cryptosystems. Second, Razborov [Raz94] and Pudlák [Pud01] demonstrated that these pairs are closely related to the theory of proof systems for propositional calculus. Specifically, Razborov showed that existence of an optimal propositional proof system implies existence of a complete pair for $\mathcal{D}$. Primarily in this paper we are interested in the question raised by Razborov [Raz94] of whether $\mathcal{D}$ contains a complete pair. We show connections between this question and earlier work on disjoint NP-pairs, and we exhibit an oracle relative to which $\mathcal{D}$ does not contain any complete pair.

From a technical point of view, disjoint pairs are simply an equivalent formulation of promise problems. There are natural notions of reducibilities between promise problems [ESY84, Sel88] that disjoint pairs inherit easily [GS88]. Hence, completeness and hardness notions follow naturally. We begin in the next section with these definitions, some easy observations, and a review of the known results.

The preliminary section tends to details concerning reductions between disjoint NP-pairs. In Section 3 we observe that if $\mathcal{D}$ does not contain a Turing-complete disjoint NP-pair, then $\mathcal{D}$ does not contain a disjoint NP-pair all of whose separators are Turing-hard for NP. The latter is a conjecture formulated by Even, Selman, and Yacobi [ESY84] and it has several known consequences: Public-key cryptosystems that are NP-hard to crack do not exist; $\mathrm{NP} \neq \mathrm{UP}$, $\mathrm{NP} \neq \mathrm{coNP}$, and $\mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$. Our main result in this section is an oracle X relative to which $\mathcal{D}$ does not contain a disjoint Turing-complete NP-pair and relative to which $\mathrm{P} \neq \mathrm{UP}$. Relative to X, by Razborov's result [Raz94], optimal propositional proof systems do not exist. P-inseparable disjoint NP-pairs exist relative to X, because $\mathrm{P} \neq \mathrm{UP}$ [GS88]. Most researchers believe that P-inseparable disjoint NP-pairs exist and we believe that no disjoint NP-pair has only NP-hard separators. Both of these properties hold relative to X. This is the first oracle relative to which both of these conditions hold simultaneously. Homer and Selman [HS92] obtained an oracle relative to which all disjoint NP-pairs are P-separable, so the conjecture of Even, Selman, and Yacobi holds relative to their oracle only for this trivial reason. Now lets say a few things about the construction of oracle X. Previous researchers have obtained oracles relative to which certain (promise) complexity classes do not have disjoint Turing-complete NP-pairs. However, the technique of Gurevich [Gur83], who proved that $\mathrm{NP} \cap \mathrm{coNP}$ has Turing-complete sets if and only if it has many-one-complete sets, does not apply. Neither does the technique of Hemaspaandra, Jain, and Vereshchagin [HJV93], who demonstrated, among other results, an oracle relative to which FewP does not have a Turing-complete set.

In Section 4 we show that the question of whether $\mathcal{D}$ contains a disjoint Turing-complete NP-pair has an equivalent natural formulation as an hypothesis about classes of single-valued partial functions. Section 5 studies *symmetric* disjoint NP-pairs. Pudlák [Pud01] defined a disjoint pair (A, B) to be symmetric if (A, B) is many-one reducible to (B, A). We easily show that P-separable implies symmetric. We give complexity-theoretic evidence of the existence of nonsymmetric disjoint NP-pairs. As a consequence, we obtain new ways to demonstrate existence of P-inseparable sets. Also, we use symmetry to show under reasonable hypotheses that many-one and Turing reducibilities differ for disjoint NP-pairs. (All reductions in this paper are polynomial-time-bounded.) Concrete candidates for P-inseparable disjoint NP-pairs come from problems in UP or in $\mathrm{NP} \cap \mathrm{coNP}$. Nevertheless, Grollmann and Selman [GS88] proved that the existence of P-inseparable disjoint NP-pairs implies the existence of P-inseparable disjoint NP-pairs, where both sets are NP-complete. Here we prove two analogous results. Existence of nonsymmetric disjoint NP-pairs implies existence of nonsymmetric disjoint NP-pairs, where both sets are NP-complete. If there exists a

many-one-complete disjoint NP-pair, then there exist such a pair, where both sets are NP-complete. Natural examples of nonsymmetric or $\leq_m^{pp}$- complete disjoint NP-pairs arise either from cryptography or from proof systems [Pud01]. Our theorems show that the existence of such pairs will imply that nonsymmetric (or $\leq_m^{pp}$-complete) disjoint NP-pairs exist where both sets of the pair are $\leq_m^p$-complete for NP.

Section 6 constructs an oracle O that possesses several interesting properties. Relative to O, many-one-complete NP-pairs exist. Therefore, while we expect that disjoint complete NP-pairs do not exist, this is not provable by relativizable techniques. P-inseparable NP-pairs exist relative to O, which we obtain by proving that nonsymmetric NP-pairs exist. The conjecture of Even, Selman and Yacobi holds relative to O. Therefore, while nonexistence of Turing-complete disjoint NP-pairs is a sufficient condition for this conjecture, the converse does not hold, even in a world in which P-inseparable pairs exist. Also, relative to O, there exists a P-inseparable set that is symmetric. Whereas nonsymmetric implies P-inseparable, again, the converse does not hold relative to O.

The construction of O involves some aspects that are unusual in complexity theory. We introduce undecidable requirements, and as a consequence, the oracle is undecidable. In particular, we need to define sets A and B, such that relative to O, the pair (A, B) is many-one complete. Therefore, we need to show that for every two nondeterministic, polynomial-time-bounded oracle Turing machines NM_i and NM_j, either $L(NM_i^O)$ and $L(NM_j^O)$ are not disjoint or there is a reduction from the disjoint pair $(L(NM_i^O), L(NM_j^O))$ to (A, B). We accomplish this as follows: Given NM_i, NM_j, and a finite initial segment X of O, we prove that either there is a finite extension Y of X such that for all oracles Z that extend Y,

$$L(NM_i^Z) \cap L(NM_j^Z) \neq \emptyset$$

or there is a finite extension Y of X such that for all oracles Z that extend Y,

$$L(NM_i^Z) \cap L(NM_j^Z) = \emptyset.$$

Then, we select the extension Y that exists. In this manner we *force* one of these two conditions to hold.

In the latter case, to obtain a reduction from the pair $(L(NM_i^O), L(NM_j^O))$ to (A, B) requires encoding information into the oracle O. The other conditions that we want O to satisfy require diagonalizations. In order to prove that there is room to diagonalize, we need to carefully control the number of words that must be reserved for encoding. This is a typical concern in oracle constructions, but even more so here. We manage this part of the construction by inventing a unique data structure that stores words reserved for the encoding, and then prove that we do not store too many such words.

2 Preliminaries

We fix the alphabet $\Sigma = \{0, 1\}$ and we denote the length of a word w by $|w|$. The set of all (resp., nonempty) words is denoted by Σ^* (resp., Σ^+). Let $\Sigma^{<n} \stackrel{df}{=} \{w \in \Sigma^* \mid |w| < n\}$, and define $\Sigma^{\leq n}$, $\Sigma^{=n}$, $\Sigma^{\geq n}$, and $\Sigma^{>n}$ analogously. For a set of words X let $X^{<n} \stackrel{df}{=} X \cap \Sigma^{<n}$, and define $X^{\leq n}$, $X^{=n}$, $X^{\geq n}$, and $X^{>n}$ analogously. For sets of words we take the complement w.r.t. Σ^*.

The set of (nonzero) natural numbers is denoted by $\mathbb{N}$ (by $\mathbb{N}^+$, respectively). Moreover, we fix a polynomial-time computable and polynomial-time invertible pairing function $\langle \cdot, \cdot \rangle : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$. For a function f, $\mathrm{dom}(f)$ denotes the domain of f.

2.1 Disjoint Pairs, Separators, and the ESY-Conjecture

Definition 2.1 *A* disjoint NP-pair *(*NP*-pair for short) is a pair of nonempty sets A and B such that $A, B \in$ NP and $A \cap B = \emptyset$. Let $\mathcal{D}$ denote the class of all disjoint* NP*- pairs.*

Given a disjoint NP-pair (A, B), a *separator* is a set S such that $A \subseteq S$ and $B \subseteq \overline{S}$; we say that S *separates* (A, B). Let $Sep(A, B)$ denote the class of all separators of (A, B). For disjoint NP-pairs (A, B), the fundamental question is whether $Sep(A, B)$ contains a set belonging to P. In that case the pair is P-*separable*; otherwise, the pair is P-*inseparable*. The following proposition summarizes the known results about P-separability.

Proposition 2.2 *1.* $\mathrm{P} \neq \mathrm{NP} \cap co\text{-}\mathrm{NP}$ *implies* NP *contains* P*-inseparable sets.*

2. $\mathrm{P} \neq \mathrm{UP}$ *implies* NP *contains* P*-inseparable sets [GS88].*

3. If NP *contains* P*-inseparable sets, then* NP *contains* NP*-complete* P*-inseparable sets [GS88].*

While it is probably the case that NP contains P-inseparable sets, there is an oracle relative to which $\mathrm{P} \neq \mathrm{NP}$ and P-inseparable sets in NP do not exist [HS92]. So $\mathrm{P} \neq \mathrm{NP}$ probably is not a sufficiently strong hypothesis to show existence of P-inseparable sets in NP.

Definition 2.3 *Let* (A, B) *be a disjoint* NP*-pair.*

1. (A, B) *is* NP-hard *if every separator of* (A, B) *is* NP-hard*.*

2. (A, B) *is* uniformly NP-hard *if there is a deterministic polynomial-time oracle Turing machine* M *such that for every* $A \in Sep(A, B)$, $\mathrm{SAT} \leq_T^p A$ *via* M*.*

Grollmann and Selman [GS88] show that NP-hard implies uniformly NP-hard, i.e., both statements of the definition are equivalent. Even, Selman, and Yacobi [ESY84] conjectured that there does not exist a disjoint NP-pair (A, B) such that all separators of (A, B) are $\leq_T^p$ hard for NP.

Conjecture 2.4 ([ESY84]) *There do not exist disjoint* NP*-pairs that are* NP*-hard.*

If Conjecture 2.4 holds, then no public-key cryptosystem is NP-hard to crack. This conjecture is a strong hypothesis with the following known consequences. In Section 3 we show a sufficient condition for Conjecture 2.4 to hold.

Proposition 2.5 ([ESY84, GS88, Sel94]) *If Conjecture 2.4 holds, then* $\mathrm{NP} \neq \mathrm{coNP}$, $\mathrm{NP} \neq \mathrm{UP}$, *and* $\mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$.

2.2 Reductions for Disjoint Pairs

We review the natural notions of reducibilities between disjoint pairs [GS88].

Definition 2.6 (non-uniform reductions for pairs) *Let* (A, B) *and* (C, D) *be disjoint pairs.*

1. (A, B) *is* many-one reducible in polynomial time *to* (C, D), $(A, B) \leq_m^{pp} (C, D)$*, if for every separator* $T \in Sep(C, D)$*, there exists a separator* $S \in Sep(A, B)$ *such that* $S \leq_m^p T$*.*

2. (A, B) *is* Turing reducible in polynomial time *to* (C, D), $(A, B) \leq_T^{pp} (C, D)$*, if for every separator* $T \in Sep(C, D)$*, there exists a separator* $S \in Sep(A, B)$ *such that* $S \leq_T^p T$*.*

Definition 2.7 (uniform reductions for pairs) *Let* (A, B) *and* (C, D) *be disjoint pairs.*

1. (A, B) *is* uniformly many-one reducible in polynomial time *to* (C, D), $(A, B) \leq^{pp}_{um} (C, D)$, *if there exists a polynomial-time computable function* f *such that for every separator* $T \in Sep(C, D)$, *there exists a separator* $S \in Sep(A, B)$ *such that* $S \leq^{p}_{m} T$ *via* f.

2. (A, B) *is* uniformly Turing reducible in polynomial time *to* (C, D), $(A, B) \leq^{pp}_{uT} (C, D)$, *if there exists a polynomial-time oracle Turing machine* M *such that for every separator* $T \in Sep(C, D)$, *there exists a separator* $S \in Sep(A, B)$ *such that* $S \leq^{p}_{T} T$ *via* M.

If f and M are as above, then we also say that $(A, B) \leq^{pp}_{um} (C, D)$ via f and $(A, B) \leq^{pp}_{uT} (C, D)$ via M. Observe that if $(A, B) \leq^{pp}_{m} (C, D)$ and (C, D) is P-separable, then so is (A, B) (and the same holds for $\leq^{pp}_{T}$, $\leq^{pp}_{um}$, and $\leq^{pp}_{uT}$). We retain the promise problem notation in order to distinguish from reducibilities between sets. Grollmann and Selman proved that Turing reductions and uniform Turing reductions are equivalent.

Proposition 2.8 ([GS88]) $(A, B) \leq^{pp}_{T} (C, D) \Leftrightarrow (A, B) \leq^{pp}_{uT} (C, D)$ *for all disjoint pairs* (A, B) *and* (C, D).

In order to obtain the corresponding theorem for $\leq^{pp}_{um}$, we can adapt the proof of Proposition 2.8, but a separate argument is required. We omit the proof in this version.

Theorem 2.9 $\leq^{pp}_{m} = \leq^{pp}_{um}$.

We obtain the following useful characterization of many-one reductions.

Theorem 2.10 $(A, B) \leq^{pp}_{m} (C, D)$ *if and only if there exists a polynomial-time computable function* f *such that* $f(A) \subseteq C$ *and* $f(B) \subseteq D$.

Proof By Theorem 2.9 there is a polynomial-time computable function f such for every $A \in Sep(S, T)$, $f^{-1}(A) \in Sep(Q, R)$. That is, if $S \subseteq A$ and $T \subseteq \overline{A}$, then $Q \subseteq f^{-1}(A)$ and $R \subseteq \overline{f^{-1}(A)}$, which implies that $f(Q) \subseteq A$ and $f(R) \cap A = \emptyset$. Well, $S \in Sep(S, T)$. So $f(Q) \subseteq S$. Also, $\overline{T} \in Sep(S, T)$. So $f(R) \cap \overline{T} = \emptyset$. That is, $f(R) \subseteq T$. The converse is immediate. □

3 Complete Disjoint NP-Pairs

Keeping with common terminology, a disjoint pair (S, T) is $\leq^{pp}_{m}$-complete ($\leq^{pp}_{T}$- complete) for the class $\mathcal{D}$ if $(S, T) \in \mathcal{D}$ and for every disjoint pair $(Q, R) \in \mathcal{D}$, $(Q, R) \leq^{pp}_{m} (S, T)$ ($(Q, R) \leq^{pp}_{T} (S, T)$, respectively).

Consider the following assertions:

1. $\mathcal{D}$ does not have a $\leq^{pp}_{T}$-complete disjoint pair.

2. $\mathcal{D}$ does not have a $\leq^{pp}_{m}$-complete disjoint pair.

3. $\mathcal{D}$ does not contain a disjoint pair all of whose separators are $\leq^{p}_{T}$-hard for NP (i.e., Conjecture 2.4 holds).

4. $\mathcal{D}$ does not contain a disjoint pair all of whose separators are $\leq^{p}_{m}$-hard for NP.

Assertions 1 and 2 are possible answers to the question raised by Razborov [Raz94] of whether $\mathcal{D}$ contains complete disjoint pairs. Assertion 3 is Conjecture 2.4. Assertion 4 is the analog of this conjecture using many-one reducibility.

We can dispense with Assertion 4 immediately, for it is equivalent to $\mathrm{NP} \neq \mathrm{coNP}$.

Proposition 3.1 $\mathrm{NP} \neq \mathrm{coNP}$ *if and only if* $\mathcal{D}$ *does not contain a disjoint pair all of whose separators are* $\leq_m^p$*-hard for* NP.

Proof If $\mathrm{NP} = \mathrm{coNP}$, then $(\mathrm{SAT}, \overline{\mathrm{SAT}})$ is a disjoint pair in $\mathcal{D}$ all of whose separators are $\leq_m^p$-hard for NP.

To show the other direction, consider the disjoint pair $(A, B) \in \mathcal{D}$ and assume that all of its separators are $\leq_m^p$-hard for NP. Since $\overline{B}$ is a separator of (A, B), SAT $\leq_m^p \overline{B}$. Therefore, $\overline{\mathrm{SAT}} \leq_m^p B$, implying that $\overline{\mathrm{SAT}} \in \mathrm{NP}$. Thus, NP = coNP. □

Proposition 3.2 *Assertion 1 implies Assertions 2 and 3. Assertions 2 and 3 imply Assertion 4.*

This Proposition states, in part, that Assertion 1 is so strong as to imply Conjecture 2.4.

Proof It is trivial that Assertion 1 implies Assertion 2 and Assertion 3 implies Assertion 4.

We prove that Assertion 1 implies Assertion 3. Assume Assertion 3 is false and let $(S, T) \in \mathcal{D}$ such that all separators are NP-hard. We claim that (S, T) is $\leq_T^{pp}$-complete for $\mathcal{D}$. Let (Q, R) belong to $\mathcal{D}$. Let L be an arbitrary separator of (S, T). Note that L is NP-hard and $Q \in \mathrm{NP}$. So $Q \leq_T^p L$. Since Q is a separator of (Q, R), this demonstrates that $(Q, R) \leq_T^{pp} (S, T)$.

Similarly, we prove that Assertion 2 implies Assertion 4. In this case, every separator L of (S, T) is $\leq_m^p$-hard for NP. So $Q \leq_m^p L$. Therefore, $(Q, R) \leq_m^{pp} (S, T)$. □

Homer and Selman [HS92] constructed an oracle relative to which $\mathrm{P} \neq \mathrm{NP}$ and every disjoint NP-pair is P-separable. Relative to this oracle, Assertion 3 holds and Assertions 1 and 2 are false. To see this, let (A, B) be an arbitrary disjoint NP-pair. We show that (A, B) is both $\leq_T^{pp}$-complete and $\leq_m^{pp}$- complete. For any other pair $(C, D) \in \mathcal{D}$, since (C, D) is P-separable, there is a separator S of (C, D) that is in P. Therefore, for any separator L of (A, B), S trivially $\leq_m^p$-reduces and $\leq_T^p$-reduces to L. So $(C, D) \leq_m^{pp} (A, B)$ and $(C, D) \leq_T^{pp} (A, B)$.

In Theorem 3.3 we construct an oracle relative to which Assertion 1 is true, and at the same time, $\mathrm{P} \neq \mathrm{UP}$. Therefore, by Proposition 3.2, with respect to the oracle in Theorem 3.3, all of the following properties hold:

1. $\mathcal{D}$ does not have a $\leq_T^{pp}$-complete disjoint pair.
2. Conjecture 2.4 holds; so $\mathrm{UP} \neq \mathrm{NP}, \mathrm{NP} \neq \mathrm{coNP}, \mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$ and NP-hard public-key cryptosystems do not exist [ESY84, Sel94].
3. $\mathrm{P} \neq \mathrm{UP}$; therefore P-inseparable disjoint NP- pairs exist [GS88].
4. Optimal propositional proof systems do not exist [Raz94].
5. There is a tally set $T \in \mathrm{coNP} - \mathrm{NP}$ and a tally set $T' \in \mathrm{coNE} - \mathrm{E}$ [BDG98].

Theorem 3.3 *There exists an oracle* X *such that* $\mathcal{D}^X$ *does not have a* $\leq_T^{pp,X}$*-complete pair and* $\mathrm{P}^X \neq \mathrm{UP}^X$.

4 Function Classes and Disjoint Pairs

We show that there exists a Turing-complete disjoint NP-pair if and only if NPSV contains a partial function that is Turing-hard for NPSV. We know already that the conjecture of Even, Selman, and Yacobi holds if and only if NPSV does not contain an NP-hard partial function. Recall [Sel94] that NPSV is the set of all partial, single-valued functions computed by nondeterministic polynomial-time bounded transducers.

If g is a single-valued total function, then we define $M[g]$, the single-valued partial function computed by M with oracle g as follows: $x \in dom(M[g])$ if and only if M reaches an accepting state on input x. In this case, $M[g](x)$ is the final value of M's output tape.

The literature contains two different definitions of reductions between partial functions, because one must decide what to do in case a query is made to the oracle function when the query is not in the domain of the oracle function. Fenner et al [FHOS97] determined that in this case the value returned should be a special symbol $\perp$. Selman [Sel94] permits the value returned in this case to be arbitrary, which is the standard paradigm for promise problems. Here we use the promise problem definition of Selman [Sel94].

Definition 4.1 *f is* Turing reducible (as a promise problem) to *g in polynomial time if for some deterministic oracle transducer M, for every single-valued total extension g' of g, $M[g']$ is an extension of f.*

Here, if the query q belongs to the domain of g, then the oracle returns a value of $g(q)$.

Definition 4.2 *A partial function f is* NP–hard *if for every single-valued total extension f' of f, the* NP–*hard problem SAT is Turing reducible to f'.*

Theorem 4.3 NPSV *contains a $\leq_T^{pp}$-complete partial function $\Leftrightarrow$ $\mathcal{D}$ contains a $\leq_T^{pp}$-complete pair.*

Proof For any $f \in$ NPSV, define the following sets.

$$R_f = \{\langle x, y\rangle | x \in dom(f), y \leq f(x)\} \tag{1}$$

$$S_f = \{\langle x, y\rangle | x \in dom(f), y > f(x)\} \tag{2}$$

Note that (R_f, S_f) is an NP-pair.

Claim. For every separator A of (R_f, S_f), there is a single-valued total extension f' of f such that $f' \leq_T^p A$.

Proof of Claim. Consider the following oracle transducer T that computes f' with oracle A. On input x, if $x \in dom(f)$, T determines the values of $f(x)$ by making repeated queries to A. Note that for $x \in dom(f)$ and for any y, if $y \leq f(x)$, then $\langle x, y\rangle \in R_f$, and if $y > f(x)$, then $\langle x, y\rangle \in S_f$. If $x \notin dom(f)$, T outputs 0. Clearly, T computes some single-valued total extension of f. This proves the claim.

Let f be a complete function for NPSV and assume that A separates R_f and S_f. By the previous claim, there is a single-valued total extension f' of f such that $f' \leq_T^{pp} A$.

Let $(U, V) \in \mathcal{D}$. We want to show that $(U, V) \leq_T^{pp} (R_f, S_f)$. Define

$$g(x) = \begin{cases} 0, & \text{if } x \in U \\ 1, & \text{if } x \in V \\ \uparrow, & \text{otherwise.} \end{cases}$$

$g \in$ NPSV, so $g \leq_T^{pp} f$. By definition, there is a deterministic oracle transducer M such that $M[f'] = g'$ is a single-valued total extension of g.

Define $L = \{x : g'(x) = 0\}$. It is easy to see that $L \leq_T^p g'$. Also note that $U \subseteq L$ and $V \subseteq \overline{L}$, and therefore, L separates U and V. Then the following sequence of reductions show that $L \leq_T^p A$.

$$L \leq_T^p g' \leq_T^{pp} f' \leq_T^p A.$$

Thus, for every separator A of (R_f, S_f), there is a separator L of (U, V) such that $L \leq_T^p A$. Therefore, (R_f, S_f) is $\leq_T^{pp}$-complete for $\mathcal{D}$.

For the other direction, assume that (U, V) is $\leq_T^{pp}$- complete for $\mathcal{D}$. Define the following function.

$$f(x) = \begin{cases} 0, & \text{if } x \in U \\ 1, & \text{if } x \in V \\ \uparrow, & \text{otherwise.} \end{cases}$$

Clearly, $f \in \mathrm{NPSV}$.

Let f' be a single-valued total extension of f, and let $L = \{x | f'(x) = 0\}$. Clearly, $L \leq_T^p f'$. Also, since $U \subseteq L$ and $V \subseteq \overline{L}$, L is a separator of (U, V).

We want to show that for any $g \in \mathrm{NPSV}$, $g \leq_T^{pp} f$. Consider the NP-pair (R_g, S_g) for the function g as defined in Equations 1 and 2. As noted in the claim, there is a single-valued total extension g' of g such that $g' \leq_T^p A$. Also, there is a separator A of (R_g, S_g) such that $A \leq_T^p L$, since L is a separator of the $\leq_T^{pp}$-complete NP-pair (U, V).

Therefore, the following sequence of reductions show that f is complete for NPSV.

$$g' \leq_T^p A \leq_T^p L \leq_T^p f'.$$

□

Corollary 4.4 *1. Let $f \in \mathrm{NPSV}$ be $\leq_T^{pp}$-complete for NPSV. Then (R_f, S_f) is $\leq_T^{pp}$-complete for disjoint pairs of NP sets.*

2. If (U, V) is a $\leq_m^{pp}$-complete NP-pair, then $f_{U,V}$ is complete for NPSV, where

$$f_{U,V}(x) = \begin{cases} 0, & \textit{if } x \in U \\ 1, & \textit{if } x \in V \\ \uparrow, & \textit{otherwise.} \end{cases}$$

3. Relative to the oracle in Theorem 3.3, NPSV does not have a $\leq_T^{pp}$-complete partial function.

5 Nonsymmetric Pairs and Separation of Reducibilities

Pudlák [Pud01] defined a disjoint pair (A, B) to be *symmetric* if $(B, A) \leq_m^{pp} (A, B)$. Otherwise, (A, B) is *nonsymmetric*. In this section we give complexity-theoretic evidence of the existence of nonsymmetric disjoint NP-pairs. As a consequence, we obtain new ways to demonstrate existence of P-inseparable sets and we show that $\leq_m^{pp}$ and $\leq_T^{pp}$ reducibilities differ for NP-pairs.

A set L is *P-printable* if there is $k \geq 1$ such that all elements of L up to length n can be printed by a deterministic Turing machine in time $n^k + k$ [HY84, HIS85]. Every P-printable set is sparse and belongs to P. A set A is *P-printable-immune* if no infinite subset of A is P-printable.

A set L is *p-selective* if there is a function $f \in \mathrm{FP}$ such that for every $x, y \in \Sigma^*$, $f(x, y) \subseteq \{x, y\}$, and $\{x, y\} \cap L \neq \emptyset \Rightarrow f(x, y) \in L$ [Sel79].

Proposition 5.1 *1. (A, B) is symmetric if and only if (B, A) is symmetric.*

2. (A, B) is P-separable $\Rightarrow$ (A, B) is symmetric.

Proof

1. If (A, B) is symmetric, then $(B, A) \leq_m^{pp} (A, B)$, i.e., there is $f \in \mathrm{FP}$ such that $f(A) \subseteq B$ and $f(B) \subseteq A$. Clearly the same function f reduces (A, B) to (B, A).

2. Let (A, B) be a P-separable disjoint NP-pair. Fix $a \in A$ and $b \in B$ and let the separator be $S \in \mathrm{P}$. Consider the following polynomial-time function f. On input x, if $x \in S$, f outputs b; otherwise, f outputs a. Therefore, for every $x \in A$, $x \in S \Rightarrow f(x) = b \in B$ and $\forall x \in B$, $x \notin S \Rightarrow f(x) = a \in A$. Therefore, $(A, B) \leq_m^{pp} (B, A)$, i.e., (A, B) is symmetric.

□

We will show the existence of a nonsymmetric NP-pair under certain hypotheses. Due to the following proposition, that will separates $\leq_m^{pp}$ and $\leq_T^{pp}$ reducibilities.

Proposition 5.2 *1. If (A, B) is a nonsymmetric disjoint* NP*-pair, then $(B, A) \not\leq_m^{pp} (A, B)$*

2. For any disjoint NP*-pair (A, B), $(B, A) \leq_T^{pp} (A, B)$*

Proof (1) follows from the definition of symmetric pairs. For (2), observe that for any S separating A and B, $\overline{S}$ separates B and A and for any set S, $\overline{S} \leq_T^p S$. □

We will use the following proposition in a crucial way to show the existence of nonsymmetric NP-pairs. In other words, we will seek to obtain an NP-pair (A, B) such that A or B is p-selective, but (A, B) is not P-separable.

Proposition 5.3 *For any* NP*-pair (A, B), if either A or B is p-selective, then (A, B) is symmetric if and only if (A, B) is P- separable.*

Proof We know from Proposition 5.1 that if (A, B) is P-separable, then it is symmetric. Now assume that (A, B) is symmetric via some function f and assume (without loss of generality) that A is p-selective and the P-selector function is g. The following algorithm M separates A and B. On input x, M runs g on the strings $(x, f(x))$, and accepts x if and only if g outputs x. If $x \in A$, $f(x) \in B$ and therefore, g has to output x. On the other hand, if $x \in B$, then $f(x) \in A$ and g will output $f(x)$ and M will reject x. Therefore, $A \subseteq L(M) \subseteq \overline{B}$. □

Now we give evidence showing the existence of nonsymmetric NP-pairs.

Theorem 5.4 *If* $\mathrm{E} \neq \mathrm{NE} \cap \mathrm{coNE}$*, then there is a set $A \in \mathrm{NP} \cap \mathrm{coNP}$ such that $(A, \overline{A})$ is not symmetric.*

Proof If $\mathrm{E} \neq \mathrm{NE} \cap \mathrm{coNE}$, then there is a tally set $T \in \mathrm{NP} \cap \mathrm{coNP} - \mathrm{P}$. From Selman [Sel79, Theorem 5], we know that the existence of such a tally set implies that there is a p-selective set $A \in \mathrm{NP} \cap \mathrm{coNP} - \mathrm{P}$. Clearly, $(A, \overline{A})$ is not P-separable. Hence, by Proposition 5.3, $(A, \overline{A})$ is nonsymmetric. □

As a corollary, we obtain that if $\mathrm{E} \neq \mathrm{NE} \cap \mathrm{coNE}$, then there is a set $A \in \mathrm{NP} \cap \mathrm{coNP}$ such that $(A, \overline{A}) \not\leq_m^{pp} (\overline{(A)}, A)$, yet clearly $(A, \overline{A}) \leq_T^{pp} (\overline{(A)}, A)$.

We will show that the hypotheses in Theorem 5.5 imply the existence of a nonsymmetric NP-pair. Note that the hypotheses in this theorem are similar to those studied by Fortnow, Pavan and Selman [FPS01] and Pavan and Selman [PS01]; however, our hypotheses are stronger than the former and weaker than the latter. We omit the proof in this version.

Theorem 5.5 *The following are equivalent:*

1. *There is an* UP*-machine* N *that accepts* 0^* *such that no polynomial-time machine can output infinitely many accepting computations of* N.
2. *There is an infinite set* S *in* UP *accepted by an* UP*-machine* M *such that* S *has exactly one string of every length and no polynomial-time machine can compute infinitely many accepting computations of* M.
3. *There is an almost-always one-one one-way function* f *such that range(*f*) =* 0^*.
4. *There is a language* $L \in \mathrm{P}$ *that has exactly one string of every length and* L *is P-printable immune.*
5. *There is a language* $L \in \mathrm{UP}$ *that has exactly one string of every length and* L *is P-printable immune.*

The Appendix contains the proof of the following theorem.

Theorem 5.6 *Each of the hypotheses stated in Theorem 5.5 implies the existence of nonsymmetric* NP*-pairs.*

If the hypotheses stated in Theorem 5.5 hold, then there exists a disjoint NP-pair (A, B) so that $(A, B) \not\leq_m^{pp} (B, A)$ while $(A, B) \leq_T^{pp} (B, A)$.

Grollmann and Selman [GS88] proved that the existence of P-inseparable NP-pairs implies the existence of P-inseparable pairs where both sets of the pair are NP-complete. The following results are in the same spirit. We note that natural examples of nonsymmetric (or $\leq_m^{pp}$-complete) disjoint NP-pairs arise either from cryptography or from proof systems. However, the following theorems show that the existence of such pairs will imply that nonsymmetric (or $\leq_m^{pp}$-complete) disjoint NP-pairs exist where both sets of the pair are $\leq_m^p$-complete for NP. These results are proven in the appendix.

Theorem 5.7 *There exists a nonsymmetric disjoint* NP*-pair* (A, B) *if and only if there exists a nonsymmetric disjoint* NP*-pair* (C, D) *where both* C *and* D *are* $\leq_m$*-complete for* NP.

Theorem 5.8 *There exists an disjoint* NP*-pair* (A, B) *that is* $\leq_m^{pp}$*-complete if and only if there exists a disjoint* NP*-pair* (C, D) *that is* $\leq_m^{pp}$*-complete where both* C *and* D *are* $\leq_m$*-complete for* NP.

6 Many-One Complete NP-Pairs Relative to an Oracle

In this section we construct an oracle O that possesses several interesting properties. Relative to O, many-one complete NP-pairs exist. Therefore, while we expect that complete NP-pairs do not exist, this is not provable by relativizable techniques. Since nonexistence of $\leq_m^{pp}$-complete NP-pairs implies Conjecture 2.4, it is natural to ask whether the converse holds. In this section we construct an oracle relative to which the converse is false. Relative to this oracle all of the following properties hold:

1. There exist $\leq_m^{pp}$-complete NP-pairs.

2. There exist nonsymmetric NP-pairs.

3. Conjecture 2.4 holds, and therefore also $\mathrm{UP} \neq \mathrm{NP} \neq \mathrm{coNP}$ and $\mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$.

4. There exist P-inseparable NP-pairs that are symmetric.

Here we show that there is a relativized world where both Conjecture 2.4 holds and P-inseparable NP-pairs exist, yet $\leq_m^{pp}$-complete NP-pairs exist. Also note that our oracle is natural in the sense that, apart from the existence of $\leq_m^{pp}$-complete NP-pairs, all of its properties are expected for the unrelativized case[1].

Property 1 requires coding information into the oracle. Properties 2 and 3 require diagonalizations. (Property 4 will be easy to obtain.) Unlike several previous oracle constructions (e.g., [BGS75, Rac82, HS92]) that balance coding requirements and diagonalizations, we cannot start with a PSPACE-complete oracle, because that would make it difficult to obtain nonsymmetric NP-pairs.

Theorem 6.1 *There exists an oracle relative to which the following holds:*

(i) There exist $\leq_m^{pp}$-complete NP-pairs.

(ii) There exist nonsymmetric NP-pairs.

(iii) Conjecture 2.4 holds.

Corollary 6.2 *The oracle O from Theorem 6.1 has the following additional properties.*

(i) $\mathrm{UP}^O \neq \mathrm{NP}^O \neq \mathrm{coNP}^O$ *and* $\mathrm{NPMV}^O \not\subseteq_c \mathrm{NPSV}^O$

(ii) There exists a $\leq_m^{pp}$-complete NP^O-pair (A, B) that satisfies the following:

- *For every NP^O-pair (E, F) there exists an $f \in \mathrm{FP}$ with $E \leq_m^p A$ via f and $F \leq_m^p B$ via f.*
- *(A, B) is P^O-inseparable but symmetric.*

Acknowledgements. The authors thank Avi Wigderson for informing them of the paper by Ben-David and Gringauze [BDG98].

[1] We believe that statement 4 holds since Pudlák [Pud01] shows that the canonical pair of resolution is symmetric, and we expect that this pair is P-inseparable.

References

[BDG98] S. Ben-David and A. Gringauze. On the existence of propositional proof systems and oracle-relativized propositional logic. Technical Report 5, Electronic Colloquium on Computational Complexity, 1998.

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

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

[FHOS97] S. Fenner, S. Homer, M. Ogihara, and A. Selman. Oracles that compute values. *SIAM Journal on Computing*, 26:1043–1065, 1997.

[FPS01] L. Fortnow, A. Pavan, and A. Selman. Distributionally hard languages. *Theory of Computing Systems*, 34:245–261, 2001.

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

[Gur83] Y. Gurevich. Algebras of feasible functions. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, pages 210–214. IEEE Computer Society Press, 1983.

[HIS85] J. Hartmanis, N. Immerman, and V. Sewelson. Sparse sets in NP - P: EXPTIME versus NEXPTIME. *Information and Control*, 65:158–181, 1985.

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

[HS92] S. Homer and A. Selman. Oracles for structural properties: The isomorphism problem and public-key cryptography. *Journal of Computer and System Sciences*, 44(2):287–301, 1992.

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

[PS01] A. Pavan and A. Selman. Separation of NP-completeness notions. In *Proceedings 16th Computational Complexity*. IEEE Computer Society, 2001.

[Pud01] P. Pudlák. On reducibility and symmetry of disjoint NP-pairs. In *Proceedings 26th International Symposium on Mathematical Foundations of Computer Science*, volume 2136 of *Lecture Notes in Computer Science*, pages 621–632. Springer-Verlag, Berlin, 2001.

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

[Raz94] A. Razborov. On provably disjoint NP-pairs. Technical Report TR94-006, Electronic Colloquium on Computational Complexity, 1994.

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial-time reducibilities on NP. *Mathematical Systems Theory*, 13:55–65, 1979.

[Sel88] A. Selman. Promise problems complete for complexity classes. *Information and Computation*, 78:87–98, 1988.

[Sel94] A. Selman. A taxonomy on complexity classes of functions. *Journal of Computer and System Sciences*, 48:357–381, 1994.

Appendix

Theorem 3.3 *There exists an oracle X such that $\mathcal{D}^X$ does not have a $\leq_T^{pp,X}$-complete disjoint pair and* $\mathrm{P}^X \neq \mathrm{UP}^X$.

Since oracle access requires full access, we define the following notions.

Definition 3.4 *For any set X, a pair of disjoint sets (A, B) is* polynomial time Turing reducible relative to X *($\leq_T^{pp,X}$) to a pair of disjoint sets (C, D) if for any separator S that separates (C, D), there exists a polynomial time deterministic oracle Turing Machine M such that $M^{S\oplus X}$ accepts a language that separates (A, B).*

Definition 3.5 *For any set X, define $\mathcal{D}^X = \{(A, B)|A \in \mathrm{NP}^X,\ B \in \mathrm{NP}^X\ and\ A \cap B = \emptyset\}$. $\mathcal{D}^X$ has $\leq_T^{pp,X}$-complete set for $\mathcal{D}^X$ if $\exists (C, D) \in \mathcal{D}^X$ and for all $(A, B) \in \mathcal{D}^X$, $(A, B) \leq_T^{pp,X} (C, D)$.*

Similarly, $\mathcal{D}^X$ has $\leq_T^{pp}$-complete set for $\mathcal{D}^X$ if $\exists (C, D) \in \mathcal{D}^X$ and for all $(A, B) \in \mathcal{D}^X$, $(A, B) \leq_T^{pp}$ (C, D).

However, the following proposition shows that if there exists a pair that is Turing-complete relative to X for $\mathcal{D}^X$, then there is a pair that is Turing complete for $\mathcal{D}^X$, where the reduction between the separators does not access the oracle.

Proposition 3.6 *For any set X, $\mathcal{D}^X$ has a Turing-complete disjoint pair relative to X if and only if $\mathcal{D}^X$ has a Turing-complete disjoint pair.*

Proof The *if* direction is trivial. We only show the *only if* direction. Suppose (C, D) is Turing complete relative to X for $\mathcal{D}^X$. We claim that $(C \oplus X, D \oplus \overline{X})$ [2] is a Turing-complete pair of disjoint sets for $\mathcal{D}^X$. Consider any $(A, B) \in \mathcal{D}^X$. Let S' be any set that separates $(C \oplus X, D \oplus \overline{X})$. Define $S = \{x|0x \in S'\}$, then S separates (C, D) and $S' = S \oplus X$. Since (C, D) is Turing-complete relative to X for $\mathcal{D}^X$, there must exist a polynomial time deterministic oracle Turing Machine M using oracle X and S that separates (A, B). Hence we can obtain a polynomial time deterministic oracle Turing machine M' using oracle S' that separates (A, B): M' on any input does exactly the same as M except whenever M queries some string x to oracle S, M' queries $0x$ to oracle S' and whenever M queries some string x to oracle X, M' queries $1x$ instead. It is easy to see that M' gets the same answer as M for each query, hence accepts the same language as M does, and M' witnesses that $(A, B) \leq_{uT}^{pp} (C \oplus X, D \oplus \overline{X})$. So $(C \oplus X, D \oplus \overline{X})$ is a Turing-complete disjoint pair for $\mathcal{D}^X$. □

It is easy to see that Theorem 3.3 can be obtained by modifying the proof of the following theorem.

Theorem 3.7 *There exists an oracle X such that $\mathcal{D}^X$ does not have a $\leq_T^{pp,X}$-complete disjoint pair.*

Proof Since Proposition 2.8 and Proposition 3.6 relativizes to all oracles, it suffices to show there is no $\leq_T^{pp}$-complete pair of disjoint sets in $\mathcal{D}^X$ under uniform Turing reduction. So we will construct an oracle X such that for every $(C, D) \in \mathcal{D}^X$ there exists a disjoint pair $(A, B) \in \mathcal{D}^X$, $(A, B) \not\leq_{uT}^{pp} (C, D)$.

Suppose $\{M_k\}_k$ and $\{N_i\}_i$ are respectively enumerations of deterministic and non-deterministic polynomial time oracle Turing machines. Let r_k and p_i be the corresponding polynomial time bounds for M_k

[2] $A \oplus B \stackrel{\text{def}}{=} \{0x|x \in A\} \cup \{1y|y \in B\}$

and N_i. For any r, s, d, let $\Sigma^d_{rs} = 0^r10^s1\Sigma^d$ and $l^d_{rs} = r + s + d + 2$, the length of strings in Σ^d_{rs}. For each i, j, define

$$A_{ij} = \{0^n | \exists x |x| = n \wedge 0^i10^j10x \in X\}$$

and

$$B_{ij} = \{0^n | \exists x |x| = n \wedge 0^i10^j11x \in X\}.$$

We construct the oracle X in stages. Initially we set $X = \emptyset$. In Stage $m = \langle i, j, k\rangle$, we will put strings from Σ^{n+1}_{ij} into X such that either $L(N_i) \cap L(N_j) \neq \emptyset$ or (A_{ij}, B_{ij}) is not uniformly Turing reducible to $(L(N_i), L(N_j))$ via M_k, where $n = n_m$ is some number chosen at Stage m. We will show later that the construction above ensures that for any i and j, $(L(N_i), L(N_j))$ is not Turing-complete for $\mathcal{D}^X$.

Let X_m be the oracle before Stage m. $X_0 = \emptyset$. For the current stage $m = \langle i, j, k\rangle$, let $m-1 = \langle i', j', k'\rangle$ and $m + 1 = \langle i'', j'', k''\rangle$. We choose some number $n = n_m$ such that n is minimal and all the following hold (For Stage 0, we just set $n_0 = 1$.):

- $n > n_{m-1}$
- $l^{n+1}_{ij} > l^{n_{m-1}+1}_{i'j'}$
- $l^{n+1}_{ij} > \max(p_{i'}(n_{m-1}), p_{j'}(n_{m-1}))$
- $l^{n+1}_{ij} > \max(p_{i'}(r_{k'}(n_{m-1})), p_{j'}(r_{k'}(n_{m-1})))$
- $2^n > r_k(n)p_i(r_k(n))p_j(r_k(n))$

Obviously, l^{n+1}_{ij} and $l^{n_{m-1}+1}_{i'j'}$ are, respectively, the length of strings we add into the oracle at Stage m and $m - 1$.

Suppose for some $S \subseteq \Sigma^{n+1}_{ij}$, $L(N_i) \cap L(N_j) \neq \emptyset$ using oracle $X_m \cup S$. Then let $s \in L(N_i^{X_m \cup S}) \cap L(N_j^{X_m \cup S})$. Define $X_{m+1} = X_m \cup S$, $n_m = |s|$ and go to the next stage $m + 1$. From now on we will skip any later Stage l, where $l = \langle i, j, k'''\rangle$.

Otherwise, we have that

$$\text{for any } S \subseteq \Sigma^{n+1}_{ij}, L(N_i) \cap L(N_j) = \emptyset \text{ using oracle } X_m \cup S. \tag{3}$$

We will consider the computation of M_k on 0^n in this case and try to add a string in Σ^{n+1}_{ij} to the oracle so that either

$$0^{n_m} \in A_{ij} \; and \; 0^{n_m} \notin L(M_k^{L(N_i^{X_{m+1}}) \cup Q})$$

or

$$0^{n_m} \in B_{ij} \; and \; 0^{n_m} \in L(M_k^{L(N_i^{X_{m+1}}) \cup Q})$$

after Stage m.

Note that this would imply (A_{ij}, B_{ij}) does not reduce to $(L(N_i^{X_{m+1}}), L(N_j^{X_{m+1}}))$ via M_k.

The difficulty rises mainly from the fact that if we want to preserve the computation of M_k on 0^n in a straightforward way (by reserving all strings in Σ^{n+1}_{ij} that are queried) to do the diagonalization, we will end up with having to reserve all strings in Σ^{n+1}_{ij}, which leaves no room for the diagonalization. Fortunately, we can do better by the following lemma.

Lemma 3.8 *Let M and N be nondeterministic polynomial-time oracle Turing machines with polynomial time bounds p_M and p_N respectively. Let Y be an oracle and $q \in \Sigma^*$, $|q| = n$.*

Then, for any set T with $\|T\| > p_M(|q|)p_N(|q|)$, at least one of the following holds.

- *$\exists S \subseteq T$ with $\|S\| \leq p_M(|q|) + p_N(|q|)$ such that $L(M^{Y \cup S}) \cap L(N^{Y \cup S}) \neq \emptyset$*
- *$\exists S' \subseteq T,\ \|S'\| \leq p_M(|q|) * p_N(|q|)$, such that either for any $S \subseteq T$, $S \cap S' = \emptyset$, $M^{Y \cup S}(q)$ rejects or for any $S \subseteq T$, $S \cap S' = \emptyset$, $N^{Y \cup S}(q)$ rejects.*

This lemma is essential to our proof. Intuitively this lemma says that we can enforce at least one of N_i and N_j to always reject some query q by reserving only polynomially many strings. Since we have only polynomially many queries, we then will just need to reserve polynomially many strings in Σ_{ij}^{n+1} in total to preserve either N_i's rejection or N_j's rejection on 0^n, and thus have room for diagonalization.

Now we will construct a set Q, the set of strings to be added to the oracle of M_k to make the oracle a separator of $(L(N_i), L(N_j))$, and reserve strings for $\overline{X_{m+1}}$ at the same time to preserve either N_i's rejection or N_j's rejection on 0^n.

Initially we set $Q = \emptyset$. We run M_k on 0^n using oracle $L(N_i^{X_m}) \cup Q$, which is a separator of $(L(N_i^{X_m}, L(N_j^{X_m}))$, until it makes some query q. Then apply Lemma 3.8 with $M = N_i$, $N = N_j$, $Y = X_m$, $T = \Sigma_{ij}^{n+1}$ Considering condition 3, we know that there is a set $S' \subseteq \Sigma_{ij}^{n+1}$ such that either

$$(A)\quad \forall S(S \subseteq \Sigma_{ij}^{n+1} \wedge S \cap S' = \emptyset),\ q \notin L(N_i)^{X_m \cup S}$$

or

$$(B)\quad \forall S(S \subseteq \Sigma_{ij}^{n+1} \wedge S \cap S' = \emptyset),\ q \notin L(N_j)^{X_m \cup S}.$$

We then reserve all strings in S' ($\|S'\| \leq p_i(r_k(n))p_j(r_k(n))$) for $\overline{X_{m+1}}$. If (A) is true, we continue running M_k with the oracle unchanged. (Hence answer "no" to query q_u.) Otherwise we continue running M_k on 0^n with $Q = Q \cup \{q_u\}$. (Hence answer "yes" to query q_u and add q_u to the oracle.) We continue running M_k until it makes the next query and then we do the same thing as above again. We keep doing this until the end of the computation of M_k on 0^n. The number of strings in Σ_{ij}^{n+1} we reserved for $\overline{X_{m+1}}$ during the above process is at most $r_k(n)p_i(r_k(n))p_j(r_k(n)) < 2^n$ since the running time of M_k is bounded by $r_k(n)$. So there exist both a string 0^i10^j10x and a string 0^i10^j11y in Σ_{ij}^{n+1} that are not reserved for $\overline{X_{m+1}}$. If M_k using oracle $L(N_i^{X_m}) \cup Q$ accepts 0^n, we define $X_{m+1} = X_m \cup \{0^i10^j11y\}$. Otherwise define $X_{m+1} = X_m \cup \{0^i10^j10x\}$.

By Lemma 3.11, the oracle $X = \cup X_m$ constructed above fulfills that there is no Turing-complete pair in $\mathcal{D}^X$.

Proof (of Lemma 3.8). Let us define the following languages:

- $L_M = \{\langle P, Q_y, Q_n\rangle$: For some set $S \subseteq T$, P is an accepting path of $M^{Y \cup S}$ on input q and Q_y (resp., Q_n) is the set of the positive queries (resp., negative)made on P for strings in T.
- $L_N = \{\langle P, Q_y, Q_n\rangle$: For some set $S \subseteq T$, P is an accepting path of $N^{Y \cup S}$ on input q and Q_y (resp., Q_n) is the set of the positive (resp., negative) queries made on P for strings in T.

We say that $(P, Q_y, Q_n) \in L_M$ *conflicts* with $(P', Q'_y, Q'_n) \in L_N$ if $Q_y \cap Q'_n \neq \emptyset$ or $Q'_y \cap Q_n \neq \emptyset$. In other words, there is a conflict if at least one query is answered differently on P and P'.

Now we consider the following cases.

Case I $\exists\ (P, Q_y, Q_n) \in L_M$ and $(P', Q'_y, Q'_n) \in L_N$ that do not conflict.

Let $S = Q_y \cup Q'_y$. We claim that in this case, $L(M^{Y \cup S}) \cap L(N^{Y \cup S}) \neq \emptyset$. Note that since there is no conflict, on input q, any positive query asked by M on P to oracle S will still be answered "yes", and any negative query on path P will still be answered "no". In other words, M will still accept q on the path P with oracle S. Similarly, N will accept q on path P' with oracle S. Therefore, $q \in L(M^{Y \cup S}) \cap L(N^{Y \cup S})$. And $\|S\| = \|Q_y\| \cup \|Q_n\| \leq p_M(|q|) + p_N(|q|)$.

Case II Every triple $(P, Q_y, Q_n) \in L_M$ conflicts with every triple $(P', Q'_y, Q'_n) \in L_N$. Note that in this case we cannot have both a triple $\langle P, \emptyset, Q_n \rangle$ in L_M and a triple $\langle P', \emptyset, Q'_n \rangle$ in L_N simply because these two triples do not conflict with each other.

We use the following algorithm to create the set S as claimed in the statement of this lemma.

```
      S' = ∅
      while (L_M ≠ ∅ and L_N ≠ ∅)
(1)       Choose some (P*, Q*_y, Q*_n) ∈ L_M
(2)       S' = S' ∪ Q*_y ∪ Q*_n
(3)       For every t = (P, Q_y, Q_n) ∈ L_M
(4)          remove t if Q_y ∩ (Q*_y ∪ Q*_n) ≠ ∅
(5)       For every t' = (P', Q'_y, Q'_n) ∈ L_N
(6)          remove t' if Q'_y ∩ (Q*_y ∪ Q*_n) ≠ ∅
      end while
```

We claim that after k iterations of the *while* loop, for every triple $(P', Q'_y, Q'_n) \in L_N, \|Q'_n\| \geq k$. If this claim is true, the while loop iterates at most $p_N(|q|)$ times, since for any triple in L_N, $\|Q'_n\|$ is bounded by the running time of N on q, i.e., $p_N(|q|)$. On the other hand, during each iteration, S' is increased by at most $p_M(|q|)$ strings, since for any triple in L_M, $\|Q_y \cup Q_n\|$ is bounded by the running time of M on q, i.e., $p_M(|q|)$. Therefore, $\|S'\| \leq p_M(|q|) * p_N(|q|)$ when this algorithm terminates.

Claim 3.9 *After k-th iteration of the* while *loop of the above algorithm, for every $t' = (P', Q'_y, Q'_n)$ that remains in $L_N, \|Q'_n\| \geq k$.*

Proof For every k, let us denote the triple $(P^k, Q^k_y, Q^k_n) \in L_M$ that is chosen during the k-th iteration by t_k. For every $t' = (P', Q'_y, Q'_n)$ that remains in L_N during this iteration, t_k conflicts with t' (otherwise, we will be in Case I). Therefore, there is a query in $(Q^k_n \cap Q'_y) \cup (Q^k_y \cap Q'_n)$. If this query is in $Q^k_n \cap Q'_y$, t' will be removed from L_N after iteration k. Otherwise, i.e., if $Q^k_y \cap Q'_n \neq \emptyset$, let q' be the first query made by L_N that is in $Q^k_y \cap Q'_n$. In this case, t' will not be removed from L_N; we say that t' *survives* k-th iteration *due to query* q'. Note that t' can survive only due to a query that is negative in P'. We will use this fact to prove that $\|Q'_n\| \geq k$ after k iterations.

We show now that any triple that is left in L_N after k iterations survives each iteration due to a different query. Since these queries are all negative, this will complete the proof of the claim. Assume that t' survives iteration k by query $q' \in Q^k_y \cap Q'_n$. If t' had survived an earlier iteration $l < k$ by the same query q', then q' is also in $Q^l_y \cap Q'_n$. Therefore, $Q^l_y \cap Q^k_y \neq \emptyset$. So $t_k = (P^k, Q^k_y, Q^k_n)$ should have been removed (by lines (3) and (4)) after iteration l, and cannot be chosen at the beginning of iteration k, as claimed. Hence, q' cannot be the query by which t' had survived iteration l. □

Therefore, now we have a set S' of the required size such that either L_M or L_N is empty. Assume that L_M is empty, and for some set $S \subseteq \Sigma_{rs}^d$, $S \cap S' = \emptyset$, $M^{(Y \cup S)}$ accepts q. Therefore, the triple (P, Q_y, Q_n), where P is the accepting path of $M^{(Y \cup S)}(q)$ and Q_y (resp., Q_n) is the set of the positive (resp., negative) queries of length $\geq l_{rs}^d$, must have been in L_M and has been removed during some iteration. That implies that during that iteration, $Q_y \cap S' \neq \emptyset$ (by line (4)), and since $Q_y \subseteq S$, this contradicts the assumption that $S \cap S' = \emptyset$.

A similar argument holds for L_N. Hence either $L_M = \emptyset$ and $M^{(Y \cup S)}$ rejects q for any $S \cap S' = \emptyset$ or $L_N = \emptyset$ and $N^{(Y \cup S)}$ rejects q for any $S \cap S' = \emptyset$. This ends the proof of Lemma 3.8.

□

Lemma 3.10 *After every Stage $m = \langle i, j, k \rangle$, $L(N^{X_{m+1}}) \cap L(N_j^{X_{m+1}}) \neq \emptyset$ or $L(M_k^{L(N_i^{X_{m+1}}) \cup Q})$ does not separate (A_{ij}, B_{ij}), where X_{m+1} is as defined in the proof of Theorem 3.7.*

Proof If at Stage m condition 3 is negated by some set $S \subseteq \Sigma_{ij}^{n_m}$, then we defined $X_{m+1} = X_m \cup S$ hence $L(N^{X_{m+1}}) \cap L(N_j^{X_{m+1}}) \neq \emptyset$. Otherwise, we will start to construct the set Q. From the construction process of Q we know that every string we add to Q is enforced to be rejected by $N_j^{X_m}$ by reserving strings for $\overline{X_{m+1}}$. So $L(N_i^{X_{m+1}}) \cup Q$ will still be a separator of $(L(N_i^{X_{m+1}}), L(N_j^{X_{m+1}}))$. All queries on the computation path of M_k on 0^n using oracle $L(N_i^{X_{m+1}}) \cup Q$ will have the same answers as using oracle $L(N_i^{X_m}) \cup Q$. The reason is as follows. For any query q, if we reserve strings in Σ_{ij}^{n+1} for $\overline{X_{m+1}}$ such that $L(N_i^{X_{m+1}})$ always rejects q in the above process, q will not be put into Q hence query q will get answer "no" from oracle $L(N_i^{X_{m+1}}) \cup Q$, which is the same as the answer from oracle $L(N_i^{X_m}) \cup Q$. If we reserve strings in Σ_{ij}^{n+1} for $\overline{X_{m+1}}$ such that $L(N_j^{X_{m+1}})$ always rejects q, q will be put into Q and hence get the same answer "yes" using oracle $L(N_i^{X_{m+1}}) \cup Q$ as using oracle $L(N_i^{X_m}) \cup Q$. Therefore the computation of M_k on input 0^n using oracle $L(N_i^{X_{m+1}}) \cup Q$ will always have the same result as using oracle $L(N_i^{X_m}) \cup Q$. So by the way we define X_{m+1}, M_k using oracle $L(N_i^{X_{m+1}}) \cup Q$ does not separate $(L(N_i^{X_{m+1}}), L(N_j^{m+1}))$, regardless of whether M_k accepts 0^n. □

Lemma 3.11 *The oracle $X = \cup X_m$ constructed in the proof of Theorem 3.7 has the desired property.*

Proof Let (C, D) be a pair in $\mathcal{D}^X$. Suppose $C = L(N_i^X)$ and $D = L(N_j^X)$ for some i and j. Then by Lemma 3.10 we know that one of the following happens during the construction of X:

- At some Stage $l = \langle i, j, k \rangle$, there exists a string $s \in L(N_i^{X_{l+1}}) \cap L(N_j^{X_{l+1}})$ and $n_l = |s|$. Since we choose the number n_m at each Stage m such that $l_{ij}^{n+1} > \max(p_{i'}(n_{m-1}), p_{j'}(n_{m-1}))$, the strings added into the oracle at a later Stage $m > l$ will not disturb the acceptance of s by N_i and N_j. So for any $m > l$ we still have $s \in L(N_i^{X_m}) \cap L(N_j^{X_m})$. Thus $C = L(N_i^X) \cap D = L(N_j^X) \neq \emptyset$. (C, D) is not in $\mathcal{D}^X$.

- For any k, $L(M_k^{L(N_i^{X_{l+1}}) \cup Q})$ does not separate (A_{ij}, B_{ij}), where $l = \langle i, j, k \rangle$. Actually, we can see from the proof of Lemma 3.10 that either

$$0^{n_l} \in A_{ij} \; and \; 0^{n_l} \notin L(M_k^{L(N_i^{X_{l+1}}) \cup Q})$$

or

$$0^{n_l} \in B_{ij} \; and \; 0^{n_l} \in L(M_k^{L(N_i^{X_{l+1}})\cup Q})$$

after Stage l. Recall that we choose the number n_m at each Stage m such that $n_m > n_{m-1}$ and $l_{ij}^{n_m+1} > \max(p_{i'}(r_{k'}(n_{m-1})), p_{j'}(r_{k'}(n_{m-1})))$ following holds. Therefore, we know that the strings added in a later stage $m > l$ will not change the following:

1. The membership of 0^{n_l} in A_{ij} and B_{ij}. Only the strings of length $n_l + 1$ are only added into the oracle at Stage l.
2. The computations set up in the last part of the proof of Theorem 3.7. The maximal length of strings that can be queried in those computations is $\max(p_{i'}(r_{k'}(n_{m-1})), p_{j'}(r_{k'}(n_{m-1}))) < l_{ij}^{n_m+1}$.

For any later Stage $m > l$, we also have either

$$0^{n_l} \in A_{ij} \; and \; 0^{n_l} \notin L(M_k^{L(N_i^{X_{m+1}})\cup Q})$$

or

$$0^{n_l} \in B_{ij} \; and \; 0^{n_l} \in L(M_k^{L(N_i^{X_{m+1}})\cup Q}).$$

So (A_{ij}, B_{ij}), which is in $\mathcal{D}^X$, is not uniformly Turing-reducible to the pair (C, D). Hence in this case (C, D) could not be a complete pair for $\mathcal{D}^X$.

□

This completes the proof of the theorem. □

Theorem 5.6 *Each of the hypotheses stated in Theorem 5.5 implies the existence of nonsymmetric* NP*-pairs.*

Proof Let us define the following function.

$$dt(i) = \begin{cases} 1 & \text{if } i = 0 \\ 2^{2^{dt(i-1)}} & \text{otherwise} \end{cases}$$

Let M be the UP-machine accepting 0^* as in the first hypothesis in Theorem 5.5. Let a_n be the accepting computation of M on 0^n. We can assume that $|a_n| = m$ where m is some fixed polynomial in n. We define the following sets.

$$\begin{aligned} L_M &= \{\langle 0^n, w\rangle : w \le a_n, n = dt(i) \text{ for some } i > 0\} \\ R_M &= \{\langle 0^n, w\rangle : w > a_n, n = dt(i) \text{ for some } i > 0\} \end{aligned}$$

Note that (L_M, R_M) is a disjoint NP-pair. We claim that L_M is p-selective. The description of a selector f for L_M follows: Assume that $\langle 0^k, w_1\rangle$ and $\langle 0^l, w_2\rangle$ are input to f. If $k = l$, then f outputs the lexicographically smaller one of w_1 and w_2. Otherwise, assume that $k < l$. In that case, $l \ge 2^{2^k} > 2^{|a_k|}$. Recall that the accepting computation of M on 0^k is a_k. The function f can find out the actual accepting computation of M on 0^k by checking all possible strings of length $|a_k|$. Therefore, in $O(l)$ time, f outputs $\langle 0^k, w_1\rangle$ if $w_1 \le a_k$, and outputs $\langle 0^l, w_2\rangle$ otherwise. Similarly, we can show that R_M is p-selective.

We claim that (L_M, R_M) is a nonsymmetric NP-pair. Assume on the contrary that this pair is symmetric. Therefore, by Proposition 5.3 (L_M, R_M) is P-separable, i.e., there is $S \in \mathrm{P}$ that is a separator for (L_M, R_M). Using a standard binary search technique, a polynomial-time machine can compute the accepting computation of M on any 0^n where $n = dt(i)$ for some $i > 0$. Since the length of the accepting computation of M on 0^n is m, this binary search algorithm can takes time $O(m)$, i.e., time polynomial in n. This contradicts Hypothesis H, since we assumed that no polynomial-time machine can compute infinitely many accepting computations of M. Therefore, (L_M, R_M) is a nonsymmetric NP-pair. □

Theorem 5.7 *There exists a nonsymmetric disjoint* NP*-pair* (A, B) *if and only if there exists a nonsymmetric disjoint* NP*-pair* (C, D)*, where both* C *and* D *are* $\leq_m^p$*-complete for* NP.

Proof The *if* direction is trivial. We prove the *only if* direction.

Let $\{NM_i\}_{i\geq 1}$ be an enumeration of polynomial-time bounded nondeterministic Turing machines with associated polynomial time bounds $\{p_i\}_{i\geq 1}$. It is well known that the following set K is NP-complete [BGS75].

$$K = \{\langle i, x, 0^n\rangle |\ \text{ some computation of } NM_i \text{ accepts } x \text{ in at most } n \text{ steps }\}.$$

For every set $A \in \mathrm{NP}$ there exists $i \geq 1$ such that $A = L(NM_i)$ and there exists an honest many-one reduction f from A to K defined by $f(x) = \langle i, x, 0^{p_i(|x|)}\rangle$. Let (A, B) be a nonsymmetric disjoint NP-pair and let f be an honest reduction from A to K.

Our first goal is to show that $(f(B), K)$ is nonsymmetric. Since f is a reduction from A to K and $A \cap B = \emptyset$, $f(A) \subseteq K$ and $f(B) \subseteq \overline{K}$, and so $f(B)$ and K are disjoint sets. Observe that $f(B)$ is in NP because on any input y, we can guess a $x \in B$ and verify that $f(x) = y$. Therefore, $(K, f(B))$ is an NP-pair with one of them being $\leq_m^p$-complete for NP.

In order to prove that this pairs is nonsymmetric, assume otherwise: then $(K, f(B)) \leq_m^{pp} (f(B), K)$ and therefore, $\exists g \in \mathrm{PF}$ such that $g(K) \subseteq f(B)$ and $g(f(B)) \subseteq K$. Consider the following polynomial-time computable function h. On input x, h computes $y = g(f(x))$. If $y = \langle i, x', 0^{p_i(|x'|)}\rangle$ for some x', h outputs x'; otherwise, it returns a fixed string $a \in A$. We claim that $h(A) \subseteq B$ and $h(B) \subseteq A$, thereby making (A, B) symmetric. For any $x \in A$, we know that $f(x) \in K$ and hence $g(f(x)) \in f(B)$ since $g(K) \subseteq f(B)$. So $h(x) = \langle i, x', 0^{p_i(|x'|)}\rangle$ for some $x' \in B$, and so $h(x) = x' \in B$. For any $x \in B$, $y = g(f(x)) \in K$, since $g(f(B)) \subseteq K$. If $y = \langle i, x', 0^{p_i(|x|)}\rangle$ for some x', then x' must be in A; else h will return $a \in A$, and so, in either case, $x \in B$ will imply that $h(x) \in A$. Therefore, $h(A) \subseteq B$ and $h(B) \subseteq A$. Thus $(A, B) \leq_m^{pp} (B, A)$, contradicting the fact that (A, B) is nonsymmetric. Hence $(K, f(B))$ is a nonsymmetric NP-pair.

To complete the proof of the theorem, apply the construction once again, this time with an honest reduction f' from $f(B)$ to K. Namely, $f'(f(B)) \subseteq K$ and $f'(K) \subseteq \overline{K}$. Then, K and $f'(K)$ are disjoint NP-complete sets and the argument already given shows that $(f'(K), K)$ is nonsymmetric. □

Theorem 5.8 *There exists an disjoint* NP*-pair* (A, B) *that is* $\leq_m^{pp}$*- complete if and only if there exists a disjoint* NP*-pair* (C, D) *that is* $\leq_m^{pp}$*-complete where both* C *and* D *are* $\leq_m$*-complete for* NP.

Proof The proof idea is similar to the proof of Theorem 5.7. Consider the one-to-one function f: $f(x) = \langle i, x, 0^{p_i(|x|)}\rangle$ that many-one reduces A to the canonical NP-complete set K.

Obviously $(A, B) \leq_m^{pp} (K, f(B))$ since $f(A) \subseteq K$ and $B \subseteq f_i(B)$ and $K \cap f_i(B) = \emptyset$ as shown in the proof of Theorem 5.7. Similar to that theorem, we apply the one-to-one function f' that many-one

reduces $f(B)$ to K to obtain another disjoint NP-pair $(f'(K), K)$ where $(K, f(B)) \leq_m^{pp} (f'(K), K)$. So $(A, B) \leq_m^{pp} (K, f(B)) \leq_m^{pp} (f'(K), K)$. Therefore $(f'(K), K)$ is also a $\leq_m^{pp}$-complete NP-pair with $f'(K)$ and K both being $\leq_m^p$-complete sets for NP. □

Theorem 6.1 *There exists an oracle O relative to which the following holds:*

(i) There exist $\leq_m^{pp}$-complete NP*-pairs.*

(ii) There exist nonsymmetric NP*-pairs.*

(iii) Conjecture 2.4 holds.

Proof We fix the following enumerations: $\{NM_i\}_i$ is an effective enumeration of nondeterministic, polynomial time-bounded oracle Turing machines; $\{M_i\}_i$ is an effective enumeration of deterministic, polynomial time-bounded oracle Turing machines; $\{T_i\}_i$ is an effective enumeration of deterministic, polynomial time-bounded oracle Turing transducers. Moreover, NM_i, M_i and T_i have running time $p_i = n^i$ independent of the choice of the oracle. Let f_i^Z denote the function that T_i^Z computes.

We use the following model of nondeterministic polynomial-time oracle Turing machines. On some input the machine starts the first phase of its computation, during which it is allowed to make nondeterministic branches. In this phase the machine is not allowed to ask any queries. At the end of the first phase the machine has computed a list of queries $q_1, \ldots, q_n$, a list of guessed answers $g_1, \ldots, g_n$, and a character, which is either $+$ or $-$. Now the machine asks in parallel all queries and gets the vector of answers $a_1, \ldots, a_n$. The machine accepts if the computed character is $+$ and $(a_1, \ldots, a_n) = (g_1, \ldots, g_n)$; otherwise the machine rejects. An easy observation shows that for all oracles X, a language L is in NP^X if and only if there exists a nondeterministic polynomial-time oracle Turing machine N such that N works in the described way and $L = L(N^X)$.[3]

A nondeterministic polynomial-time oracle Turing machine N and an input x determine a computation path P. P contains all nondeterministic guesses, all queries and all guessed answers. A computation path P that has the character $+$ (resp., $-$) is called a positive (resp., negative) path. The set of queries that are guessed to be answered positively (resp., negatively) is denoted by P^{yes} (resp., P^{no}); the set of all queries is denoted by $P^{\mathrm{all}} \stackrel{df}{=} P^{\mathrm{yes}} \cup P^{\mathrm{no}}$. The length of P (i.e., the number of computation steps) is denoted by $|P|$. Note that this description of paths makes it possible to talk about paths of computations without specifying the oracle, i.e., we can say that a computation $N(x)$ has a positive path P such that P^{yes} and P^{no} satisfy certain conditions. However, when talking about accepting and rejecting paths we always have to specify the oracle. (A positive path can be accepting for certain oracles, and it can be rejecting for other oracles.)

In this proof we need to consider injective, partial functions $\mathbb{N}^+ \to \mathbb{N}^+ \times \mathbb{N}^+$ that have a finite domain. Usually, such functions are denoted by μ. We do not distinguish between the function and the set of all (n, i, j) with $\mu(n) = (i, j)$. Both are denoted by μ. For $X, Y \subseteq \Sigma^*$ we write $Y \supseteq_m X$ if and only if $X \subseteq \Sigma^{\leq m}$ and $Y^{\leq m} = X$. We write $Y \subseteq_m X$ if and only if $X \supseteq_m Y$.

Definition 6.2 *Let μ and μ' be injective, partial functions $\mathbb{N}^+ \to \mathbb{N}^+ \times \mathbb{N}^+$ that have a finite domain. If $\mu \neq \emptyset$, then $\mu_{\max} \stackrel{df}{=} \max(\mathrm{dom}(\mu))$. We write $\mu \preceq \mu'$ if either $\mu = \emptyset$, or $\mu \subseteq \mu'$ and $\mu_{\max} < n$ for all $n \in \mathrm{dom}(\mu' - \mu)$. We write $\mu \prec \mu'$ if $\mu \preceq \mu'$ and $\mu \neq \mu'$.*

[3] Note that for this equivalence we need both, the character to be $+$ and the g_i to be guessed correctly. If the machine accepted just when the answers were guessed correctly, then we would miss the machine that accepts $\emptyset$ for every oracle.

In our construction we use the following witness languages, which depend on an oracle Z.

$$\begin{aligned}
A(Z) &\stackrel{df}{=} \{w \,|\, w = 0^n 10^t 1x \text{ for } n,t \geq 1, x \in \Sigma^* \text{ and } (\exists y \in \Sigma^{|w|+1})[0wy \in Z]\} \\
B(Z) &\stackrel{df}{=} \{w \,|\, w = 0^n 10^t 1x \text{ for } n,t \geq 1, x \in \Sigma^* \text{ and } (\exists y \in \Sigma^{|w|+1})[1wy \in Z]\} \\
C(Z) &\stackrel{df}{=} \{0^k \,|\, k \equiv 1 (\mathrm{mod}\ 4), (\exists y \in \Sigma^{k-1})[0y \in Z]\} \\
D(Z) &\stackrel{df}{=} \{0^k \,|\, k \equiv 1 (\mathrm{mod}\ 4), (\exists y \in \Sigma^{k-1})[1y \in Z]\} \\
E(Z) &\stackrel{df}{=} \{0^k \,|\, k \equiv 3 (\mathrm{mod}\ 4), (\exists y \in \Sigma^{k})[y \in Z]\}
\end{aligned}$$

These languages are in NP^Z. We construct the oracle O such that $A(O) \cap B(O) = C(O) \cap D(O) = \emptyset$ and the following holds.

(i) $(A(O), B(O))$ is $\leq_m^{pp}$-complete. That is,

$$(\forall (F,G) \in \mathcal{D}^O)(\exists f \in \mathrm{FP})[f(F) \subseteq A(O) \wedge f(G) \subseteq B(O) \wedge f(\overline{F \cup G}) \subseteq \overline{A(O) \cup B(O)}]. \quad (4)$$

(ii) $(C(O), D(O))$ is nonsymmetric. That is,

$$(\forall f \in \mathrm{FP}^O)[f(C(O)) \not\subseteq D(O) \vee f(D(O)) \not\subseteq C(O)]. \quad (5)$$

(iii) $E(O) \not\leq_T^{pp,O} (A(O), B(O))$. That is,

$$(\exists S, A(O) \subseteq S \subseteq \overline{B(O)})[E(O) \notin \mathrm{P}^S]. \quad (6)$$

In (4) and (6) we really mean $f \in \mathrm{FP}$ and $E(O) \notin \mathrm{P}^S$; we explain why this is equivalent to $f \in \mathrm{FP}^O$ and $E(O) \notin \mathrm{P}^{S,O}$. We have to see that the expressions (4), (5), and (6) imply the statements (i), (ii), and (iii) of the theorem. For (4) and (5) this follows from Theorem 2.10 and the fact that $f \in \mathrm{FP}$ implies $f \in \mathrm{FP}^O$. Note that in (4) we actually do not need the inclusion $f(\overline{F \cup G}) \subseteq \overline{A(O) \cup B(O)}$. We state it here because the proof yields this condition, which in turn shows that the oracle even applies to a notion stronger than $\leq_m^{pp}$. In (iii) we actually should have $E(O) \notin \mathrm{P}^{S,O}$ since the reducing machine has access to the oracle O. However, since (i) holds and since $(O, \overline{O}) \in \mathcal{D}^O$, there exists an $f \in \mathrm{FP}$ with $f(O) \subseteq A(O) \subseteq S$ and $f(\overline{O}) \subseteq B(O) \subseteq \overline{S}$. Hence, $q \in O \Leftrightarrow f(q) \in S$. So we can transform queries to O into queries to S, i.e., it suffices to show $E(O) \notin \mathrm{P}^S$.

We define the following list $\mathcal{T}$ of requirements. At the beginning of the construction, $\mathcal{T}$ contains all pairs (i,n) with $i \in \{0,1,2\}$ and $n \in \mathbb{N}^+$. These pairs have the following interpretations.

- $(0, \langle i,j \rangle)$ means: ensure $L(NM_i^O) \cap L(NM_j^O) \neq \emptyset$ or $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$
- $(1,i)$ means: ensure $[0^n \in C(O) \wedge T_i^O(0^n) \notin D(O)]$ or $[0^n \in D(O) \wedge T_i^O(0^n) \notin C(O)]$
- $(2,i)$ means: ensure that $(A(O), B(O))$ has a separator S with $0^n \in E(O) \Leftrightarrow 0^n \notin L(M_i^S)$

Once a requirement is satisfied we delete it from the list. The latter two types of conditions are reachable by the construction of one counter example. In contrast, if we cannot reach $L(NM_i^O) \cap L(NM_j^O) \neq \emptyset$ for a condition of the first type, then we have to ensure $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$. But this condition cannot be reached by a finite segment of an oracle; instead it influences the whole remaining construction of the oracle. We have to encode answers to queries "does x belong to $L(NM_i^O)$ or to $L(NM_j^O)$" into the oracle O. For this reason we introduce the notion of (μ,k)-valid oracles. Here k is a natural number and μ is an

injective, partial function $\mathbb{N}^+ \to \mathbb{N}^+ \times \mathbb{N}^+$ that has a finite domain. Each (μ, k)-valid oracle is a subset of $\Sigma^{\leq k}$. Roughly speaking, μ can be thought of as a finite set of pairs (i, j), for which $L(NM_i^O) \cap L(NM_j^O) = \emptyset$ is forced, and therefore, we must construct O so that $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$ holds. For the latter condition we have to encode certain information into O, and the number k says up to which level this encoding have been done. So (μ, k)-valid oracles should be considered as finite prefixes of oracles that contain these encodings. For the moment we postpone the formal definition of (μ, k)-valid oracles (Definition 6.4); instead we mention its essential properties, which will be proved later.

(a) The oracle $\emptyset$ is $(\emptyset, 0)$-valid.

(b) If X is a finite oracle that is (μ, k)-valid, then for all $\mu' \preceq \mu$, X is (μ', k)-valid.

(c) If O is an oracle such that $O^{\leq k}$ is (μ, k)-valid for infinitely many k, then $A(O) \cap B(O) = C(O) \cap D(O) = \emptyset$, and for all $(i, j) \in \mathrm{range}(\mu)$ it holds that $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$ via some $f \in \mathrm{FP}$. Even more it holds that $f(\overline{L(NM_i^O) \cup L(NM_j^O)}) \subseteq \overline{A(O) \cup B(O)}$.

The properties (a), (b), and (c) will be proved later in Proposition 6.5. Moreover, the following holds for all $i, j \geq 1$ and all (μ, k)-valid X.

P1: There exists an $l > k$ and a (μ', l)-valid $Y \supseteq_k X$, $\mu \preceq \mu'$ such that

- either for all $Z \supseteq_l Y$, $L(NM_i^Z) \cap L(NM_j^Z) \neq \emptyset$,
- or $(i, j) \in \mathrm{range}(\mu')$.

P2: There exists an $l > k$ and a (μ, l)-valid $Y \supseteq_k X$ such that for all $Z \supseteq_l Y$, if $C(Z) \cap D(Z) = \emptyset$, then $(C(Z), D(Z))$ does not $\leq_m^{pp,O}$-reduce to $(D(Z), C(Z))$ via T_i^Z.

P3: There exists an $l > k$ and a (μ, l)-valid $Y \supseteq_k X$ such that for all $Z \supseteq_l Y$, if $A(Z) \cap B(Z) = \emptyset$, then there exists a separator S of $(A(Z), B(Z))$ such that $E(Z) \neq L(M_i^S)$.

We will prove the properties P1, P2, and P3 in the Propositions 6.11, 6.12, and 6.14. In the following, we construct an ascending sequence of finite oracles $X_0 \subseteq_{k_0} X_1 \subseteq_{k_1} X_2 \subseteq_{k_2} \cdots$ such that each X_r is (μ_r, k_r)-valid, $k_0 < k_1 < k_2 < \cdots$ and $\mu_0 \preceq \mu_1 \preceq \mu_2 \preceq \cdots$. By definition, $O = \bigcup_{r \geq 0} X_r$. By items (b) and (c), $A(O) \cap B(O) = C(O) \cap D(O) = \emptyset$ follows immediately. We claim that for each $r \geq 0$ and $i \geq 1$, $X_{r+i} \supseteq_{k_r} X_r$ and $\mu_r \preceq \mu_{r+i}$.

1. $r := 0$, $k_r := 0$, $\mu_r := \emptyset$, and $X_r := \emptyset$. Then by (a), X_r is (μ_r, k_r)-valid.

2. Remove the next requirement e from $\mathcal{T}$ and do the following:

   - If $e = (0, \langle i, j \rangle)$, then we apply property P1 to X_r. Define $k_{r+1} = l$, $\mu_{r+1} = \mu'$ and $X_{r+1} = Y$. Then $k_r < k_{r+1}$, $\mu_r \preceq \mu_{r+1}$ and $X_{r+1} \supseteq_{k_r} X_r$ is (μ_{r+1}, k_{r+1})-valid such that
     - either for all $Z \supseteq_{k_{r+1}} X_{r+1}$, $L(NM_i^Z) \cap L(NM_j^Z) \neq \emptyset$,
     - or $(i, j) \in \mathrm{range}(\mu_{r+1})$.

     Comment: If the former holds, then, since $O \supseteq_{k_{r+1}} X_{r+1}$, it holds that $L(NM_i^O) \cap L(NM_j^O) \neq \emptyset$, and therefore, $(L(NM_i^O), L(NM_j^O)) \notin \mathcal{D}^O$. Otherwise, $(i, j) \in \mathrm{range}(\mu_{r+1})$. By (b), for all $i \geq 1$, X_{r+i} is (μ_{r+1}, k_{r+i})-valid. Therefore, by (c), $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$ via some $f \in \mathrm{FP}$.

- If $e = (1, i)$, then $\mu_{r+1} \stackrel{df}{=} \mu_r$ and apply property P2 to X_r. We define $k_{r+1} = l$ and $X_{r+1} = Y$. Then $k_{r+1} > k_r$ and $X_{r+1} \supseteq_{k_r} X_r$ is (μ_{r+1}, k_{r+1})-valid so that for all $Z \supseteq_{k_{r+1}} X_{r+1}$, with $C(Z) \cap D(Z) = \emptyset$, $(C(Z), D(Z))$ does not $\leq_m^{pp,O}$-reduce to $(D(Z), C(Z))$ via T_i^Z.

  Comment: Since $O \supseteq_{k_{r+1}} X_{r+1}$ and $C(O) \cap D(O) = \emptyset$ this ensures that $(C(O), D(O))$ does not $\leq_m^{pp,O}$-reduce to $(D(O), C(O))$ via T_i^O.

- If $e = (2, i)$, then $\mu_{r+1} \stackrel{df}{=} \mu_r$ and apply property P3 to X_r. We define $k_{r+1} = l$ and $X_{r+1} = Y$. Then $k_{r+1} > k_r$ and $X_{r+1} \supseteq_{k_r} X_r$ is (μ_{r+1}, k_{r+1})-valid such that for all $Z \supseteq_{k_{r+1}} X_{r+1}$, $A(Z) \cap B(Z) = \emptyset$, there exists a separator S of $(A(Z), B(Z))$ such that $E(Z) \neq L(M_i^S)$.

  Comment: Since $O \supseteq_{k_{r+1}} X_{r+1}$ and $A(O) \cap B(O) = \emptyset$ this ensures that there exists a separator S of $(A(O), B(O))$ such that $E(O) \neq L(M_i^S)$.

3. $r := r + 1$, go to step 2.

We see that this construction ensures (i), (ii), and (iii). This proves the theorem except to show that we can define an appropriate notation of a (μ, k)-valid oracle that has the properties (a), (b), (c), and P1, P2, P3.

We want to construct our oracle such that $(A(O), B(O))$ is a $\leq_m^{pp}$-complete NP^O-pair. So we have to make sure that pairs $(L(M_i), L(M_j))$ that are enforced to be disjoint (which means that $(i, j) \in \mathrm{range}(\mu)$) can be many-one reduced to $(A(O), B(O))$. Therefore, we put certain code-words into O if and only if the computation $M_i^O(x)$ (resp., $M_j^O(x)$) accepts within t steps.

Definition 6.3 (μ-code-word) *Let $\mu : \mathbb{N}^+ \to \mathbb{N}^+ \times \mathbb{N}^+$ be an injective, partial function with a finite domain. A word w is called* μ-code-word *if $w = 00^n10^t1xy$ or $w = 10^n10^t1xy$ such that $|y| = |00^n10^t1x|$ and $\mu(n) = (i, j)$. If $w = 00^n10^t1xy$, then we say that w is a μ-code-word for (i, t, x); if $w = 10^n10^t1xy$, then we say it is a μ-code-word for (j, t, x).*

Condition (i) of Theorem 6.1 opposes the conditions (ii) and (iii), because for (i) we have to encode information about NP^O computations into O, and (ii) and (iii) say that we cannot encode too much information (e.g., enough information for $\mathrm{UP}^O = \mathrm{NP}^O$). For this reason we have to look at certain finite oracles that contain the needed information for (i) and that allow all diagonalization needed to reach (ii) and (iii). We call such oracles (μ, k)-valid.

Definition 6.4 ((μ, k)-valid oracle) *Let $k \geq 0$ and let $\mu : \mathbb{N}^+ \to \mathbb{N}^+ \times \mathbb{N}^+$ be an injective, partial function with a finite domain. We define a finite oracle X to be (μ, k)-valid by induction over the size of the domain of μ.*

- *If $\|\mu\| = 0$, then X is (μ, k)-valid $\stackrel{df}{\iff}$ $X \subseteq \Sigma^{\leq k}$ and $A(X) \cap B(X) = C(X) \cap D(X) = \emptyset$.*
- *If $\|\mu\| > 0$, then $\mu = \mu' \cup \{(\mu_{\max}, i_0, j_0)\}$ for a suitable $\mu' \prec \mu$. X is (μ, k)-valid $\stackrel{df}{\iff}$*
  1. *$k \geq \mu_{\max}$ and X is (μ', k)-valid.*
  2. *For all $(n, i, j) \in \mu$, $t \geq 1$ and $x \in \Sigma^*$ with $2 \cdot |00^n10^t1x| \leq k$,*
     (a) $(\exists y, |y| = |00^n10^t1x|)[00^n10^t1xy \in X] \Leftrightarrow NM_i^X(x)$ accepts within t steps, and
     (b) $(\exists y, |y| = |10^n10^t1x|)[10^n10^t1xy \in X] \Leftrightarrow NM_j^X(x)$ accepts within t steps.
  3. *For all $l \geq \mu_{\max}$ and all (μ', l)-valid Y, $Y^{\leq \mu_{\max}} = X^{\leq \mu_{\max}}$, $L(NM_{i_0}^Y) \cap L(NM_{j_0}^Y) \cap \Sigma^{\leq l} = \emptyset$.*

Due to the last condition, (μ, k)-valid oracles can be extended to (μ, k')-valid oracles with $k' > k$ (Lemma 6.9). There we really need the intersection with $\Sigma^{\leq l}$; otherwise it would be possible that for a small oracle $Y \subseteq \Sigma^{\leq l}$ both machines accept the same word w that is much longer than l, but there is no way to extend Y in a valid way to the level $|w|$ such that both machines still accept w (the reason is that the reservations (Definition 6.6) become to large).

Proposition 6.5 (basic properties of validity)

1. *The oracle $\emptyset$ is $(\emptyset, 0)$-valid.* *(property (a))*
2. *For every (μ, k)-valid X and every $\mu' \preceq \mu$, X is (μ', k)-valid.* *(property (b))*
3. *If X is (μ, k)-valid and k is even, then for every $S \subseteq \Sigma^{k+1}$, if $C(S) \cap D(S) = \emptyset$, then $X \cup S$ is $(\mu, k+1)$-valid.*
4. *For every (μ, k)-valid X and every $(i, j) \in \mathrm{range}(\mu)$, $L(NM_i^X) \cap L(NM_j^X) \cap \Sigma^{\leq k} = \emptyset$.*
5. *If X is (μ, k)-valid, then for every l, $\mu_{\max} \leq l \leq k$, it holds that $X^{\leq l}$ is (μ, l)-valid.*
6. *Let O be an oracle such that for infinitely many k, $O^{\leq k}$ is (μ, k)-valid. Then the following hold:* *(property (c))*
   - $A(O) \cap B(O) = C(O) \cap D(O) = \emptyset$.
   - *For all $(i, j) \in \mathrm{range}(\mu)$ it holds that $L(NM_i^O) \cap L(NM_j^O) = \emptyset$ and there exists some $f \in \mathrm{FP}$ such that $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$ via f. Even more, it holds that $f(\overline{L(NM_i^O) \cup L(NM_j^O)}) \subseteq \overline{A(O) \cup B(O)}$.*

Proof The statements 6.5.1 and 6.5.2 follow immediately from Definition 6.4.

We prove statement 6.5.3 by induction on $\|\mu\|$. First of all we note that $A(S) = B(S) = \emptyset$ since S contains only words of odd length. If $\|\mu\| = 0$, then, by Definition 6.4, $X \cup S$ is $(\mu, k+1)$-valid. So assume $\|\mu\| > 0$ and choose μ', i_0, j_0 as in Definition 6.4. We assume as induction hypothesis that if X is (μ', k)-valid, then $X \cup S$ is $(\mu', k+1)$-valid. We have to verify the statements 6.4.1–6.4.3 for $X \cup S$ and $k+1$. Clearly, $k+1 > k \geq \mu_{\max}$. Since X is (μ, k)-valid it is also (μ', k)-valid. By induction hypothesis we obtain that $X \cup S$ is $(\mu', k+1)$-valid, i.e., 6.4.1 holds. Since k is even, the condition $2 \cdot |00^n 10^t 1x| \leq k+1$ is equivalent to $2 \cdot |00^n 10^t 1x| \leq k$. Moreover, since $t < k$ the computations mentioned in 6.4.2 cannot ask queries longer than k. This means that in 6.4.2 we can change the oracle from $X \cup S$ to X. The resulting condition holds since X is (μ, k)-valid. Therefore, 6.4.2 holds for $X \cup S$ and $k+1$. Finally, 6.4.3 holds for $X \cup S$ and $k+1$, since this condition does not depend on k and $(X \cup S) \cap \Sigma^{\leq k} = X^{\leq k}$. This proves 6.5.3.

Assume that $L(NM_i^X) \cap L(NM_j^X) \cap \Sigma^{\leq k} \neq \emptyset$ for some $(i, j) \in \mathrm{range}(\mu)$. Choose n such that $(n, i, j) \in \mu$. Let $\mu' \stackrel{\mathrm{df}}{=} \{(n', i', j') \in \mu \,\big|\, n' < n\}$ and observe that $\mu' \cup \{(n, i, j)\} \preceq \mu$. By 6.5.2, X is $(\mu' \cup \{(n, i, j)\}, k)$-valid and also (μ', k)-valid. Together with 6.4.3 this implies that $L(NM_i^X) \cap L(NM_j^X) \cap \Sigma^{\leq k} = \emptyset$, which contradicts our assumption. This shows 6.5.4.

Statement 6.5.5 follows from Definition 6.5 by induction on $\|\mu\|$. This induction is similar to that used in the proof of 6.5.3.

Let O be as in statement 6.5.6 and let $(i, j) \in \mathrm{range}(\mu)$. Choose n such that $(n, i, j) \in \mu$. Assume that $A(O) \cap B(O) \neq \emptyset$ and let $w \in A(O) \cap B(O)$. Then, for $k = 2 \cdot (|w| + 1)$, w is already in $A(O^{\leq k}) \cap B(O^{\leq k})$. This contradicts the assumption that there exists a $k' \geq k$ such that $O^{\leq k'}$ is (μ, k')-valid. Therefore, $A(O) \cap B(O) = \emptyset$. Analogously we see that $C(O) \cap D(O) = \emptyset$ and $L(NM_i^O) \cap L(NM_j^O) = \emptyset$ (Here we use Proposition 6.5.4.). By our assumption and Definition 6.4, for infinitely many k the following holds: For all $t \geq 1$ and $x \in \Sigma^*$ with $2 \cdot |00^n 10^t 1x| \leq k$,

- $(\exists y, |y| = |00^n 10^t 1x|)[00^n 10^t 1xy \in O^{\leq k}] \Leftrightarrow NM_i^{O^{\leq k}}(x)$ accepts within t steps, and

- $(\exists y, |y| = |10^n10^t1x|)[10^n10^t1xy \in O^{\leq k}] \Leftrightarrow NM_j^{O^{\leq k}}(x)$ accepts within t steps.

During the first t steps a machine can only ask queries of length $\leq t < k$. Therefore, above we can replace $NM_i^{O^{\leq k}}(x)$ and $NM_j^{O^{\leq k}}(x)$ by $NM_i^O(x)$ and $NM_j^O(x)$, respectively. Since all this holds for infinitely many k, the following holds for all $t \geq 1$ and $x \in \Sigma^*$.

- $(\exists y, |y| = |00^n10^t1x|)[00^n10^t1xy \in O] \Leftrightarrow NM_i^O(x)$ accepts within t steps, and
- $(\exists y, |y| = |10^n10^t1x|)[10^n10^t1xy \in O] \Leftrightarrow NM_j^O(x)$ accepts within t steps.

This shows that $(L(NM_i^O), L(NM_j^O)) \leq_m^{pp} (A(O), B(O))$ via some $f \in \mathrm{FP}$.[4] Even more, if both machines do not accept x within t steps, then 0^n10^t1x is neither in $A(O)$ nor is in $B(O)$. This means $f(\overline{L(NM_i^O) \cup L(NM_j^O)}) \subseteq \overline{A(O) \cup B(O)}$. □

Remember that our construction consists of a coding part to obtain condition (i) of Theorem 6.1 and of separating parts to obtain conditions (ii) and (iii). In order to diagonalize, we will fix certain words that are needed for the coding part and we will change our oracle on nonfixed positions to obtain the separation. For this we introduce the notion of a reservation for an oracle. A reservation consists of two sets Y and N where Y contains words that are reserved for the oracle while N contains words that are reserved for the complement of the oracle. This notion has two important properties:

- Whenever an oracle X agrees with a reservation that is not too large, we can find an extension of X that agrees with the reservation (Lemma 6.8).
- If we want to fix a certain word to be in the oracle, then this is possible by a reservation of small size. For this reason we can fix certain words to be in the oracle and still be able to diagonalize. (Lemma 6.10)

Definition 6.6 ((μ, k)-reservation) *(Y, N) is a (μ, k)-*reservation for *X if X is (μ, k)-valid, $Y \cap N = \emptyset$, $Y^{\leq k} \subseteq X, N \subseteq \overline{X}$, all words in $Y^{>k}$ are μ-code-words, and if $w \in Y^{>k}$ is a μ-code-word for (i, t, x), then $NM_i(x)$ has a positive path P with $|P| \leq t$, $P^{\mathrm{yes}} \subseteq Y$ and $P^{\mathrm{no}} \subseteq N$.*

Proposition 6.7 (basic properties of reservations) *The following holds for every (μ, k)-valid X.*

1. *$(\emptyset, \emptyset)$ is a (μ, k)-reservation for X.*
2. *If (Y, N) is a (μ, k)-reservation for X, then also $(Y, N \cup N')$ for every $N' \subseteq \overline{Y \cup X}$.*
3. *For every $N \subseteq \overline{X}$, $(\emptyset, N)$ is a (μ, k)-reservation for X.*
4. *If (Y, N) is a (μ, k)-reservation for X, then (Y, N) is a $(\mu, k+1)$-reservation for each $(\mu, k+1)$-valid $Z \supseteq_k X$ with $Y^{=k+1} \subseteq Z^{=k+1} \subseteq \overline{N}^{=k+1}$.*

Proof This follows immediately from Definition 6.6. □

Whenever a (μ, k)-reservation of some oracle X is not too large, then X has a (μ, m)-valid extension Z that agrees with the reservation.

[4] We can use $f(x) \stackrel{\mathrm{df}}{=} 0^n10^{n^{i+j}}1x$, since NM_i and NM_j have computation times n^i and n^j, respectively.

Lemma 6.8 *Let (Y, N) be a (μ, k)-reservation for X and let $m \stackrel{\text{df}}{=} \max(\{|w| \,\big|\, w \in Y \cup N\} \cup \{k\})$. If $\|N\| \leq 2^{k/2}$, then there exists a (μ, m)-valid $Z \supseteq_k X$ with $Y \subseteq Z, N \subseteq \overline{Z}$, and $Z^{>k}$ contains only μ-code-words.*

Proof Assume that $\|N\| \leq 2^{k/2}$. We show the lemma by induction on $n \stackrel{\text{df}}{=} m - k$. If $n = 0$, then $Y = N = \emptyset$ and we are done.

Now assume $n > 0$. First of all we want to see that it suffices to find a $(\mu, k+1)$-valid $Z' \supseteq_k X$ such that $Y^{=k+1} \subseteq Z'^{=k+1} \subseteq \overline{N}^{=k+1}$ and $Z'^{=k+1}$ contains only μ-code-words. In this case, Proposition 6.7.4 implies that (Y, N) is a $(\mu, k+1)$-reservation for Z'. Then we can apply the induction hypothesis to (Y, N) considered as a $(\mu, k+1)$-reservation for Z'. We obtain a (μ, m)-valid $Z \supseteq_{k+1} Z'$ such that $Y \subseteq Z \subseteq \overline{N}$ and $Z^{>k+1}$ contains only μ-code-words. Together this yields $Z \supseteq_k X$ and $Z^{>k}$ contains only μ-code-words. It remains to find the mentioned Z'.

If $k+1$ is odd, then $Y^{=k+1} = \emptyset$, since $Y^{=k+1}$ contains only μ-code-words and such words have an even length. By Proposition 6.5.3, X is $(\mu, k+1)$-valid. Therefore, with $Z' \stackrel{\text{df}}{=} X$ we found the desired Z'.

If $k+1$ is even, then, starting with the empty set, we construct a set $S \subseteq \Sigma^{k+1}$ by doing the following for each $(n, i, j) \in \mu$, each $t \geq 1$ and each $x \in \Sigma^*$ with $2 \cdot |00^n 10^t 1x| = k+1$:

- If $NM_i^X(x)$ accepts within t steps, then choose some $y \in \Sigma^{|00^n 10^t 1x|}$ with $00^n 10^t 1xy \notin N$ and add $00^n 10^t 1xy$ to S.
- If $NM_j^X(x)$ accepts within t steps, then choose some $y \in \Sigma^{|10^n 10^t 1x|}$ with $10^n 10^t 1xy \notin N$ and add $10^n 10^t 1xy$ to S.

Observe that the choices of words y are possible since $\|N\| \leq 2^{k/2} < 2^{(k+1)/2} = \|\Sigma^{|00^n 10^t 1x|}\|$. For $Z' \stackrel{\text{df}}{=} X \cup S \cup Y^{=k+1}$ we have $Z' \supseteq_k X$ and $Y^{=k+1} \subseteq Z'^{=k+1} \subseteq \overline{N}^{=k+1}$ since $S \subseteq \overline{N} \cap \Sigma^{k+1}$. Moreover, $Z'^{=k+1}$ contains only μ-code-words since S and $Y^{=k+1}$ do so. It remains to show that Z' is $(\mu, k+1)$-valid.

Claim 1: $A(Z') \cap B(Z') = C(Z') \cap D(Z') = \emptyset$.

Since X is (μ, k)-valid we have $A(X) \cap B(X) = C(X) \cap D(X) = \emptyset$. When we look at the definitions of $A(X)$, $B(X)$, $C(X)$ and $D(X)$ we see that in order to show Claim 1, it suffices to show

$$A(Z') \cap B(Z') \cap \Sigma^{\frac{(k+1)}{2}-1} = C(Z') \cap D(Z') \cap \Sigma^{k+1} = \emptyset.$$

We immediately obtain $C(Z') \cap D(Z') \cap \Sigma^{k+1} = \emptyset$, since by definition, $C(Z')$ and $D(Z')$ contain only words of odd lengths. Assume that $A(Z') \cap B(Z') \cap \Sigma^{(k+1)/2-1} \neq \emptyset$, and choose some $w \in A(Z') \cap B(Z') \cap \Sigma^{(k+1)/2-1}$. So there exist $n, t \geq 1$, $x \in \Sigma^*$ and $y_0, y_1 \in \Sigma^{|w|+1}$ such that $w = 0^n 10^t 1x$ and $0wy_0, 1wy_1 \in Z'$. Since all words in S and all words in Y are μ-code-words, there exist $i, j \geq 1$ such that $(n, i, j) \in \mu$. Note that $0wy_0, 1wy_1 \in S \cup Y^{=k+1}$. We claim that $NM_i^X(x)$ accepts within t steps, regardless of whether $0wy_0$ belongs to S or to $Y^{=k+1}$. This can be seen as follows:

- If $0wy_0 \in S$, then from the construction of S it follows that $NM_i^X(x)$ accepts within t steps.
- If $0wy_0 \in Y^{=k+1}$, then $NM_i(x)$ has a positive path P with $|P| \leq t$, $P^{\text{yes}} \subseteq Y$ and $P^{\text{no}} \subseteq N$. Since $t < k$ it follows that $P^{\text{yes}} \cup P^{\text{no}} \subseteq \Sigma^{\leq k}$ and therefore, $P^{\text{yes}} \subseteq X$ and $P^{\text{no}} \subseteq \Sigma^{\leq k} - X$. It follows that $NM_i^X(x)$ accepts within t steps.

Analogously we obtain that $NM_j^X(x)$ accepts within t steps. Since $|x| \leq k$ it holds that $L(NM_i^X) \cap L(NM_j^X) \cap \Sigma^{\leq k} \neq \emptyset$ and $(i, j) \in \text{range}(\mu)$. This contradicts Proposition 6.5.4 and finishes the proof of Claim 1.

Claim 2: Z' is $(\mu', k+1)$-valid for every $\mu' \preceq \mu$.

We prove the claim by induction on $\|\mu'\|$. If $\|\mu'\| = 0$, then Z' is $(\mu', k+1)$-valid by Claim 1.

Assume now $\|\mu'\| > 0$, and choose suitable μ'', i_0, j_0 such that $\mu' = \mu'' \cup \{(\mu'_{\max}, i_0, j_0)\}$ and $\mu'' \prec \mu$. From the induction hypothesis of this claim it follows that Z' is $(\mu'', k+1)$-valid. Together with $\mu'_{\max} \leq k < k+1$ this shows 6.4.1 for Z' and $(\mu', k+1)$.

Observe that if $(n, i, j) \in \mu'$, $t \geq 1$ and $x \in \Sigma^*$ with $2 \cdot |00^n 10^t 1x| \leq k+1$, then the equivalences in 6.4.2 hold for Z' and $(\mu', k+1)$.

- For $2 \cdot |00^n 10^t 1x| \leq k$ they hold since X is (μ', k)-valid and $Z' \supseteq_k X$.
- For $2 \cdot |00^n 10^t 1x| = k+1$, the implications "$\Leftarrow$" in statement 6.4.2 hold since $S \subseteq Z'$. For the other direction, let $w = 0^n 10^t 1x$ and assume that there exists some $y \in \Sigma^{|w|+1}$ such that $0wy \in Z'$. If $0wy \in S$, then we have put this word to S, because $NM_i^X(x)$ accepts within t steps. Since $t < k$, also $NM_i^{Z'}(x)$ accepts within t steps. If $0wy \in Y^{=k+1}$, then, since (Y, N) is a (μ, k)-reservation for X, $NM_i(x)$ has a positive path P with $|P| \leq t$, $P^{\text{yes}} \subseteq Y$ and $P^{\text{no}} \subseteq N$. Since $t < k$, we have $P^{\text{yes}} \subseteq X$ and $P^{\text{no}} \subseteq \Sigma^{\leq k} - X$. Hence, $NM_i^X(x)$ accepts within t steps, and therefore, $NM_i^{Z'}(x)$ accepts within t steps. This shows the implication "$\Rightarrow$" in 6.4.2.2a. Analogously we see the implication "$\Rightarrow$" in 6.4.2.2b.

Finally, statement 6.4.3 holds for Z' and $(\mu', k+1)$ since X is (μ', k)-valid, $\mu'_{\max} \leq k$ and therefore $Z'^{\leq \mu'_{\max}} = X^{\leq \mu'_{\max}}$. This proves Claim 2.

In particular, Claim 2 implies that Z' is $(\mu, k+1)$-valid. This completes the proof of the lemma. □

One of the main consequences of this lemma is that (μ, k)-valid oracles can be extended to (μ, k')-valid oracles for larger k'. We needed to include the third condition in Definition 6.4 in order to obtain this property. Otherwise it would have been possible that a certain way of extending the finite oracle X to some oracle X' has no extension to an infinite oracle O so that $L(NM_i^O) \cap L(NM_j^O) = \emptyset$. If this were the case, then by statement 6.4.2, for all extensions to an infinite oracle O, $A(O)$ and $B(O)$ would not be disjoint.

Lemma 6.9 *If X is (μ, k)-valid, then for every $m > k$ there exists a (μ, m)-valid $Z \supseteq_k X$.*

Proof It suffices to show the assertion for $m = k+1$. Let $Y = \emptyset$ and $N = 0^{k+1}$. By Proposition 6.7.3, (Y, N) is a (μ, k)-reservation for X. Since $\|N\| = 1 \leq 2^{k/2}$ we can apply Lemma 6.8 and we obtain a $(\mu, k+1)$-valid $Z \supseteq_k X$. □

For a finite $X \subseteq \Sigma^*$, let $\ell(X) \stackrel{df}{=} \sum_{w \in X} |w|$.

Lemma 6.10 *Let X be (μ, k)-valid and let $Z \supseteq_k X$ be (μ, m)-valid such that $m \geq k$ and $Z^{>k}$ contains only μ-code-words. For every $w \in Z$ there exists a (μ, k)-reservation (Y, N) for X such that $w \in Y$, $Y \cup N \subseteq \Sigma^{\leq |w|}$, $\ell(Y \cup N) \leq 2 \cdot |w|$ and $Y \subseteq Z \subseteq \overline{N}$.*

Proof For every $Y \subseteq Z$ let

$$\mathcal{D}(Y) \stackrel{df}{=} \{q \mid Y^{>k} \text{ contains a } \mu\text{-code-word for } (i, t, x) \text{ and } q \in P^{\text{all}}_{i,t,x}\},$$

where $P_{i,t,x}$ is the lexicographically smallest path among all paths of $NM_i^Z(x)$ that are accepting and that are of length $\leq t$. Note that $\mathcal{D}(Y)$ is well-defined: On the one hand we know that all elements of $Z^{>k}$

are μ-code-words. On the other hand, if $Y^{>k} \subseteq Z$ contains a μ-code-word for (i,t,x), then (since Z is (μ,m)-valid) the path $P_{i,t,x}$ really exists.

When looking at the definition of $\mathcal{D}(Y)$ we see that if w is a μ-code-word for (i,t,x), then $|P_{i,t,x}| \leq t < |w|/2$. Therefore, the sum of lengths of q's that are induced by w is at most $|w|/2$. This shows the following.

Claim 1: For all $Y \subseteq Z$: $\ell(\mathcal{D}(Y)) \leq \ell(Y)/2$ and words in $\mathcal{D}(Y)$ are not longer than the longest word in Y.

We compute the (μ,k)-reservation (Y,N) with help of the procedure below.

```
1    Y_0 := {w}
2    N_0 := ∅
3    c := 0
4    do
5         c := c + 1
6         Y_c := D(Y_{c-1}) ∩ Z
7         N_c := D(Y_{c-1}) ∩ Z̄
8    repeat until Y_c = N_c = ∅
9    Y := Y_0 ∪ Y_1 ∪ ··· ∪ Y_c
10   N := N_0 ∪ N_1 ∪ ··· ∪ N_c
```

Note that since all Y_c are subsets of Z, the expressions $\mathcal{D}(Y_{c-1})$ in the lines 6 and 7 are defined. It is immediately clear that $w \in Y \subseteq Z \subseteq \overline{N}$ and therefore $Y \cap N = \emptyset$. From Claim 1 we obtain $Y \cup N \subseteq \Sigma^{\leq |w|}$ and $\ell(Y_i \cup N_i) = \ell(\mathcal{D}(Y_{i-1})) \leq \ell(Y_{i-1})/2$ for $1 \leq i \leq c$. Therefore, $\ell(Y \cup N) \leq 2 \cdot \ell(Y_0) = 2 \cdot |w|$. It remains to show the following.

Claim 2: (Y,N) is a (μ,k)-reservation for X.

Clearly, $Y^{\leq k} \subseteq X \subseteq \overline{N}$. Moreover, all words in $Y^{>k}$ are μ-code-words since all Y_i are subsets of Z. So let $v \in Y^{>k}$ be a μ-code-word for (i,t,x). More precisely, $v \in Y_{i'}$ for a suitable $i' < c$. Since Z is (μ,m)-valid and v is a μ-code-word in Z it follows from Definition 6.4 that $NM_i^Z(x)$ accepts within t steps. Therefore, the path $P_{i,t,x}$ exists and we obtain $P^{\text{all}}_{i,t,x} \subseteq \mathcal{D}(Y_{i'})$. It follows that $P^{\text{yes}}_{i,t,x} \subseteq Y_{i'+1} \subseteq Y$ and $P^{\text{no}}_{i,t,x} \subseteq N_{i'+1} \subseteq N$. Therefore, $NM_i(x)$ has a positive path P with $|P| \leq t$, $P^{\text{yes}} \subseteq Y$ and $P^{\text{no}} \subseteq N$. This proves the claim and finishes the proof of the lemma. □

For any (μ,k)-valid oracle either we can find a finite extension that makes the languages accepted by NM_i and NM_j not disjoint, or we can force these languages to be disjoint for all valid extensions.

Proposition 6.11 *(Property P1) Let $i,j \geq 1$ and let X be (μ,k)-valid. There exists an $l > k$ and a (μ',l)-valid $Y \supseteq_k X$, $\mu \preceq \mu'$ such that*

- *either for all $Z \supseteq_l Y$, $L(NM_i^Z) \cap L(NM_j^Z) \cap \Sigma^{\leq l} \neq \emptyset$*
- *or $(i,j) \in \text{range}(\mu')$.*

This proposition tells us that if the first property does not hold, then by Definition 6.4, since Y is (μ',l)-valid, $L(NM_i^Z) \cap L(NM_j^Z) \cap \Sigma^{\leq m} = \emptyset$ for all (μ,m)-valid extensions Z of Y, where $m \geq l$.

Proof Let $i,j \geq 1$ and let X be (μ,k)-valid. By Lemma 6.9, we can assume that k is large enough so that $2 \cdot k^{i+j} < 2^{k/2}$. If $(i,j) \in \text{range}(\mu)$, then we are done. Otherwise we distinguish two cases.

Case 1: There exists an $l' > k$ and a (μ, l')-valid $Y' \supseteq_k X$ such that $L(NM_i^{Y'}) \cap L(NM_j^{Y'}) \cap \Sigma^{\leq l'} \neq \emptyset$. Choose some $x \in L(NM_i^{Y'}) \cap L(NM_j^{Y'}) \cap \Sigma^{\leq l'}$ and let P_i, P_j be accepting paths of the computations $NM_i^{Y'}(x)$, $NM_j^{Y'}(x)$, respectively. Note that $(P_i^{\text{yes}} \cup P_j^{\text{yes}}) \cap \Sigma^{>l'} = \emptyset$ and let $N \stackrel{df}{=} (P_i^{\text{no}} \cup P_j^{\text{no}}) \cap \Sigma^{>l'}$. By Proposition 6.7, $(\emptyset, N)$ is a (μ, l')-reservation for Y'. Since $\|N\| \leq 2 \cdot |x|^{i+j} \leq 2 \cdot {l'}^{i+j} < 2^{l'/2}$ we can apply Lemma 6.8. We obtain some $l \geq l' > k$ and some (μ, l)-valid $Y \supseteq_{l'} Y' \supseteq_k X$ such that $N \subseteq \Sigma^{\leq l}$ and $Y \subseteq \overline{N}$. Therefore, for every $Z \supseteq_l Y$ the computations $NM_i^Z(x)$ and $NM_j^Z(x)$ will accept at the paths P_i and P_j, respectively. Hence $L(NM_i^Z) \cap L(NM_j^Z) \cap \Sigma^{\leq l} \neq \emptyset$ for every $Z \supseteq_l Y$.

Case 2: For every $l' > k$ and every (μ, l')-valid $Y' \supseteq_k X$ it holds that $L(NM_i^{Y'}) \cap L(NM_j^{Y'}) \cap \Sigma^{\leq l'} = \emptyset$. By Lemma 6.9, there exists a (μ, l)-valid $Y \supseteq_k X$ with $l \stackrel{df}{=} k + 1$. Let $\mu' \stackrel{df}{=} \mu \cup \{(l, i, j)\}$ and observe that $\mu \preceq \mu'$ since $l > k \geq \mu_{\max}$. We will show that Y is (μ', l)-valid.

Since $l = \mu'_{\max}$ and since Y is (μ, l)-valid, 6.4.1 holds. When looking at 6.4.2 for $(l, i, j) \in \mu'$ we realize that $2 \cdot |00^l 10^t 1x| \leq l$ is not possible. Therefore, we only have to verify 6.4.2 for elements from μ. This is immediate, since Y is (μ, l)-valid. Finally, 6.4.3 follows from our assumption in Case 2. Therefore, Y is (μ', l)-valid. □

In order to show that $(C(O), D(O))$ is not symmetric we have to diagonalize against every possible reducing function, i.e., against every deterministic polynomial-time oracle transducer. The following proposition makes sure that this diagonalization is compatible with the notion of valid oracles.

Proposition 6.12 *(Property P2) Let $i \geq 1$ and let X be (μ, k)-valid. There exists an $l > k$ and a (μ, l)-valid $Y \supseteq_k X$ such that for all $Z \supseteq_l Y$, if $C(Z) \cap D(Z) = \emptyset$, then $(C(Z), D(Z))$ does not $\leq_m^{pp,O}$-reduce to $(D(Z), C(Z))$ via T_i^Z.*

Proof By Lemma 6.9 we can assume that $k \equiv 0 (\mathrm{mod}\, 4)$ and $(k+1)^i + 1 < 2^{(k+1)/2}$. Consider the computation $T_i^X(0^{k+1})$, let x be the output of this computation, and let N be the set of queries that are of length greater than k. If $|x| > k$, then additionally we add the word $0^{|x|}$ to N. Note that this yields an N such that $X \cap N = \emptyset$ and $\|N\| \leq (k+1)^i + 1 < 2^{(k+1)/2}$.

If $x \in C(X)$ (note that this means $x = 0^{k'}$ for some $k' \leq k$), then choose some $y \in 0\Sigma^k - N$ and let $S \stackrel{df}{=} \{y\}$. In this case it holds that $0^{k+1} \in C(X \cup S) \wedge x \notin D(X \cup S)$. The latter holds, since X is (μ, k)-valid and therefore, $C(X) \cap D(X) = \emptyset$. Otherwise, if $x \notin C(X)$, then choose some $y \in 1\Sigma^k - N$ and let $S \stackrel{df}{=} \{y\}$. Here we obtain $0^{k+1} \in D(X \cup S) \wedge x \notin C(X \cup S)$. Together this means that we find some $y \in \Sigma^{k+1} - N$ such that with $S \stackrel{df}{=} \{y\}$ it holds that

$$[0^{k+1} \in C(X \cup S) \wedge x \notin D(X \cup S)] \quad \vee \quad [0^{k+1} \in D(X \cup S) \wedge x \notin C(X \cup S)]. \tag{7}$$

$S \subseteq \Sigma^{k+1}$ and $C(S) \cap D(S) = \emptyset$. From Proposition 6.5.3 it follows that $X \cup S$ is $(\mu, k+1)$-valid. So by Proposition 6.7.3, $(\emptyset, N^{>k+1})$ is a $(\mu, k+1)$-reservation for $X \cup S$. Since $\|N^{>k+1}\| < 2^{(k+1)/2}$ we can apply Lemma 6.8. For $l \stackrel{df}{=} \max(\{|w| \,\big|\, w \in N\} \cup \{k+1\})$ we obtain a (μ, l)-valid $Y \supseteq_{k+1} X \cup S$ such that $Y \subseteq \overline{N^{>k+1}}$ and $Y^{>k+1}$ contains only μ-code-words. From $S \subseteq \overline{N}$ it follows that $Y \subseteq \overline{N \cap \Sigma^{>k}}$. Therefore, $T_i^Y(0^{k+1})$ computes x. Since all queries asked at this computation are of length $\leq l$, we obtain that $T_i^Z(0^{k+1})$ computes x for every $Z \supseteq_l Y$. Since $Y^{>k+1}$ does not contain any words of odd length we have $C(Z) \cap \Sigma^{\leq l} = C(X \cup S)$ and $D(Z) \cap \Sigma^{\leq l} = D(X \cup S)$ for each $Z \supseteq_l Y$. Since $0^{|x|} \in N$, we have $0^{k+1}, x \in \Sigma^{\leq l}$. Therefore, by equation (7), the following holds for every $Z \supseteq_l Y$.

$$[0^{k+1} \in C(Z) \wedge T_i^Z(0^{k+1}) \notin D(Z)] \quad \vee \quad [0^{k+1} \in D(Z) \wedge T_i^Z(0^{k+1}) \notin C(Z)] \tag{8}$$

Hence, for every $Z \supseteq_l Y$ with $C(Z) \cap D(Z) = \emptyset$ it holds that $(C(Z), D(Z))$ does not $\leq_m^{pp,O}$-reduce to $(D(Z), C(Z))$ via T_i^Z. □

Recall that we want to construct the oracle in a way such that $(A(O), B(O))$ is not $\leq_T^{pp,O}$-hard for NP^O. At the beginning of this proof we have seen that it suffices to construct $E(O)$ such that it does not $\leq_T^{pp}$-reduce to $(A(O), B(O))$. We prevent $E(O) \leq_T^{pp} (A(O), B(O))$ via M_i as follows: We consider the computation $M_i(0^n)$ where the machine can ask queries to the pair $(A(X), B(X))$. In Lemma 6.13 we show that each query to this pair can be forced either to be in the complement of $A(X)$ or to be in the complement of $B(X)$. For this forcing it is enough to reserve polynomially many words for the complement of X. If we forced the query to be in the complement of $A(X)$, then we can safely answer that it belongs to $B(X)$. Otherwise we can safely answer that it belongs to $A(X)$. After forcing all queries of the computation, we add an unreserved word to $E(X)$ if and only if the computation rejects. This will show that $E(X)$ does not $\leq_T^{pp}$-reduce to $(A(X), B(X))$ via M_i (Proposition 6.14).

Lemma 6.13 *Let $k \equiv 2 (\mathrm{mod}\ 4)$ and let X be (μ, k)-valid. For every $q \in \Sigma^*$, $|q| \leq 2^{k/2-3} - 2$, there exists an $N \subseteq \Sigma^{>k}$ such that $\|N\| \leq (4 \cdot |q| + 5)^2$ and one of the following properties holds.*

1. *For all (μ, m)-valid $Z \supseteq_k X$, if $m > k$, $Z \subseteq \overline{N}$ and $Z^{>k+1}$ contains only μ-code-words, then $q \notin A(Z)$.*
2. *For all (μ, m)-valid $Z \supseteq_k X$, if $m > k$, $Z \subseteq \overline{N}$ and $Z^{>k+1}$ contains only μ-code-words, then $q \notin B(Z)$.*

Proof We can assume that there exist $n, t \geq 1$ and $x \in \Sigma^*$ such that $q = 0^n 10^t 1x$. Otherwise, by definition of A and B, q cannot belong to $A(Z) \cup B(Z)$ for all oracles Z, and we are done. Define the following sets.

$$\begin{aligned} L_A &\stackrel{df}{=} \{(Y_A, N_A) \,\big|\, (Y_A, N_A) \text{ is a } (\mu, k+1)\text{-reservation for some } (\mu, k+1)\text{-valid } Z \supseteq_k X, \\ &\qquad \ell(Y_A \cup N_A) \leq 4 \cdot (|q| + 1), \text{ and } (\exists y \in \Sigma^{|q|+1})[0qy \in Y_A]\} \\ L_B &\stackrel{df}{=} \{(Y_B, N_B) \,\big|\, (Y_B, N_B) \text{ is a } (\mu, k+1)\text{-reservation for some } (\mu, k+1)\text{-valid } Z \supseteq_k X, \\ &\qquad \ell(Y_B \cup N_B) \leq 4 \cdot (|q| + 1), \text{ and } (\exists y \in \Sigma^{|q|+1})[1qy \in Y_B]\} \end{aligned}$$

We say that $(Y_A, N_A) \in L_A$ *conflicts* with $(Y_B, N_B) \in L_B$ if and only if $Y_A \cap N_B \neq \emptyset$ or $N_A \cap Y_B \neq \emptyset$. Note that if (Y_A, N_A) and (Y_B, N_B) conflict, then even $Y_A \cap N_B \cap \Sigma^{>k} \neq \emptyset$ or $N_A \cap Y_B \cap \Sigma^{>k} \neq \emptyset$.

Claim 1: Every $(Y_A, N_A) \in L_A$ conflicts with every $(Y_B, N_B) \in L_B$.

Assume that there exist $(Y_A, N_A) \in L_A$ and $(Y_B, N_B) \in L_B$ that do not conflict. Let $Y' \stackrel{df}{=} Y_A \cup Y_B$, $N' \stackrel{df}{=} N_A \cup N_B$ and $Z' \stackrel{df}{=} X \cup {Y_A}^{=k+1} \cup {Y_B}^{=k+1}$. Since $k+1 \equiv 3 (\mathrm{mod}\ 4)$, this implies $C(Z') \cap D(Z') = \emptyset$. By Proposition 6.5.3, Z' is $(\mu, k+1)$-valid. Observe that (Y', N') is a $(\mu, k+1)$-reservation for Z'. Moreover, from the definition of L_A and L_B it follows that $\|N'\| \leq 8 \cdot |q| + 10 \leq 2^{k/2}$. By Lemma 6.8, there exist an $m \geq k+1$ and a (μ, m)-valid $Z \supseteq_{k+1} Z'$ such that $Y' \subseteq Z$. Since $(Y_A, N_A) \in L_A$ and $(Y_B, N_B) \in L_B$, there exist $y_0, y_1 \in \Sigma^{|q|+1}$ such that $0qy_0 \in Y_A \subseteq Y' \subseteq Z$ and $1qy_1 \in Y_B \subseteq Y' \subseteq Z$. Therefore, $q \in A(Z) \cap B(Z)$, which contradicts the fact that Z is (μ, m)-valid. This proves Claim 1.

We use the following algorithm to create the set N as claimed in the statement of this lemma.

```
1    N := ∅
2    while (L_A ≠ ∅ and L_B ≠ ∅)
3        choose some (Y'_A, N'_A) ∈ L_A
4        N := N ∪ Y'_A^{>k} ∪ N'_A^{>k}
5        for every (Y_A, N_A) ∈ L_A
6            remove (Y_A, N_A) if Y_A ∩ (Y'_A^{>k} ∪ N'_A^{>k}) ≠ ∅
7        for every (Y_B, N_B) ∈ L_B
8            remove (Y_B, N_B) if Y_B ∩ (Y'_A^{>k} ∪ N'_A^{>k}) ≠ ∅
9    end while
```

We claim that after l iterations of the *while* loop, for every $(Y_B, N_B) \in L_B$, $\|N_B\| \geq l$. If this claim is true, the while loop iterates at most $4 \cdot |q| + 5$ times, since for any $(Y_B, N_B) \in L_B$, $\ell(N_B) \leq 4 \cdot |q| + 4$, and therefore, $\|N_B\| \leq 4 \cdot |q| + 5$. On the other hand, during each iteration, N is increased by at most $4 \cdot |q| + 5$ strings. Therefore, $\|N\| \leq (4 \cdot |q| + 5)^2$ and $N \subseteq \Sigma^{>k}$ when this algorithm terminates.

Claim 2: After l iterations of the *while* loop, for every (Y_B, N_B) that remains in L_B, $\|N_B\| \geq l$.

For every l, let us denote the pair that is chosen during the l-th iteration in step 3 by (Y_A^l, N_A^l). By Claim 1, every (Y_B, N_B) that belongs to L_B at the beginning of this iteration conflicts with (Y_A^l, N_A^l), i.e., $N_A^l \cap Y_B \cap \Sigma^{>k} \neq \emptyset$ or $Y_A^l \cap N_B \cap \Sigma^{>k} \neq \emptyset$. If $N_A^l \cap Y_B \cap \Sigma^{>k} \neq \emptyset$, then (Y_B, N_B) will be removed from L_B in step 8. Otherwise, $Y_A^l \cap N_B \cap \Sigma^{>k}$ is not empty, and therefore, there exists a lexicographically smallest word w_l in this set. In this case, (Y_B, N_B) will not be removed from L_B; we say that (Y_B, N_B) *survives* the l-th iteration *due to the word* w_l. Note that (Y_B, N_B) can survive only due to a word that belongs to N_B. We will use this fact to prove that $\|N_B\| \geq l$ after l iterations.

We show now that any pair (Y_B, N_B) that is left in L_B after l iterations survives each of these iteration due to a different word. Since these words all belong to N_B, this will complete the proof of the claim. Assume that there exist iterations l and l' with $l < l'$ such that $w_l = w_{l'}$. Then $w_l \in Y_A^l \cap N_B \cap \Sigma^{>k}$ and $w_{l'} \in Y_A^{l'} \cap N_B \cap \Sigma^{>k}$. Therefore, $Y_A^l \cap Y_A^{l'} \cap \Sigma^{>k} \neq \emptyset$. So the pair $(Y_B^{l'}, N_B^{l'})$ should have been removed in iteration l (step 6), and cannot be chosen at the beginning of iteration l', as claimed. Hence, $w_l \neq w_{l'}$. This proves Claim 2.

Therefore, we now have a set N of the required size such that either L_A or L_B will be empty. Assume that L_A is empty; we will show that 6.13.1 holds (analogously we show that if L_B is empty, then 6.13.2). Assume that for some $m \geq k+1$ there exists a (μ, m)-valid $Z \supseteq_k X$ such that $q \in A(Z)$, $Z \subseteq \overline{N}$ and $Z^{>k+1}$ contains only μ-code-words. Hence, there exists some $y \in \Sigma^{|q|+1}$ such that $0qy \in Z$. [5]

Let $Z' \stackrel{df}{=} Z^{\leq k+1}$. From Proposition 6.5.5 it follows that Z' is $(\mu, k+1)$-valid (since $k+1 > k \geq \mu_{\max}$). Since $Z^{>k+1}$ contains only μ-code-words, we can apply Lemma 6.10. We obtain a $(\mu, k+1)$-reservation (Y', N') for Z' such that $0qy \in Y'$, $Y' \cup N' \subseteq \Sigma^{\leq |0qy|}$, $\ell(Y' \cup N') \leq 2 \cdot |0qy|$ and $Y' \subseteq Z \subseteq \overline{N'}$. Together with $Z \subseteq \overline{N}$, this implies

$$Y' \cap N = \emptyset. \tag{9}$$

Note that (Y', N') must have been in L_A and has been removed during some iteration. This implies that during that iteration, $Y' \cap ({Y'_A}^{>k} \cup {N'_A}^{>k}) \neq \emptyset$ (by line 6). Moreover, by line 4, ${Y'_A}^{>k} \cup {N'_A}^{>k}$ is a subset of N when the algorithms stops. This implies $Y' \cap N \neq \emptyset$, which contradicts equation (9). This shows that for all (μ, m)-valid $Z \supseteq_k X$, if $m > k$, $Z \subseteq \overline{N}$ and $Z^{>k+1}$ contains only μ-code-words, then $q \notin A(Z)$. $\square$

Proposition 6.14 *(Property P3) Let $i \geq 1$ and let X be (μ, k)-valid. There exists an $l > k$ and a (μ, l)-valid $Y \supseteq_k X$ such that for all $Z \supseteq_l Y$, if $A(Z) \cap B(Z) = \emptyset$, then there exists a separator S of $(A(Z), B(Z))$ such that $E(Z) \neq L(M_i^S)$.*

Proof Let $i \geq 1$ and let X be (μ, k)-valid. By Lemma 6.9, we can assume that $k \equiv 2 (\mathrm{mod}\ 4)$ and that k is large enough such that $16(k+3)^{3i} < 2^{k/2}$.

[5] Actually, it even holds that $0qy \in Z - X$, but we do not need this explicitly in our argumentation. In order to see this, we assume that $0qy$ is in X. Then q is in $A(X)$ and $(\{0qy\}, \emptyset)$ is a (μ, k)-reservation for X. Therefore, $(\{0qy\}, \emptyset)$ is a $(\mu, k+1)$-reservation for every $(\mu, k+1)$-valid $Z \supseteq_k X$. Hence, $(\{0qy\}, \emptyset)$ is in L_A at the beginning of the algorithm. So it has been removed during the algorithm. But this is not possible since elements in L_A can only be removed in step 6, and there we remove only (Y_A, N_A) with $Y_A \cap \Sigma^{>k} \neq \emptyset$. This shows $0qy \in Z - X$.

We describe the construction of S_A and S_B, which are sets of queries we reserve for $\overline{B(Y)}$ and $\overline{A(Y)}$, respectively. Let $S_A := A(X)$ and $S_B := B(X)$. We simulate the computation $M_i^{S_A}(0^{k+1})$ until we reach a query q_1 that neither belongs to S_A nor belongs to S_B. Note that $|q_1| \leq (k+1)^i \leq 2^{k/2-3} - 2$. From Lemma 6.13 we obtain some $N_1 \subseteq \Sigma^{>k}$ such that $\|N_1\| \leq (4 \cdot |q_1| + 5)^2$ and either 6.13.1 or 6.13.2 holds. If 6.13.1, then add q_1 to S_B, otherwise add q_1 to S_A. Now return the answer of "$q_1 \in S_A$?" to the computation. We continue the simulation until we reach a query q_2 that neither belongs to S_A nor belongs to S_B. Again we apply Lemma 6.13, obtain the set N_2, and add q_2 either to S_A or to S_B. We continue the simulation until the computation stops. Let n be the number of queries that were added to S_A or S_B. Observe that $S_A \cap S_B = \emptyset$ at the end of our simulation.

Let $N \stackrel{df}{=} N_1 \cup \cdots \cup N_n \cup \{0^{2(k+1)^i+2}\}$. Then $\|N\| \leq (k+1)^i \cdot (4 \cdot (k+1)^i + 5)^2 + 1 \leq 2^{k/2}$. Hence there exists some $w \in \Sigma^{k+1} - N$. If the simulation accepts, then let $Y' \stackrel{df}{=} X$, otherwise let $Y' \stackrel{df}{=} X \cup \{w\}$. By Proposition 6.5.3, Y' is $(\mu, k+1)$-valid. Since $N \subseteq \Sigma^{>k}$ we have $N \subseteq \overline{Y'}$. Therefore, by Proposition 6.7.3, $(\emptyset, N)$ is a $(\mu, k+1)$-reservation for Y'. By Lemma 6.8, there exist an $l \geq 2(k+1)^i + 2$ and a (μ, l)-valid $Y \supseteq_{k+1} Y'$ such that $Y \subseteq \overline{N}$ and $Y^{>k+1}$ contains only μ-code-words. In particular, it holds that $l > k$ and $Y \supseteq_k X$.

Claim 1: $S_A \subseteq \overline{B(Z)}$ and $S_B \subseteq \overline{A(Z)}$ for every $Z \supseteq_l Y$.

Assume that $S_A \cap B(Z) \neq \emptyset$ for some $Z \supseteq_l Y$, and choose a $v \in S_A \cap B(Z)$. Since S_A contains only words of length $\leq (k+1)^i$ we obtain $v \in S_A \cap B(Z^{\leq 2(k+1)^i+2}) = S_A \cap B(Y)$. So v cannot belong to $A(Y)$ since $A(Y) \cap B(Y) = \emptyset$. In particular this means $v \in S_A - A(X)$, i.e., $v = q_j$ for a suitable j with $1 \leq j \leq n$. By our construction q_j was only added to S_A when 6.13.2 holds. Remember that Y is (μ, l)-valid with $l > k$, $Y \supseteq_k X$, $Y \subseteq \overline{N} \subseteq \overline{N_j}$ and $Y^{>k+1}$ contains only μ-code-words. Therefore, from 6.13.2 it follows that $v = q_j \notin B(Y)$, which contradicts $v \in S_A \cap B(Y)$. This shows $S_A \subseteq \overline{B(Z)}$. By the symmetric argument we obtain $S_B \subseteq \overline{A(Z)}$. This proves the claim.

Consider any $Z \supseteq_l Y$ with $A(Z) \cap B(Z) = \emptyset$. Let $S \stackrel{df}{=} A(Z) \cup S_A$. Assume that S is not a separator of $(A(Z), B(Z))$. Since $A(Z) \subseteq S$, we must have $S \cap B(Z) \neq \emptyset$. Since $A(Z) \cap B(Z) = \emptyset$, this implies $S_A \cap B(Z) \neq \emptyset$. This contradicts Claim 1. So S is a separator of $(A(Z), B(Z))$. It remains to show $E(Z) \neq L(M_i^S)$.

By our construction, $0^{k+1} \in E(Y')$ if and only if $M_i^{S_A}(0^{k+1})$ rejects. Since $Z \supseteq_{k+1} Y'$ it holds that $0^{k+1} \in E(Z)$ if and only if $M_i^{S_A}(0^{k+1})$ rejects. Assume that there exists a query q that is answered differently in the computations $M_i^{S_A}(0^{k+1})$ and $M_i^S(0^{k+1})$ (take the first such query). Since $S_A \subseteq S$ we obtain $q \in S - S_A$, i.e., $q \in A(Z)$. If q is in $B(X)$, then q is in $B(Z) \subseteq \overline{S}$, which is not possible. So q is neither in S_A nor in $B(X)$, but q is asked in the computation $M_i^{S_A}(0^{k+1})$. It follows that $q = q_j$ for some j with $1 \leq j \leq n$, and during the construction we added q_j to S_B. So we have $q \in S_B \cap A(Z)$, which contradicts Claim 1. Therefore, $M_i^{S_A}(0^{k+1})$ accepts if and only if $M_i^S(0^{k+1})$ accepts. This shows $0^{k+1} \in E(Z)$ if and only if $M_i^S(0^{k+1})$ rejects, i.e., $E(Z) \neq L(M_i^S)$. □

This finishes the proof of Theorem 6.1. □

Corollary 6.2 *The oracle O from Theorem 6.1 has the following additional properties.*

(i) $\mathrm{UP}^O \neq \mathrm{NP}^O \neq \mathrm{coNP}^O$ *and* $\mathrm{NPMV}^O \not\subseteq_c \mathrm{NPSV}^O$

(ii) There exists a $\leq_m^{pp}$-complete NP^O-pair (A, B) that satisfies the following:

- *For every NP^O-pair (E, F) there exists an $f \in \mathrm{FP}$ with $E \leq_m^p A$ via f and $F \leq_m^p B$ via f.*

– (A, B) *is* P^O*-inseparable but symmetric.*

Proof It is known that Conjecture 2.4 implies item (i) [ESY84, GS88, Sel94]. The first statement of (ii) follows immediately from the proof of Theorem 6.1 (equation (4)). The pair (A, B) is symmetric because it is $\leq_m^{pp}$-complete. If (A, B) is P^O-separable, then every NP^O-pair is P^O-separable, and therefore symmetric. This contradicts item (ii) of Theorem 6.1. So (A, B) is P^O-inseparable. □

Note that statement (ii) shows that (A, B) is complete even in a stronger sense of many-one reductions.