Oracle with $\mathsf{P} = \mathsf{NP} \cap \mathsf{coNP}$, but No Many-One Completeness in UP, DisjNP, and DisjCoNP

Anton Ehrmanntraut
Julius-Maximilians-Universität Würzburg, Germany

Fabian Egidy
Julius-Maximilians-Universität Würzburg, Germany

Christian Glaßer
Julius-Maximilians-Universität Würzburg, Germany

Abstract

We construct an oracle relative to which $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}$, but there are no many-one complete sets in UP, no many-one complete disjoint NP-pairs, and no many-one complete disjoint coNP-pairs.

This contributes to a research program initiated by Pudlák [33], which studies incompleteness in the finite domain and which mentions the construction of such oracles as open problem. The oracle shows that $\mathsf{NP} \cap \mathsf{coNP}$ is indispensable in the list of hypotheses studied by Pudlák. Hence one should consider stronger hypotheses, in order to find a universal one.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Oracles and decision trees; Theory of computation → Proof complexity

Keywords and phrases computational complexity, promise classes, proof complexity, complete sets, oracle construction

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.45

Related Version *Full Version*: `https://arxiv.org/abs/2203.11079`

1 Introduction

Questions of the existence of complete sets in promise classes have a long history. They turned out to be difficult and remained open. Consider the following examples, where the questions are expressed as hypotheses.

$$
\begin{aligned}
\mathsf{NP} \cap \mathsf{coNP} &: \mathrm{NP} \cap \mathrm{coNP} \text{ does not contain many-one complete sets } [23] \\
\mathsf{UP} &: \mathrm{UP} \text{ does not contain many-one complete sets } [22] \\
\mathsf{CON} &: \text{p-optimal proof systems for TAUT do not exist } [26] \\
\mathsf{SAT} &: \text{p-optimal proof systems for SAT do not exist } [15] \\
\mathsf{TFNP} &: \mathrm{TFNP} \text{ does not contain many-one complete problems } [27] \\
\mathsf{DisjNP} &: \mathrm{DisjNP} \text{ does not contain many-one complete pairs } [34] \\
\mathsf{DisjCoNP} &: \mathrm{DisjCoNP} \text{ does not contain many-one complete pairs } [28, 32]
\end{aligned}
$$

So far, the following implications are known: $\mathsf{DisjNP} \Rightarrow \mathsf{CON}$ [34], $\mathsf{UP} \Rightarrow \mathsf{CON}$ [25], $\mathsf{DisjCoNP} \Rightarrow \mathsf{TFNP}$ [33], $\mathsf{TFNP} \Rightarrow \mathsf{SAT}$ [5, 33], and $\mathsf{NP} \cap \mathsf{coNP} \Rightarrow \mathsf{CON} \vee \mathsf{SAT}$ [25]. This raises the question of whether further implications are provable with the currently available means. Thanks to a work by Pudlák [33], this question recently gained momentum. In fact, Pudlák's interest goes beyond: He initiated a research program to find a general principle from which the remaining hypotheses follow as special cases. This is motivated by the study of incompleteness in the finite domain, since these hypotheses can either be expressed as the non-existence of complete elements in promise classes or as statements about the unprovability of sentences of some specific form in weak theories.

Pudlák [33] states as open problem to construct oracles that show that the relativized conjectures are different or show that they are equivalent. Such oracles have been constructed by Verbitskii [38], Glaßer et al. [18], Khaniki [24], Dose [10, 11, 9], and Dose and Glaßer [12]. The restriction to relativizable proofs arises from the following idea: We consider the mentioned hypotheses as conjectures, hence we expect that they are equivalent. In this situation we are not primarily concerned with the question of whether two hypotheses are equivalent, but rather whether their equivalence can be *recognized* with the currently available means. An accepted formalization of this is the notion of relativizable proofs.

Our Contribution. We construct an oracle relative to which the following holds: UP, DisjNP, $\mathsf{DisjCoNP}$, but $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}$, which implies $\neg\mathsf{NP} \cap \mathsf{coNP}$. Hence there is no relativizable proof for $\mathsf{NP} \cap \mathsf{coNP}$, even if we simultaneously assume all remaining hypotheses we mentioned so far. This demonstrates that $\mathsf{NP} \cap \mathsf{coNP}$ is indispensable in the list of currently viewed hypotheses and suggests to broaden the focus and include stronger statements.

Pudlák [33] ranks $\mathsf{NP} \cap \mathsf{coNP}$ as a plausible conjecture that is apparently incomparable with CON and TFNP. Our oracle supports this estimation, as it rules out relativizable proofs for "$\mathsf{CON} \Rightarrow \mathsf{NP} \cap \mathsf{coNP}$" and "$\mathsf{TFNP} \Rightarrow \mathsf{NP} \cap \mathsf{coNP}$." By Dose [10, 11], the same holds for the converse implications. Overall, we recognize a strong independence between $\mathsf{NP} \cap \mathsf{coNP}$ and all remaining hypotheses:

(i) There does not exist a relativizable proof for $\mathsf{NP} \cap \mathsf{coNP}$, even if we simultaneously assume all remaining hypotheses.

(ii) There exists a relativizable proof for the implication $\mathsf{NP} \cap \mathsf{coNP} \Rightarrow \mathsf{CON} \vee \mathsf{SAT}$ [25]. But there does not exist a relativizable proof showing that $\mathsf{NP} \cap \mathsf{coNP}$ implies one of the remaining hypotheses [10, 11].

Our oracle combines several separations with the collapse $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}$. This leads to conclusions on the independence of the statement $\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP}$ from typical assumptions. For instance, the oracle shows that $\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP}$ cannot be proved by relativizing means, even under the strong but likely assumption $\mathsf{UP} \wedge \mathsf{DisjNP} \wedge \mathsf{DisjCoNP}$.

Further characteristics of our oracle are, for example, $\mathrm{NE} \neq \mathrm{coNE}$, $\mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$, and the shrinking and separation properties do not hold for NP and coNP. Corollary 10 presents a list of additional properties.

Open Questions. Currently, for almost every pair A, B of hypotheses, we either know a relativizable proof for the implication $\mathsf{A} \Rightarrow \mathsf{B}$, or we know an oracle relative to which $\mathsf{A} \wedge \neg\mathsf{B}$. Only three cases are left: (1) $\mathsf{UP} \overset{?}{\Rightarrow} \mathsf{DisjNP}$, (2) $\mathsf{TFNP} \overset{?}{\Rightarrow} \mathsf{DisjCoNP}$, and (3) $\mathsf{SAT} \overset{?}{\Rightarrow} \mathsf{TFNP}$. This leads to the following task for future research: Prove these implications or construct oracles relative to which they do not hold.

Background on Connections Between Promise Classes and Proof Systems. Informally, promise classes are complexity classes that are characterized by machines that satisfy certain properties. Usually, these properties are hard or even impossible to validate. Thus, when working with an element of a promise class, one has to trust the *promise* that the respective machine has said property. We are mainly interested in the following well-studied promise classes: The class of disjoint NP-pairs $\mathrm{DisjNP} = \{(A, B) \mid A, B \in \mathrm{NP}, A \cap B = \emptyset\}$ [35, 21], the class of disjoint coNP-pairs DisjCoNP [14, 15] (defined respectively), the class of sets accepted by nondeterministic polynomial-time machines with at most one accepting computation path UP [37], the class $\mathrm{NP} \cap \mathrm{coNP}$ [13], and the class of all total polynomial search problems TFNP [27]. As an example, the machines characterizing UP promise that on every input,

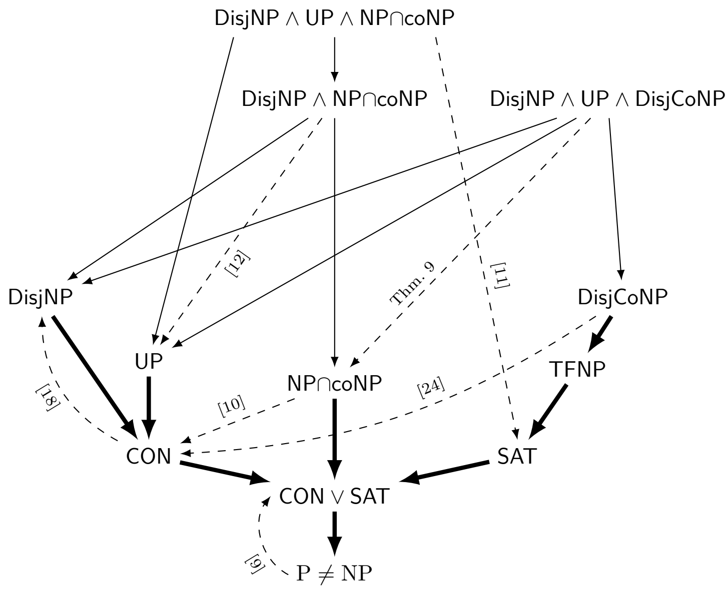

Figure 1 Solid arrows mean implications. All implications occurring in the figure have relativizable proofs. (The only nontrivial ones are DisjNP ⇒ CON [34], UP ⇒ CON [25, Cor. 4.1], DisjCoNP ⇒ TFNP ⇒ SAT [33, Prop. 5.6][5, Thm. 25][33, Prop. 5.10].) Implications between the conjectures originally considered by Pudlák (i.e., not the conjunctions) are highlighted bold. A dashed arrow from one conjecture A to another conjecture B means that there is an oracle X against the implication A ⇒ B, meaning that A ∧ ¬B holds relative to X.

they either reject on all computation paths, or accept on exactly one computation path. Furthermore, we are interested in proof systems defined by Cook and Reckhow [8], especially optimal and p-optimal proof systems for the set of satisfiable formulas SAT, for the set of tautologies TAUT.

The connections between propositional proof systems and promise classes have been studied intensively. Krajícek and Pudlák [26] linked propositional proof systems (and thus the hypothesis CON) to standard complexity classes by proving that NE = coNE implies the existence of optimal propositional proof systems and E = NE implies the existence of p-optimal propositional proof systems. These results were subsequently improved by Köbler, Messner, and Torán [25].

Glaßer, Selman, and Sengupta [17] give several characterizations of DisjNP. Some characterizations use different notions of reducibility while others use the existence of $\leq_m^p$-complete functions in NPSV and the uniform enumerability of disjoint NP-pairs. Glaßer, Selman, and Zhang [19, 20] connect propositional proof systems to disjoint NP-pairs. They prove that the degree structure of DisjNP and of all canonical disjoint pairs of propositional proof systems is the same. Beyersdorff [1, 2, 3, 4] and Beyersdorff and Sadowkski [6] investigate further connections between disjoint NP-pairs and propositional proof systems.

Pudlák [30, 31, 33] draws connections between the finite consistency problem, proof systems, and promise classes like DisjNP and TFNP. Moreover, he asks for oracles that separate hypotheses regarding proof systems and promise classes. Several oracles have been constructed since Pudlák formulated his research questions. Concerning the listed hypotheses, Figure 1 summarizes all known (relativizing) implications and implications that do not hold relative to some oracle.

The paper is organized as follows: Section 2 defines the complexity classes mentioned above and presents our notations. Section 3 contains the oracle construction: the first part defines the construction, the second part proves that it is well-defined, and the last part shows the claimed properties.

2 Preliminaries

Basic Notation. Throughout this paper, let Σ be the alphabet $\{0,1\}$. The set Σ^* denotes the set of finite words over Σ. The set Σ^ω denotes the set of ω-infinite words, i.e., the ω-infinite sequences of characters from Σ. Let $\Sigma^{\leq n} := \{w \in \Sigma^* \mid |w| \leq n\}$. For a word $w \in \Sigma^* \cup \Sigma^\omega$, we denote with $w(i)$ the i-th character of w for $0 \leq i < |w| \leq \omega$. We write $v \sqsubseteq w$ when v is a prefix of w, that is, $|v| \leq |w|$ and $v(i) = w(i)$ for all $0 \leq i < |v|$. Accordingly, $v \sqsubsetneq w$ when $v \sqsubseteq w$ and $v \neq w$. The empty word is denoted by ε. For a finite set $A \subseteq \Sigma^*$, we define $\ell(A) := \sum_{w \in A} |w|$.

Let $\mathbb{N}$ denote the set of non-negative integers, and $\mathbb{N}^+$ the set of positive integers. We say that two sets X and Y *agree on set* Z when $X \cap Z = Y \cap Z$.

The finite words Σ^* can be linearly ordered by their quasi-lexicographic (i.e., "shortlex") order $\prec_{\text{lex}}$, uniquely defined by requiring $0 \prec_{\text{lex}} 1$. Under this definition, there is a unique order-isomorphism between $(\Sigma^*, \prec_{\text{lex}})$ and $(\mathbb{N}, <)$, which induces a polynomial-time computable, polynomial-time invertible bijection between Σ^* and $\mathbb{N}$. Hence, we can transfer the notations, relations, and operations for Σ^* to $\mathbb{N}$ and vice versa. In particular, $|n|$ denotes the length of the word represented by $n \in \mathbb{N}$. By definition of $\prec_{\text{lex}}$, whenever $a \leq b$, then $|a| \leq |b|$. We eliminate the ambiguity of the expressions 0^i and 1^i by always interpreting them over Σ^i. Moreover, $<$ denotes both the less-than relation for natural numbers and the quasi-lexicographic order $\prec_{\text{lex}}$ for finite words. Similarly for $\leq$ and $\preceq_{\text{lex}}$. From the properties of order-isomorphism, this is compatible with the above identification of words and numbers.

Complexity Classes. We understand P (resp., NP) as the usual complexity class of languages decidable by a deterministic (resp., nondeterministic) polynomial-time Turing machine. The class FP refers to the class of total functions that can be computed by a deterministic polynomial-time Turing transducer [29]. Valiant [37] defined UP as the set of all languages that can be recognized by a nondeterministic polynomial-time machine that, on every input, accepts on at most one computation path. For a complexity class $\mathcal{C}$ we define $\text{co}\mathcal{C} := \{\overline{A} \mid A \in \mathcal{C}\}$ as the complementary complexity class of $\mathcal{C}$. Between sets of words, we employ the usual *polynomial-time many-one reducibility*: $A \leq_{\text{m}}^{\text{p}} B$ if there exists an $f \in \text{FP}$ such that $x \in A \Leftrightarrow f(x) \in B$. The usual notion of $\leq_{\text{m}}^{\text{p}}$-completeness and -hardness follows.

A *disjoint* NP*-pair* is a pair (A, B) of disjoint sets in NP. Selman [35] and Grollmann and Selman [21] defined the class DisjNP as the set of disjoint NP-pairs. The classes DisjCoNP [14, 15], DisjUP, and DisjCoUP are defined similarly. Between two pairs, we employ the following related notion of reducibility [34, 18]: Let (A, B) and (C, D) be two disjoint pairs. We say that (A, B) is *polynomial-time many-one reducible to* (C, D), denoted by $(A, B) \leq_{\text{m}}^{\text{pp}} (C, D)$, if there is a function $h \in \text{FP}$ such that $h(A) \subseteq C$ and $h(B) \subseteq D$. The terms $\leq_{\text{m}}^{\text{pp}}$-completeness and -hardness also follow directly from this definition of reduction.

Proof Systems. We use the notion of proof systems for sets by Cook and Reckhow [8]: A function $f \in \text{FP}$ is called a *proof system for* $\text{img}(f)$. Specifically, a proof system f for TAUT is a *propositional* proof system. We say that a proof system g is *(p-)simulated* by a proof system f, denoted by $f \leq g$ (resp., $f \leq^{\text{p}} g$), if there exists a total function π (resp.,

$\pi \in \mathrm{FP}$) and a polynomial p such that $|\pi(x)| \leq p(|x|)$ and $f(\pi(x)) = g(x)$ for all x. We call a proof system f *(p-)optimal* for the set $\mathrm{img}(f)$, if $g \leq f$ (resp., $g \leq^{\mathrm{p}} f$) for all $g \in \mathrm{FP}$ with $\mathrm{img}(g) = \mathrm{img}(f)$.

Relativizations. We can relativize each complexity and function class to some oracle O, by equipping all machines corresponding to the respective class with oracle access to O. That is, e.g., $\mathrm{UP}^O \coloneqq \{L(M^O) \mid M$ is a nondeterministic polynomial-time oracle Turing machine, and for all inputs x, $M^O(x)$ accepts on at most one path$\}$. The classes P^O, NP^O and so on are defined similarly. We can also relativize our notions of reducibility by using functions from FP^O instead of FP. In other words, we allow the reduction functions to access the oracle in relativized instances. This results in polynomial-time many-one reducibilities relative to an oracle O, which we denote as $\leq_{\mathrm{m}}^{\mathrm{p},O}$ for sets and $\leq_{\mathrm{m}}^{\mathrm{pp},O}$ for pairs of disjoint sets. In the same way, we can relativize (p-)simulation of proof systems to some oracle O, and denote the relativized simulation as $\leq^O$ resp. $\leq^{\mathrm{p},O}$. When it is clear from context that some statements refer to the relativized ones relative to some fixed oracle O, we sometimes omit the indication of O in the superscripts.

We define $p_i(n) \coloneqq n^i + i$. Let $\{M_i\}_{i\in\mathbb{N}}$ and $\{F_i\}_{i\in\mathbb{N}}$ be, respectively, standard enumerations of nondeterministic polynomial-time (oracle) Turing machines resp. deterministic polynomial-time (oracle) Turing transducers, having the property that runtime of M_i, F_i is bounded by p_i relative to any oracle. Note that $\{L(M_i^O) \mid i \in \mathbb{N}\} = \mathrm{NP}^O$, $\{F_i^O \mid i \in \mathbb{N}\} = \mathrm{FP}^O$.

Specific Notation Used in our Oracle Construction. We now take on the notations proposed by Dose and Glaßer [12] designed for the construction of oracles. The domain of definition, image, and support for partial function $t\colon A \to \mathbb{N}$ are defined as $\mathrm{dom}(t) \coloneqq \{x \in A \mid t(x) \text{ defined}\}$, $\mathrm{img}(t) \coloneqq \{t(x) \mid x \in A, t(x) \text{ defined}\}$, $\mathrm{supp}(t) \coloneqq \{x \in A \mid t(x) \text{ defined and } t(x) > 0\}$. We say that t is *injective on its support* if, for any $a, b \in \mathrm{supp}(t)$, $t(a) = t(b)$ implies $a = b$. If t is not defined at point x, then $t \cup \{x \mapsto y\}$ denotes the extension t' of t that at x has value y and satisfies $\mathrm{dom}(t') = \mathrm{dom}(t) \cup \{x\}$.

For a set A, we denote with $A(x)$ the characteristic function at point x, i.e., $A(x)$ is 1 if $x \in A$, and 0 otherwise. We can identify an oracle $A \subseteq \mathbb{N}$ with its characteristic ω-word $A(0)A(1)A(2)\cdots$ over Σ^ω. In this way, $A(i)$ denotes both the characteristic function at point i and the i-th character of its characteristic word. Similarly, for a finite word $w \in \Sigma^*$, we also understand w as the set $\{i \mid w(i) = 1\}$ and, e.g., we write $A = w \cup B$ where A and B are sets. (However, we understand $|w|$ as the length of the word w, and not the cardinality of set $\{i \mid w(i) = 1\}$.) Thus, a finite word w describes an oracle which is partially defined, i.e., only defined for natural numbers (or equivalently words) $x < |w|$. Being able to interpret a word w as a set and partial oracle is very useful for the oracle construction. In most construction steps we decide the membership of the smallest undefined word of a partial oracle w, which is simply $|w|$. This gives access to very concise notation.

In particular, for oracle machines M, the notation $M^w(x)$ refers to $M^{\{i \mid w(i)=1\}}(x)$ (that is, oracle queries that w is not defined for are negatively answered). This also allows us to define the following notion: we say that $M^w(x)$ is *definite* if all queries on all computation paths are $< |w|$ (or equivalently: $w(q)$ is defined for all queries q on all computation paths); we say that $M^w(x)$ *definitely accepts* (resp., *definitely rejects*) if $M^w(x)$ is definite and accepts (resp., rejects). Intuitively, the term definite describes computations that do not change when extending the respective oracle, because the queries are too short. This allows the following observation:

▶ **Observation 1.**
(i) *When $M^w(x)$ is a definite computation, and $v \sqsupseteq w$, then $M^v(x)$ is definite. Computation $M^v(x)$ accepts if and only if $M^w(x)$ accepts.*
(ii) *When w is defined for all words of length $p_i(|x|)$, then $M_i^w(x)$ is definite.*
(iii) *When $M^w(x)$ accepts on some computation path with set of oracle queries Q, and w, v agree on Q, then $M^v(x)$ accepts on the same computation path and with the same set of oracle queries Q.*

For an oracle w, a transducer F, and a machine M, we occasionally understand the notation $M^w(F^w(x))$ as the single computation of the machine $M \circ F$ on input x relative to w. Consequently, we say that $M^w(F^w(x))$ definitely accepts (resp., rejects) when $M \circ F$ definitely accepts (resp., rejects) input x relative to w.

In our oracle construction, we want to injectively reserve and assign countably infinitely many *levels* n, that are, words of same length n, for a countably infinite family of witness languages, with increasingly large gaps. For this, let $e(0) := 2$, $e(i) := 2^{e(i-1)}$. There is a polynomial-time computable, polynomial-time invertible injective function f, mapping $(m, h) \in \mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. Now define $H_m := \{e(f(m, h)) \mid h \in \mathbb{N}\}$ as the set of levels reserved for witness language m. This definition ensures

▶ **Observation 2.**
(i) *The set H_m is countably infinite, a subset of the even numbers, and all $H_0, H_1, \ldots$ are pairwise disjoint.*
(ii) *The sequence $\min H_0, \min H_1, \ldots$ is unbounded.*
(iii) *When $n \in H_m$, then $n < n' < 2^n$ implies $n' \notin H_0, H_1, \ldots$.*
(iv) *Every set $H_m \in \mathrm{P}$ for all $m \in \mathbb{N}$.*

3 Oracle Construction

We are primarily interested in an oracle O with the property that relative to that oracle, UP, DisjNP, DisjCoNP, and ¬NP ∩ coNP hold, but our construction yields the following slightly stronger statements:
(i) NP ∩ coNP = P (implying ¬NP ∩ coNP).
(ii) DisjNP does not contain $\leq_m^{pp}$-hard pairs for DisjUP (implying DisjNP).
(iii) UP does not contain $\leq_m^{p}$-complete languages (i.e., UP).
(iv) DisjCoNP does not contain $\leq_m^{pp}$-hard pairs for DisjCoUP (implying DisjCoNP).
Given a (possibly partial) oracle O and $m \in \mathbb{N}$, we define the following witness languages:

$$A_m^O := \{0^n \mid n \in H_m, \text{there exists } x \in \Sigma^n \text{ such that } x \in O \text{ and } x \text{ ends with } 0\}$$
$$B_m^O := \{0^n \mid n \in H_m, \text{there exists } x \in \Sigma^n \text{ such that } x \in O \text{ and } x \text{ ends with } 1\}$$
$$C_m^O := \{0^n \mid n \in H_m, \text{there exists } x \in \Sigma^n \text{ such that } x \in O\}$$
$$D_m^O := \{0^n \mid n \in H_m, \text{for all } x \in \Sigma^n,\ x \in O \to x \text{ ends with } 0\}$$
$$E_m^O := \{0^n \mid n \in H_m, \text{for all } x \in \Sigma^n,\ x \in O \to x \text{ ends with } 1\}$$

Their purpose is to be a “witness” that an element of DisjNP (resp., UP, DisjCoNP) is not complete by admitting no reduction to this element. This only works if the witness languages themselves belong to the respective classes. The following observation shows how the membership of the witness languages to the respective classes depends on the oracle.

▶ **Observation 3.**

(i) *If for all $n \in H_m$, $|O \cap \Sigma^n| \leq 1$, then (A_m^O, B_m^O) is in DisjUP^O, and C_m^O is in UP^O.*

(ii) *If for all $n \in H_m$, $O \cap \Sigma^n$ contains at least one word but not two words with the same parity, (i.e., there exists $\alpha \in \Sigma^{n-1}0$, $\beta \in \Sigma^{n-1}1$ such that the set $O \cap \Sigma^n$ is equal to $\{\alpha\}$ or $\{\beta\}$ or $\{\alpha, \beta\}$), then (D_m^O, E_m^O) is in $\mathrm{DisjCoUP}^O$.*

Preview of the Construction. The construction is quite technical, since the oracle has to satisfy several properties which simultaneously demand structure (i.e., property (i)) and freedom (i.e., properties (ii), (iii) and (iv)) during the construction. This leads to several dependencies and special cases that need to be addressed, mostly by combinatorial arguments and various extensions of the oracle constructed so far. To keep track of the progress of the construction, it is divided into tasks corresponding to the desired properties (i)–(iv). Each task contributes to the goal of satisfying its corresponding property.

1. Work towards $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}$: For all $a \neq b$, the construction tries to achieve that M_a, M_b do not accept complementary. (*Accepting complementary* should mean that for each input x, precisely one of $M_a(x), M_b(x)$ accepts and the other rejects.) If this is not possible, (M_a, M_b) inherently accept complementary, and thus $L(M_a) \in \mathrm{NP} \cap \mathrm{coNP}$. Then, we start to *encode* into the oracle, whether M_a accepts some inputs or not. Thus, the final oracle will contain the encodings for almost all inputs, thus allowing to recover the accepting behavior of M_a and hence to decide $L(M_i)$ in P using oracle queries.
2. Work towards (ii), which implies DisjNP: For all $i \neq j$, the construction tries to achieve that M_i, M_j both accept some input x, hence $x \in L(M_i) \cap L(M_j)$ and $(L(M_i), L(M_j)) \notin \mathrm{DisjNP}$. If this is not possible, (M_i, M_j) inherently is a disjoint NP-pair. In this case, we fix some m, make sure that (A_m, B_m) is a disjoint UP-pair and diagonalize against every transducer F_r, so that F_r does not realize the reduction $(A_m, B_m) \leq_{\mathrm{m}}^{\mathrm{pp}} (L(M_i), L(M_j))$. This is achieved by, (i) for all $n \in H_m$, insert at most one word of length n into O (and thus $(A_m, B_m) \in \mathrm{DisjUP}$), and (ii) for every r there is an $n \in H_m$ such that $0^n \in A_m$ but $M_i(F_r(0^n))$ rejects (or analogously $0^n \in B_m$ but $M_j(F_r(0^n))$ rejects).
3. Work towards (iii), i.e., UP: Try to make M_i accept on two separate paths. If this is not possible, then $L(M_i)$ inherently is a UP-language. In this case, we fix some m, make sure that C_m is a language in UP and diagonalize against every transducer F_r so that F_r does not realize the reduction $C_m \leq_{\mathrm{m}}^{\mathrm{p}} L(M_i)$. This is achieved by, (i) for all $n \in H_m$, insert at most one word of length n into O (and thus $C_m \in \mathrm{UP}$), and (ii) for every r there is an $n \in H_m$ such that $0^n \in C_m$ if and only if $M_i(F_r(0^n))$ rejects.
4. Work towards (iv), which implies DisjCoNP: Try to achieve that M_i, M_j both reject some input x, hence $x \in \overline{L(M_i)} \cap \overline{L(M_j)}$ and $(L(M_i), L(M_j)) \notin \mathrm{DisjNP}$. If this is not possible, (M_i, M_j) inherently is a disjoint coNP-pair. In this case, we fix some m, make sure that (D_m, E_m) is a disjoint coUP-pair and diagonalize against every transducer F_r, so that F_r does not realize the reduction $(D_m, E_m) \leq_{\mathrm{m}}^{\mathrm{pp}} (L(M_i), L(M_j))$. This is achieved by, (i) for all $n \in H_m$, insert at least one word of length n into O but not two words with same parity (and thus $(D_m, E_m) \in \mathrm{DisjCoUP}$), and (ii) for every r there is an $n \in H_m$ such that $0^n \in D_m$ but $M_i(F_r(0^n))$ accepts (or analogously $0^n \in E_m$ but $M_j(F_r(0^n))$ accepts).

To these requirements, we assign the following symbols representing tasks: $\tau_{a,b}^1$, $\tau_{i,j}^2$, $\tau_{i,j,r}^2$, τ_i^3, $\tau_{i,r}^3$, $\tau_{i,j}^4$, $\tau_{i,j,r}^4$ for all $a, b, i, j, r \in \mathbb{N}, i \neq j$, $a \neq b$. The symbol $\tau_{a,b}^1$ represents the coding or the destruction of $\mathrm{NP} \cap \mathrm{coNP}$-pairs. The symbol $\tau_{i,j}^2$ represents the destruction of a disjoint NP-pair, $\tau_{i,j,r}^2$ the diagonalization of that pair against transducer F_r. Analogously for UP and $\tau_i^3, \tau_{i,r}^3$. Analogously for DisjCoNP and $\tau_{i,j}^4, \tau_{i,j,r}^4$.

For the coding, we injectively define the code word $c(a,b,x) \coloneqq 0^a10^b10^l10^p1x$ with $p = p_a(|x|) + p_b(|x|)$, $l \in \mathbb{N}$ minimal such that $l \geq 7/8|c(a,b,x)|$ and $c(a,b,x)$ has odd length. By this, a code word contains the word x as information and is padded to sufficient length. We call any word of the form $c(\cdot,\cdot,\cdot)$ a *code word*. This ensures the following properties:

▷ Claim 4. For all $a, b \in \mathbb{N}$, $x \in \Sigma^*$, the following holds:

(i) $|c(a,b,x)| \notin H_m$ for any m.

(ii) For fixed a, b, the function $x \mapsto c(a,b,x)$ is polynomial-time computable, and polynomial-time invertible with respect to $|x|$.

(iii) Relative to any oracle, the running times of $M_a(x)$ and $M_b(x)$ are both bounded by $< |c(a,b,x)|/8$.

(iv) For every partial oracle $w \in \Sigma^*$, if $c(a,b,x) \leq |w|$, then $M_a^w(x)$ and $M_b^w(x)$ are definite.

During the construction we successively add requirements that we maintain. To keep track of these requirements, we use a partial function t belonging to some set $\mathcal{T}$, which we define below. In particular, the partial function t maps some of the above task symbols to $\mathbb{N}$. In fact, these requirements determine to a large extent how tasks are treated and are mainly responsible that the oracle satisfies the desired properties. To add a requirement in the construction, we can extend the function t.

Define $\mathcal{T}$ as the set of all partial functions t mapping $\tau^1_{a,b}$, $\tau^2_{i,j}$, τ^3_i, $\tau^4_{i,j}$, $i \neq j$, $a \neq b$ to $\mathbb{N}$, and $\mathrm{dom}(t)$ is finite, and t is injective on its support.

To now link the maintenance of the requirements with the oracle construction, we introduce the notion of *validity*. A partial oracle $w \in \Sigma^*$ is called *t-valid* for $t \in \mathcal{T}$ if it satisfies the following properties:

V1 If $t(\tau^1_{a,b}) = 0$, then there exists an x such that $M_a^w(x)$, $M_b^w(x)$ both definitely accept or both definitely reject.
(Meaning: if $t(\tau^1_{a,b}) = 0$, then for every extension of the oracle, M_a, M_b do not accept complementary.)

V2 If $0 < t(\tau^1_{a,b}) \leq c(a,b,x) < |w|$, then $M_a^w(x)$ is definite. Additionally, the computation $M_a^w(x)$ accepts when $c(a,b,x) \in w$, and rejects when $c(a,b,x) \notin w$. When the above conditions are met by $c(a,b,x)$ we sometimes refer to these code words as *mandatory* code words with respect to some t-valid partial oracle w. Note that when the previous conditions are not met ($\tau^1_{a,b} \notin \mathrm{dom}(t)$ or $t(\tau^1_{a,b}) = 0$ or $t(\tau^1_{a,b}) > c(a,b,x)$) then the code word $c(a,b,x)$ may be a member of oracle w, independent of M_a, M_b.
(Meaning: if $t(\tau^1_{a,b}) > 0$, then from $t(\tau^1_{a,b})$ on, we encode $L(M_a)$ into the oracle. That is, $L(M_a^O) = (\{x \mid c(a,b,x) \in O\} \cup \text{some finite set}) \in \mathrm{P}^O$.)

V3 If $t(\tau^2_{i,j}) = 0$, then there exists an x such that $M_i^w(x)$, $M_j^w(x)$ both definitely accept.
(Meaning: if $t(\tau^2_{i,j}) = 0$, then for every extension of the oracle, $(L(M_i), L(M_j)) \notin \mathrm{DisjNP}$.)

V4 If $t(\tau^2_{i,j}) = m > 0$, then for every $n \in H_m$ it holds that $|\Sigma^n \cap w| \leq 1$.
(Meaning: if $t(\tau^2_{i,j}) = m > 0$, then ensure that $(A_m, B_m) \in \mathrm{DisjUP}$ relative to the final oracle.)

V5 If $t(\tau^3_i) = 0$, then there exists an x such that $M_i^w(x)$ is definite and accepts on two different paths.
(Meaning: if $t(\tau^3_i) = 0$, then for every extension of the oracle, $L(M_i) \notin \mathrm{UP}$.)

V6 If $t(\tau^3_i) = m > 0$, then for every $n \in H_m$ it holds that $|\Sigma^n \cap w| \leq 1$.
(Meaning: if $t(\tau^3_i) = m > 0$, then ensure that $C_m \in \mathrm{UP}$ relative to the final oracle.)

V7 If $t(\tau^4_{i,j}) = 0$, then there exists an x such that $M_i^w(x)$, $M_j^w(x)$ both definitely reject.
(Meaning: if $t(\tau^4_{i,j}) = 0$, then for every extension of the oracle, $(\overline{L(M_i)}, \overline{L(M_j)}) \notin \mathrm{DisjCoNP}$.)

V8 If $t(\tau^4_{i,j}) = m > 0$, then for every $n \in H_m$ it holds that all words in $\Sigma^n \cap w$ have pairwise different parity. If additionally w is defined for all words of length n, then $|\Sigma^n \cap w| > 0$. (Meaning: if $t(\tau^4_{i,j}) = m > 0$, then ensure that $(D_m, E_m) \in \mathrm{DisjCoUP}$ relative to the final oracle.)

Intuitively, a t-valid oracle is a possibly partial oracle which has the desired properties (i)–(iv) "partially satisfied". This notion of validness helps in the oracle construction, since our oracle is defined inductively, the induction step deals with a partial oracle and therefore t-validity fits great as part of an induction hypothesis which states that the partial oracle is constructed properly so far. Observe that V4, V6, V8 do not (pairwise) contradict each other, since t is injective on its support and all $H_1, H_2, \dots$ are pairwise disjoint, by Observation 2(i). Also observe that V2 and V4 (resp., V2 and V6, V2 and V8) do not contradict each other, as $c(\cdot,\cdot,\cdot)$ has odd length, but all n in all H_m are even by Observation 2(i).

Oracle Construction. Let T be a countable enumeration of

$$\begin{aligned}\{\tau^1_{a,b} \mid a,b \in \mathbb{N}, a \neq b\} &\cup \{\tau^2_{i,j} \mid i,j \in \mathbb{N}, i \neq j\} \cup \{\tau^2_{i,j,r} \mid i,j,r \in \mathbb{N}, i \neq j\} \\ &\cup \{\tau^3_{i} \mid i \in \mathbb{N}\} \qquad\quad \cup \{\tau^3_{i,r} \mid i,r \in \mathbb{N}\} \\ &\cup \{\tau^4_{i,j} \mid i,j \in \mathbb{N}, i \neq j\} \cup \{\tau^4_{i,j,r} \mid i,j,r \in \mathbb{N}, i \neq j\}\end{aligned}$$

with the property that $\tau^2_{i,j}$ appears earlier than $\tau^2_{i,j,r}$, τ^3_i appears earlier than $\tau^3_{i,r}$, $\tau^4_{i,j}$ earlier than $\tau^4_{i,j,r}$.

We inductively define an infinite sequence $\{(w_s, t_s)\}_{s \in \mathbb{N}}$, where the s-th term of the sequence is a pair (w_s, t_s) of a partial oracle and a function in $\mathcal{T}$. We call the s-th term the *stage* s. We ensure that for all $s \in \mathbb{N}$, w_s is a t_s-valid partial oracle.

In each stage, we treat the smallest task in the order specified by T, and after treating a task we remove it and possibly other higher tasks from T. In the next stage, we continue with the next task not already removed from T. (In every stage, there always exists a task not already removed, as we never remove *all* remaining tasks from T in any stage.)

We start with the nowhere defined function $t_0 \in \mathcal{T}$ and the t_0-valid oracle $w_0 := \varepsilon$ as 0-th stage. Then we begin treating the tasks.

Thus, for stage $s > 0$, we have that $w_0, w_1, \dots, w_{s-1}$ and $t_0, t_1, \dots, t_{s-1}$ are defined. With this, we define the s-th stage (w_s, t_s) such that (a) $w_{s-1} \sqsubsetneq w_s$, and $t_s \in \mathcal{T}$ is a (not necessarily strict) extension of t_{s-1}, and (b) w_s is t_s-valid, and (c) the earliest task τ still in T is treated and removed in some way.

So for each task we strictly extend the oracle and are allowed to add more requirements, by extending the valid function, that have to be maintained in the further construction. Finally, we choose $O := \bigcup_{i \in \mathbb{N}} w_i$. (Note that O is totally defined since in each step we strictly extend the oracle.) Also, every task in T is assigned some stage s where it was treated (or removed from T).

We now define stage $s > 0$, which starts with some $t_{s-1} \in \mathcal{T}$ and a t_{s-1}-valid oracle w_{s-1} and treats the first task that still is in T choosing an extension $t_s \in \mathcal{T}$ of t_{s-1} and a t_s-valid $w_s \sqsupsetneq w_{s-1}$. Let us recall that each task is immediately deleted from T after it is treated. There are seven cases depending on the form of the task that is treated in stage s:

- Task $\tau^1_{a,b}$: Let $t' := t_{s-1} \cup \{\tau^1_{a,b} \mapsto 0\}$. If there exists a t'-valid $v \sqsupsetneq w_{s-1}$, then assign $t_s := t'$ and let $w_s := v$.
  Otherwise, let $t_s := t_{s-1} \cup \{\tau^1_{a,b} \mapsto n\}$ with $n \in \mathbb{N}^+$ sufficiently large such that $n > |w_s|, \max \mathrm{img}(t_{s-1})$. Thus t_s is injective on its support, and w_{s-1} is t_s-valid. Let $w_s := w_{s-1}y$ with $y \in \{0,1\}$ such that w_s is t_s-valid. Lemma 5 shows that such y does indeed exist.

(Meaning: try to ensure that M_a, M_b do not accept complementary, cf. V1. If that is impossible, require that from now on the computations of M_a are encoded into the oracle, cf. V2.)

- Task $\tau^2_{i,j}$: Let $t' := t_{s-1} \cup \{\tau^2_{i,j} \mapsto 0\}$. If there exists t'-valid $v \sqsupsetneq w_{s-1}$, then assign $t_s := t'$ and $w_s := v$. Besides task $\tau^2_{i,j}$, also remove all tasks $\tau^2_{i,j,0}, \tau^2_{i,j,1}, \ldots$ from T.
  Otherwise, let $t_s := t_{s-1} \cup \{\tau^2_{i,j} \mapsto m\}$ with $m \in \mathbb{N}^+$ sufficiently large such that $m \notin \mathrm{img}(t_{s-1})$ and that w_{s-1} defines no word of length $\min H_m$. Thus t_s is injective on its support, and w_{s-1} is t_s-valid. Let $w_s := w_{s-1}y$ with $y \in \{0, 1\}$ such that w_s is t_s-valid. Again, Lemma 5 shows that such y does indeed exist.

  (Meaning: try to ensure that M_i, M_j do not accept disjointly, cf. V3. If that is impossible, choose a sufficiently large "fresh" m and require for the further construction that $(A_m, B_m) \in \mathrm{DisjUP}$ (cf. V4). The treatment of the tasks $\tau^2_{i,j,0}, \tau^2_{i,j,1}, \ldots$ defined below makes sure that (A_m, B_m) cannot be reduced to $(L(M_i), L(M_j))$.)
- Task τ^3_i: Defined symmetrically to task $\tau^2_{i,j}$. Let $t' := t_{s-1} \cup \{\tau^3_i \mapsto 0\}$. If there exists t'-valid $v \sqsupsetneq w_{s-1}$, then assign $t_s := t'$ and $w_s := v$. Besides task τ^3_i, also remove all tasks $\tau^3_{i,0}, \tau^3_{i,1}, \ldots$ from T.
  Otherwise, let $t_s := t_{s-1} \cup \{\tau^3_i \mapsto m\}$ with $m \in \mathbb{N}^+$ sufficiently large such that $m \notin \mathrm{img}(t_{s-1})$ and that w_{s-1} defines no word of length $\min H_m$. Thus t_s is injective on its support, and w_{s-1} is t_s-valid. Let $w_s := w_{s-1}y$ with $y \in \{0, 1\}$ such that w_s is t_s-valid. Again, Lemma 5 shows that such y does indeed exist.

  (Meaning: try to ensure that M_i does accept on two different paths, cf. V5. If that is impossible, choose a sufficiently large "fresh" m and require for the further construction that $C_m \in \mathrm{UP}$ (cf. V6). The treatment of the tasks $\tau^3_{i,0}, \tau^3_{i,1}, \ldots$ defined below makes sure that C_m cannot be reduced to $L(M_i)$.)
- Task $\tau^4_{i,j}$: Defined symmetrically to task $\tau^2_{i,j}$. (Meaning: try to ensure that M_i, M_j do not reject disjointly, cf. V7. If that is impossible, choose a sufficiently large "fresh" m and require for the further construction that $(D_m, E_m) \in \mathrm{DisjCoUP}$ (cf. V8). The treatment of the tasks $\tau^4_{i,j,0}, \tau^4_{i,j,1}, \ldots$ defined below makes sure that (D_m, E_m) cannot be reduced to $(\overline{L(M_i)}, \overline{L(M_j)})$.)
- Task $\tau^2_{i,j,r}$: We have $t_{s-1}(\tau^2_{i,j}) = m \in \mathbb{N}^+$. Let $t_s := t_{s-1}$ and choose a t_s-valid $w_s \sqsupsetneq w_{s-1}$ such that there is some $n \in \mathbb{N}$ and at least one of the following holds:
  - $0^n \in A^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_i(F^{w_s}_r(0^n))$ definitely rejects.
  - $0^n \in B^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_j(F^{w_s}_r(0^n))$ definitely rejects.

  In Theorem 6 we show that such w_s does exist.

  (Meaning: ensure that F_r does not reduce (A_m, B_m) to $(L(M_i), L(M_j))$.)
- Task $\tau^3_{i,r}$: We have $t_{s-1}(\tau^3_i) = m \in \mathbb{N}^+$. Let $t_s := t_{s-1}$ and choose a t_s-valid $w_s \sqsupsetneq w_{s-1}$ such that there is some $n \in \mathbb{N}$ and at least one of the following holds:
  - $0^n \in C^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_i(F^{w_s}_r(0^n))$ definitely rejects.
  - $0^n \notin C^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_i(F^{w_s}_r(0^n))$ definitely accepts.

  In Theorem 7 we show that such w_s does exist.

  (Meaning: ensure that F_r does not reduce C_m to $L(M_i)$.)
- Task $\tau^4_{i,j,r}$: Defined symmetrically to $\tau^2_{i,j,r}$. Choose a t_s-valid $w_s \sqsupsetneq w_{s-1}$ such that for some $n \in \mathbb{N}$, one of the two holds:
  - $0^n \in D^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_i(F^{w_s}_r(0^n))$ definitely accepts.
  - $0^n \in E^v_m$ for all $v \sqsupseteq w_s$ and $M^{w_s}_j(F^{w_s}_r(0^n))$ definitely accepts.

  In Theorem 8 we show that such w_s does exist.

  (Meaning: ensure that F_r does not reduce (D_m, E_m) to $(\overline{L(M_i)}, \overline{L(M_j)})$.)

Observe that t_s is always defined to be in $\mathcal{T}$. Remember that the treated task is immediately deleted from T. This completes the definition of stage s, and thus, the entire sequence $\{(w_s, t_s)\}_{s\in\mathbb{N}}$. We now show that this construction is indeed possible. The proofs of the theorems/lemma announced in the definition are either roughly sketched or omitted. For detailed proofs, we refer to the full version of the paper. It is not difficult to see that a valid oracle can be extended by one bit such that it remains valid:

▶ **Lemma 5.** *Let $s \in \mathbb{N}$, $(w_0, t_0), \dots, (w_s, t_s)$ defined, and let $w \in \Sigma^*$ be a t_s-valid oracle with $w \sqsupseteq w_s$, and $z := |w|$. (Think of z as the next word we need to decide its membership to the oracle, i.e., $z \notin w0$ or $z \in w1$.) Then there exists $y \in \{0,1\}$ such that wy is t_s-valid. Specifically:*

- **(i)** *If $z = c(a, b, x)$ and $0 < t_s(\tau^1_{a,b}) \le z$, then $w1$ is t_s-valid if $M^w_a(x)$ accepts (or when $M^w_b(x)$ rejects), and $w0$ is t_s-valid if $M^w_a(x)$ rejects (or when $M^w_b(x)$ accepts). (Meaning: if we are at a position of some mandatory code word, add the word as appropriate for the* NP ∩ coNP*-pair.)*
- **(ii)** *If there exists $\tau = \tau^2_{i,j}$ or $\tau = \tau^3_i$ with $m = t_s(\tau) > 0$ and $n \in H_m$ such that $|z| = n$, $w \cap \Sigma^n \neq \emptyset$, then $w0$ is t_s-valid. (Meaning: if we are on a level n belonging to a* DisjUP*-pair or a* UP*-language, ensure that there is no more than one word on that level.)*
- **(iii)** *If there exists $\tau^4_{i,j}$, $m = t_s(\tau^4_{i,j}) > 0$ and $n \in H_m$ such that $|z| = n$ and there is some other word $x \in w \cap \Sigma^n$ with same parity as z, then $w0$ is t_s-valid. (Meaning: if we are on a level n belonging to a* DisjCoUP*-pair, ensure that on that level, there are no two words with the same parity.)*
- **(iv)** *If there exists $\tau^4_{i,j}$, $m = t_s(\tau^4_{i,j}) > 0$ and $n \in H_m$ such that $|z| = n$, $|z+1| > n$, $w \cap \Sigma^n = \emptyset$, then $w1$ is t_s-valid. (Meaning: if we finalize level n belonging to a* DisjCoUP *witness pair, ensure that there is at least one word on that level.)*
- **(v)** *In all other cases, $w0$ and $w1$ are t_s-valid.*

Lemma 5 shows that the construction is possible for the tasks $\tau^1_{a,b}$, $\tau^2_{i,j}$, τ^3_i and $\tau^4_{i,j}$. Now we show that the construction is possible for $\tau^2_{i,j,r}$, $\tau^3_{i,r}$ and $\tau^4_{i,j,r}$, respectively. We first consider task $\tau^2_{i,j,r}$.

▶ **Theorem 6.** *Let $s \in \mathbb{N}^+$, $(w_0, t_0), \dots, (w_{s-1}, t_{s-1})$ defined. Consider task $\tau^2_{i,j,r}$.*

Suppose that $t_s = t_{s-1}$, $t_s(\tau^2_{i,j}) = m > 0$. Then there exists a t_s-valid $w \sqsupsetneq w_{s-1}$ and $n \in \mathbb{N}$ such that one of the two holds:

- **(i)** *$0^n \in A^v_m$ for all $v \sqsupseteq w$ and $M^w_i(F^w_r(0^n))$ definitely rejects.*
- **(ii)** *$0^n \in B^v_m$ for all $v \sqsupseteq w$ and $M^w_j(F^w_r(0^n))$ definitely rejects.*

Proof sketch. We prove the Theorem by contradiction. Let $\hat{s} < s$ be the stage that treated $\tau^2_{i,j}$ with $t_{\hat{s}} = t_{\hat{s}-1} \cup \{\tau^2_{i,j} \mapsto m\}$ and $m > 0$. We construct a suitable alternative oracle $u' \sqsupsetneq w_{\hat{s}-1}$, which is valid with respect to $t' := t_{\hat{s}-1} \cup \{\tau^2_{i,j} \mapsto 0\}$. Then, by definition, we obtain that u' is one possible t'-valid extension of $w_{\hat{s}-1}$ in stage $\hat{s}$, hence $t_{\hat{s}} = t'$ and $t_{\hat{s}}(\tau^2_{i,j}) = 0$, contradicting $t_s(\tau^2_{i,j}) = m > 0$ in the hypothesis of this Theorem 6.

Assume that (i) and (ii) do not hold. Fix some sufficiently large $n \in H_m$. This is some level that belongs to the witness NP-pair (A_m, B_m). For every $\xi \in \Sigma^n$, one can provisionally keep extending w_{s-1} bitwise with Lemma 5 while inserting precisely the word ξ into level n. Continue extending until a sufficiently long but fixed length n' such that $M_i(F_r(0^n))$ and $M_j(F_r(0^n))$ are definite (relative to any oracle), and call the resulting oracle u_ξ. By construction, this oracle is t_s-valid. By assumption, when $\alpha \in \Sigma^{n-1}0$, then $M_i(F_r(0^n))$ accepts relative to u_α, and when $\beta \in \Sigma^{n-1}1$, then $M_j(F_r(0^n))$ accepts relative to u_β.

We want to maintain these accepting computations relative to an oracle containing, in level n, exactly one $\alpha \in \Sigma^{n-1}0$ and one $\beta \in \Sigma^{n-1}1$. The accepting behavior for any such computation, say $M_i(F_r(0^n))$ relative to u_α, depends on the oracle queries posed on that path. However, this computation might also query certain mandatory code words $c(a,b,x)$, whose memberships depend on further, shorter queries of computations $M_a(x)$, $M_b(x)$. Continuing this recursively, we obtain a set of queries Q^+_α the original computation $M_i(F_r(0^n))$ transitively depends on.

Similarly, for each $\beta \in \Sigma^{n-1}1$ we can define a set of queries Q^+_β an accepting path of the computation $M_j(F_r(0^n))$ relative to u_β depends on. One can verify that Q^+_α and Q^+_β only have polynomially many elements.

The crucial idea that completes the proof is to find suitable $\alpha \in \Sigma^{n-1}0$ and $\beta \in \Sigma^{n-1}1$ such that u_α and u_β agree on the set $Q^+_\alpha \cap Q^+_\beta$. Such a pair of words α and β exists; we skip here the combinatorial argument, but intuitively this assertion holds from the fact that there are exponentially many choices for α, β but for each choice, Q^+_α and Q^+_β create only polynomially many dependencies. With this property, it is possible to construct a $t_{\hat{s}-1}$-valid oracle $u' \sqsupsetneq w_{s-1}$ such that $\alpha, \beta \in u'$ holds, u' and u_α agree on Q^+_α, and symmetric u' and u_β agree on Q^+_β. By assumption and Observation 1(iii), this means that *both* $M_i(F_r(0^n))$ and $M_j(F_r(0^n))$ definitely accept relative to u'. Thus u' is also t'-valid, as desired. ◀

With only slight modifications, one can give the same Theorem concerning tasks $\tau^3_{i,r}$. We omit the specific details.

▶ **Theorem 7.** *Let $s \in \mathbb{N}^+$, $(w_0, t_0), \ldots, (w_{s-1}, t_{s-1})$ defined. Consider task $\tau^3_{i,r}$.*

Suppose that $t_s = t_{s-1}$, $t_s(\tau^3_i) = m > 0$. Then there exists a t_s-valid $w \sqsupsetneq w_{s-1}$ and $n \in \mathbb{N}$ such that one of the following holds:

(i) *$0^n \in C^v_m$ for all $v \sqsupseteq w$ and $M^w_i(F^w_r(0^n))$ definitely rejects.*

(ii) *$0^n \notin C^v_m$ for all $v \sqsupseteq w$ and $M^w_i(F^w_r(0^n))$ definitely accepts.*

Lastly, handling task $\tau^4_{i,j,r}$ is also possible. For this we use techniques similar to previous theorems. In this setting, it is in fact possible to explicitly construct a suitable extension for the task. Due to many additional technical details required, we omit the proof of this theorem and refer to the full version of the paper.

▶ **Theorem 8.** *Let $s \in \mathbb{N}^+$, (w_{s-1}, t_{s-1}) defined. Consider task $\tau^4_{i,j,r}$.*

Suppose that $t_s = t_{s-1}$, $t_s(\tau^4_{i,j}) = m > 0$. Then there exists a t_s-valid $w \sqsupsetneq w_{s-1}$ and $n \in \mathbb{N}$ such that one of the following holds:

(i) *$0^n \in D^v_m$ for all $v \sqsupseteq w$ and $M^w_i(F^w_r(0^n))$ definitely accepts.*

(ii) *$0^n \in E^v_m$ for all $v \sqsupseteq w$ and $M^w_j(F^w_r(0^n))$ definitely accepts.*

We have now completed the proofs showing that the oracle construction can be performed as desired. The following theorem confirms the desired properties of $O := \bigcup_{i \in \mathbb{N}} w_i$. Remember that $|w_0| < |w_1| < \ldots$ is unbounded, hence for any z there is a sufficiently large s such that $|w_s| > z$. Also remember that w_s is t_s-valid for all $s \in \mathbb{N}$. Using these facts and the properties V1–V8 of t_s-valid oracles, one can easily state the following result for the final oracle.

▶ **Theorem 9.** *Relative to $O = \bigcup_{i \in \mathbb{N}} w_i$, the following holds:*

(i) $\mathrm{NP} \cap \mathrm{coNP} = \mathrm{P}$*, which implies* $\neg\mathsf{NP} \cap \mathsf{coNP}$.

(ii) *No pair in* DisjNP *is* $\leq^{\mathrm{pp}}_{\mathrm{m}}$*-hard for* DisjUP*, which implies* DisjNP.

(iii) *No language in* UP *is* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-complete for* UP*, i.e.,* UP.

(iv) *No pair in* DisjCoNP *is* $\leq^{\mathrm{pp}}_{\mathrm{m}}$*-hard for* DisjCoUP*, which implies* $\mathsf{DisjCoNP}$.

From Theorem 9 and known relativizable results, we obtain the following additional properties that hold relative to the oracle. See, e.g., the work of Fenner et al. [15] for a definition of the mentioned function classes $\mathrm{NPSV}, \mathrm{NPbV}, \mathrm{NP}k\mathrm{V}, \mathrm{NPMV}$ and their total variants $\mathrm{NPSV_t}, \mathrm{NPbV_t}, \mathrm{NP}k\mathrm{V_t}, \mathrm{NPMV_t}$. Here, NEE means $\mathrm{NTIME}\left(2^{O(2^n)}\right)$.

▶ **Corollary 10.** *The following holds relative to the oracle O constructed in this section.*

(i) $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP} \subsetneq \mathrm{UP} \subsetneq \mathrm{NP}$
(ii) UP*,* NP*,* NE*, and* NEE *are not closed under complement.*
(iii) $\mathrm{UP} \not\subseteq \mathrm{coNP}$
(iv) $\mathrm{NEE} \cap \mathrm{TALLY} \not\subseteq \mathrm{coNEE}$
(v) $\mathrm{NPSV_t} \subseteq \mathrm{PF}$
(vi) $\mathrm{NPbV_t} \not\subseteq_c \mathrm{NPSV_t}$
(vii) $\mathrm{NP}k\mathrm{V_t} \not\subseteq_c \mathrm{NPSV_t}$ *for all* $k \geq 2$
(viii) $\mathrm{NPMV_t} \not\subseteq_c \mathrm{NPSV_t}$
(ix) $\mathrm{NPMV} \not\subseteq_c \mathrm{NPSV}$
(x) $\mathrm{TFNP} \not\subseteq_c \mathrm{PF}$
(xi) $\mathrm{NP} \cap \mathrm{coNP}$ *has* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-complete sets, i.e.,* $\neg\mathsf{NP} \cap \mathsf{coNP}$*.*
(xii) UP *has no* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-complete sets, i.e.,* UP*.*
(xiii) DisjNP *has no* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-complete pairs, i.e.,* DisjNP*.*
(xiv) DisjCoNP *has no* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-complete pairs, i.e.,* $\mathsf{DisjCoNP}$*.*
(xv) *No pair in* DisjNP *is* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hard for* DisjUP*.*
(xvi) *No pair in* DisjCoNP *is* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hard for* DisjCoUP*.*
(xvii) *There are no p-optimal proof systems for* TAUT*, i.e.,* CON*.*
(xviii) *There are no optimal proof systems for* TAUT*.*
(xix) *There are no p-optimal proof systems for* SAT*, i.e.,* SAT*.*
(xx) TFNP *has no* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-complete problems, i.e.,* TFNP*.*
(xxi) $\mathrm{NPMV_t}$ *has no* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-complete functions.*
(xxii) NP *and* coNP *do not have the shrinking property. [7, 16]*
(xxiii) NP *and* coNP *do not have the separation property. [16]*
(xxiv) DisjNP *and* DisjCoNP *contain* P*-inseparable pairs.*

References

1 O. Beyersdorff. Representable disjoint NP-pairs. In *Proceedings 24th International Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 3328 of *Lecture Notes in Computer Science*, pages 122–134. Springer, 2004. doi:10.1007/978-3-540-30538-5_11.

2 O. Beyersdorff. Disjoint NP-pairs from propositional proof systems. In *Proceedings of Third International Conference on Theory and Applications of Models of Computation*, volume 3959 of *Lecture Notes in Computer Science*, pages 236–247. Springer, 2006. doi:10.18452/15520.

3 O. Beyersdorff. Classes of representable disjoint NP-pairs. *Theoretical Computer Science*, 377(1-3):93–109, 2007. doi:10.1016/j.tcs.2007.02.005.

4 O. Beyersdorff. The deduction theorem for strong propositional proof systems. *Theory of Computing Systems*, 47(1):162–178, 2010. doi:10.1007/s00224-008-9146-6.

5 O. Beyersdorff, J. Köbler, and J. Messner. Nondeterministic functions and the existence of optimal proof systems. *Theoretical Computer Science*, 410(38-40):3839–3855, 2009. doi:10.1016/j.tcs.2009.05.021.

6 O. Beyersdorff and Z. Sadowski. Do there exist complete sets for promise classes? *Mathematical Logic Quarterly*, 57(6):535–550, 2011. doi:10.1002/malq.201010021.

7 A. Blass and Y. Gurevich. Equivalence relations, invariants, and normal forms. *SIAM Journal on Computing*, 13(4):682–689, 1984. doi:10.1137/0213042.

8 S. Cook and R. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44:36–50, 1979. doi:10.2307/2273702.

9 T. Dose. *Balance Problems for Integer Circuits and Separations of Relativized Conjectures on Incompleteness in Promise Classes*. PhD thesis, Fakultät für Mathematik und Informatik, Universität Würzburg, 2020. doi:10.25972/OPUS-22220.

10 T. Dose. Further oracles separating conjectures about incompleteness in the finite domain. *Theoretical Computer Science*, 847:76–94, 2020. doi:10.1016/j.tcs.2020.09.040.

11 T. Dose. An oracle separating conjectures about incompleteness in the finite domain. *Theoretical Computer Science*, 809:466–481, 2020. doi:10.1016/j.tcs.2020.01.003.

12 T. Dose and C. Glaßer. NP-completeness, proof systems, and disjoint NP-pairs. In C. Paul and M. Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 9:1–9:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.STACS.2020.9.

13 J. Edmonds. Minimum partition of a matroid into independent subsets. *Journal of Research of the National Bureau of Standards*, 69B:67–72, 1965. doi:10.6028/JRES.069B.004.

14 S. Fenner, L. Fortnow, A. Naik, and J. Rogers. On inverting onto functions. In *Proceedings 11th Conference on Computational Complexity*, pages 213–223. IEEE Computer Society Press, 1996.

15 S. A. Fenner, L. Fortnow, A. V. Naik, and J. D. Rogers. Inverting onto functions. *Information and Computation*, 186(1):90–103, 2003. doi:10.1016/S0890-5401(03)00119-6.

16 C. Glaßer, C. Reitwießner, and V. L. Selivanov. The shrinking property for NP and coNP. *Theoretical Computer Science*, 412(8-10):853–864, 2011. doi:10.1016/j.tcs.2010.11.035.

17 C. Glaßer, A. L. Selman, and S. Sengupta. Reductions between disjoint NP-pairs. *Information and Computation*, 200:247–267, 2005. doi:10.1016/j.ic.2005.03.003.

18 C. Glaßer, A. L. Selman, S. Sengupta, and L. Zhang. Disjoint NP-pairs. *SIAM Journal on Computing*, 33(6):1369–1416, 2004. doi:10.1137/S0097539703425848.

19 C. Glaßer, A. L. Selman, and L. Zhang. Canonical disjoint NP-pairs of propositional proof systems. *Theoretical Computer Science*, 370:60–73, 2007. doi:10.1007/11549345_35.

20 C. Glaßer, A. L. Selman, and L. Zhang. The informational content of canonical disjoint NP-pairs. *International Journal of Foundations of Computer Science*, 20(3):501–522, 2009. doi:10.1007/978-3-540-73545-8_31.

21 J. Grollmann and A. L. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988. doi:10.1137/0217018.

22 J. Hartmanis and L. A. Hemachandra. Complexity classes without machines: On complete languages for UP. *Theoretical Computer Science*, 58:129–142, 1988. doi:10.1016/0304-3975(88)90022-9.

23 Kannan, 1979. Sipser [36] cites an unpublished work by Kannan for asking if there is a set complete for NP ∩ coNP.

24 E. Khaniki. New relations and separations of conjectures about incompleteness in the finite domain. *ArXiv preprint*, 2019. arXiv:1904.01362.

25 J. Köbler, J. Messner, and J. Torán. Optimal proof systems imply complete sets for promise classes. *Information and Computation*, 184(1):71–92, 2003. doi:10.1016/S0890-5401(03)00058-0.

26 J. Krajícek and P. Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *Journal of Symbolic Logic*, 54:1063–1079, 1989. doi:10.2307/2274765.

27 N. Megiddo and C. H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991. doi:10.1016/0304-3975(91)90200-L.

28 J. Messner. *On the Simulation Order of Proof Systems*. PhD thesis, Universität Ulm, 2000.

29 C. H. Papadimitriou. On the complexity of integer programming. *Journal of the ACM*, 28(4):765–768, 1981. doi:10.1145/322276.322287.

30 P. Pudlák. On the lengths of proofs of consistency. In *Collegium Logicum*, pages 65–86. Springer Vienna, 1996. doi:10.1016/S0049-237X(08)70462-2.

31 P. Pudlák. On reducibility and symmetry of disjoint NP pairs. *Theoretical Computer Science*, 295:323–339, 2003. doi:10.1016/S0304-3975(02)00411-5.

32 P. Pudlák. On some problems in proof complexity. In O. Beyersdorff, E. A. Hirsch, J. Krajícek, and R. Santhanam, editors, *Optimal algorithms and proofs (Dagstuhl Seminar 14421)*, volume 4, pages 63–63, Dagstuhl, Germany, 2014. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/DagRep.4.10.51.

33 P. Pudlák. Incompleteness in the finite domain. *The Bulletin of Symbolic Logic*, 23(4):405–441, 2017. doi:10.1017/bsl.2017.32.

34 A. Razborov. On provably disjoint NP-pairs. *BRICS Report Series*, 36, 1994. doi:10.7146/brics.v1i36.21607.

35 A. L. Selman. Promise problems complete for complexity classes. *Information and Computation*, 78:87–98, 1988. doi:10.1016/0890-5401(88)90030-2.

36 M. Sipser. On relativization and the existence of complete sets. In *Proceedings 9th ICALP*, volume 140 of *Lecture Notes in Computer Science*, pages 523–531. Springer Verlag, 1982. doi:10.1007/BFb0012797.

37 L. G. Valiant. Relative complexity of checking and evaluation. *Information Processing Letters*, 5:20–23, 1976. doi:10.1016/0020-0190(76)90097-1.

38 O. V. Verbitskii. Optimal algorithms for coNP-sets and the EXP=?NEXP problem. *Mathematical notes of the Academy of Sciences of the USSR*, 50(2):796–801, 1991. doi:10.1007/BF01157564.