Complete Problems for Promise Classes by Optimal Proof Systems for Test Sets

Johannes Köbler Jochen Messner

Abteilung Theoretische Informatik
Universität Ulm
89069 Ulm, Germany
{koebler,messner}@informatik.uni-ulm.de

Abstract

We present a uniform approach to investigate the relationship between the existence of complete sets for promise classes and the existence of (p-)optimal proof systems for certain languages. Central to our approach is the notion of a test set *which can be used to verify that a given nondeterministic polynomial-time machine obeys the promise on a given input. Basically, we show that a promise class $\mathcal{C}$ has a many-one complete language if and only if there is a test set for $\mathcal{C}$ which has a p-optimal proof system. As an application we are able to improve earlier results. For example, we show that $\mathcal{NP} \cap$ co-$\mathcal{NP}$ has a many-one complete language, provided that the set* TAUT *of all valid boolean formulas as well as the set* SAT *of all satisfiable boolean formulas have p-optimal proof systems. We also apply the result to other classes and show, for example, that the probabilistic complexity classes $\mathcal{BPP}$, $\mathcal{RP}$, and $\mathcal{ZPP}$ have many-one complete languages, provided that the set* TAUT_2 *of all valid Π_2-formulas in quantified propositional logic has a p-optimal proof system. Finally it is shown that already a collapse of tally sets at the double exponential time level implies the existence of a (p-)optimal proof system for* TAUT.

1 Introduction

In this paper we continue a line of research started independently in [Sad97] and [MT98]. The results in those papers reveal a connection between the existence of optimal or p-optimal proof systems for certain logical languages and the existence of many-one complete languages for promise classes. A *proof system* for a language L is just a polynomial-time computable function h whose range $\{h(x) \mid x \in \{0,1\}^*\}$ is exactly L. This notion was introduced by Cook and Reckhow in [CR79].

The notions of simulation and p-simulation are useful to compare the strength of different proof systems. A proof system h for L is said to *simulate* a proof system g for L if there is a polynomial p such that for any $x \in L$ and each g-proof w for x (meaning that $g(w) = x$) there is an h-proof w' for x of length at most $p(|w|)$. A proof system h is said to *p-simulate* g if there is a polynomial-time computable function $t \in \mathcal{FP}$ translating g-proofs to h-proofs, i.e., $g(w) = h(t(w))$ for all w. A proof system is called *(p-)optimal* for L when it (p-)simulates any proof system for L.

In [Sad97] it was shown that the existence of a p-optimal proof system for the tautologies of the quantified propositional logic (QBF) implies the existence of a many-one complete set for $\mathcal{NP} \cap$ co-$\mathcal{NP}$. The consequences of the existence of p-optimal and optimal proof systems for the tautologies of propositional logic (TAUT) were investigated in [MT98]. It was shown that a many-one complete language for $\mathcal{UP}$ exists in case there is a p-optimal proof system for TAUT. A similar result was obtained for $\mathcal{NP} \cap$ *Sparse* under the assumption that there is an optimal proof system for TAUT.

We formalize the intuitive connection between the existence of (p-)optimal proof systems and the existence of many-one complete sets by introducing the concept of test sets. Roughly speaking, a test set allows us to verify that a given nondeterministic polynomial-time machine behaves well (i.e., in accordance with the promise) on a given x. This concept allows us to formalize and even improve the approach given in [Sad97, MT98]. We state two master theorems which are in turn applied to obtain sufficient conditions for the existence of complete problems for the classes $\mathcal{BPP}$, $\mathcal{RP}$, $\mathcal{ZPP}$, $\mathcal{AM}$ and $\mathcal{MA}$. Furthermore, we improve the main result of [Sad97] by showing that already the existence of p-optimal proof systems for TAUT and for SAT suffices to obtain a complete problem for $\mathcal{NP} \cap$ co-$\mathcal{NP}$.

In [Raz94] some observations were made which imply that the existence of an optimal proof system for TAUT implies a complete pair for the class of disjoint $\mathcal{NP}$-pairs. A pair (A, B) of $\mathcal{NP}$-languages belongs to this class when $A \cap B = \emptyset$. However, in [Raz94] a somewhat weak form

of many-one reducibility is used which is only concerned with inputs from $A \cup B$. Formally, in [Raz94] a pair (A, B) is said to many-one reduce to a pair (C, D) if for some $f \in \mathcal{FP}$, $f(A) \subseteq C$ and $f(B) \subseteq D$. By generalizing our approach to function classes we can use the assumption that TAUT has an optimal proof system to conclude that the class of disjoint $\mathcal{NP}$-pairs has a complete pair with respect to the following stronger notion of many-one reducibility: (A, B) strongly many-one reduces to (C, D) if for some $f \in \mathcal{FP}$, $f^{-1}(C) = A$ and $f^{-1}(D) = B$.

Finally we improve a sufficient condition given in [MT98] for the existence of (p-)optimal proof systems for TAUT by showing that a collapse of tally sets in double exponential time classes suffices. We note that already in [Pud86, KP89] it was shown that the collapse of exponential time classes implies the existence of (p-)optimal proof systems for TAUT.

2 Preliminaries

Let QBF be the set of all valid quantified boolean formulas

$$(Q_1 x_1) \cdots (Q_n x_n)\, F(x_1, \ldots, x_n),$$

where $F(x_1, \ldots, x_n)$ is a boolean formula over the variables $x_1, \ldots, x_n$ and each Q_i is either "$\exists$" or "$\forall$". By TAUT_k we denote the set of all QBF formulas with at most $k-1$ quantifier alternations (a "$\exists$" followed by "$\forall$," or a "$\forall$" followed by "$\exists$") starting with a "$\forall$" quantifier. Similarly, SAT_k denotes the set of all QBF formulas with at most $k-1$ quantifier alternations starting with a "$\exists$" quantifier. As usual, in the case of $k = 1$ we omit the index and simply write SAT for SAT_1 and TAUT for TAUT_1.

We assume some familiarity with complexity theory and refer the reader to [BDG95, BDG90] for standard notions and for the definition of complexity classes. A language A many-one reduces to a language B (in symbols: $A \leq_m^p B$) if there is a polynomial-time computable function f (in symbols: $f \in \mathcal{FP}$) such that for all strings x, $x \in A$ if and only if $f(x) \in B$.

We use Turing machines as our basic computational model. In particular, we consider clocked nondeterministic polynomial-time Turing machines (NPTM for short) and clocked deterministic polynomial-time Turing transducers (PTM for short). To represent those machines we use an encoding which allows us to obtain from the machine code N easily a polynomial p_N that bounds the running time of the machine (in Section 4 we will consider some further restrictions on the encodings of NPTMs and PTMs). In the sequel, we will not distinguish between a machine and its code.

We consider only languages over the alphabet $\Sigma = \{0, 1\}$ (this means that problem instance as, e.g., Boolean formulas have to be suitably encoded). By Σ^* we denote the set of all binary strings, and by $\Sigma^{\leq n}$, for an $n \geq 0$, we denote the set $\{w \in \Sigma^* \mid |w| \leq n\}$ of strings of length at most n. A set T is called *tally* (in symbols: $T \in$ *Tally*) if T is a subset of $\{0^n \mid n \geq 0\}$; a set S is called *sparse* (in symbols: $T \in$ *Sparse*) if there is a polynomial $p(n)$ that bounds the cardinality of $S \cap \Sigma^{\leq n}$. Finally, $\langle \cdot, \cdots, \cdot \rangle$ denotes some standard polynomial-time computable tupling function.

3 Languages with optimal proof systems

In this section we investigate closure properties of the class of all languages which have a (p-)optimal proof system. In fact, we show that this class is closed under $\leq_m^p$ and under join which, by [Bor94], implies that it is a promise class in the classical sense of the tree-structure or of the leaf-language approach.

All observations made in this section refer to the notions of optimality and p-optimality interchangeably. We only give the proofs for the p-optimal case since they can easily be adapted to the case of optimality.

Lemma 1 *If A has a (p-)optimal proof system and if B is a nonempty set that many-one reduces to A, then B has a (p-)optimal proof system, too.*

Proof. Let h be a p-optimal proof system for A and let B many-one reduce to A via f. Then h' defined by

$$h'(\langle x, w \rangle) = \begin{cases} x, & h(w) = f(x), \\ x_0, & \text{otherwise} \end{cases}$$

is certainly a proof system for B (here, x_0 is a fixed element in B). To show that h' is p-optimal, let g' be a proof system for B. It is clear that there is a proof system g for A with $g(1w) = f(g'(w))$. Since h is p-optimal, there is a function $t \in \mathcal{FP}$ translating g-proofs to h-proofs implying that

$$h(t(1w)) = g(1w) = f(g'(w)).$$

This implies

$$h'(\langle g'(w), t(1w) \rangle) = g'(w).$$

Hence, h' p-simulates g'. ∎

The *join* $A \oplus B$ of two languages is given by $0A \cup 1B$. It is a least upper bound for A and B with respect to the $\leq_m^p$-ordering. The *direct product* of A and B is given by $A \times B = \{\langle a, b \rangle \mid a \in A, b \in B\}$. Clearly, if a class is closed under $\leq_m^p$, then closure under intersection implies closure under direct product, as follows from the equality $A \times B = (A \times \Sigma^*) \cap (\Sigma^* \times B)$. Closure under direct product in turn implies closure under join, as $A \oplus B \leq_m^p A \times B$ for nonempty A and B.

Lemma 2 *If A and B have (p-)optimal proof systems, then so have $A \times B$, and $A \oplus B$. If additionally $A \cap B \neq \emptyset$ then also $A \cap B$ has a (p-)optimal proof system.*

Proof. As observed above it suffices to consider the case of nonempty intersection. Let h_1 and h_2 be p-optimal proof systems for A and B, respectively. Let $y \in A \cap B$ be some fixed element. A p-optimal proof system for $A \cap B$ is given by the $\mathcal{FP}$ function

$$h : w \mapsto \begin{cases} x & \text{if } w = \langle u, v \rangle, \text{ and } x = h_1(u) = h_2(v), \\ y & \text{otherwise.} \end{cases}$$

Clearly, h is a proof system for $A \cap B$. To show that h is p-optimal, let f be some proof system for $A \cap B$. By setting $f_i(1w) = f(w)$ and $f_i(0w) = h_i(w)$, $i = 1, 2$, f can be extended to proof systems f_1 and f_2 for A and B, respectively. Let $t_i \in \mathcal{FP}$ be a function translating f_i-proofs to h_i-proofs. Then the function $t(w) = \langle t_1(1w), t_2(1w) \rangle$ translates f-proofs to h-proofs, i.e., $h(t(w)) = f(w)$, as $f(w) = h_1(t_1(1w)) = h_2(t_2(1w))$. ∎

The proof of the next lemma is straightforward.

Lemma 3

- *Any nonempty set in $\mathcal{P}$ has a p-optimal proof system.*
- *Any nonempty set in $\mathcal{NP}$ has an optimal proof system.*

It is an open question whether sets with a (p-)optimal proof system exist outside of $\mathcal{NP}$ (respectively $\mathcal{P}$).

4 Complete sets for promise classes

In this section we provide a quite general approach to promise classes. In the classical leaf-language or tree-structure approach [BCS92, Ver93, Bor94], promises are restricted to be predicates on computation trees (respectively leaf strings) of nondeterministic polynomial-time Turing machines. However it does not seem to be an easy task to define a natural promise for the $\leq_m^p$-closure of, for example, $\mathcal{NP} \cap$ *Sparse* (in spite of the fact that results in [Bor94] show that it is actually a promise class, as it is closed under $\leq_m^p$ and under join).

In order to state our results as general as possible, we allow promises to be formulated in a quite unrestricted way. In this paper, a *promise* R is described by a predicate on the set of all pairs consisting of an NPTM N and an input string x, i.e., $R(N, x)$ means that N obeys promise R on input x. We call N an *R-machine* if N obeys R on any input x. In the sequel, we will only allow promises R for which at least one R-machine exists. The *acceptance criterion* is also a binary predicate Q on NPTMs and strings. The language accepted by the NPTM N (when applying the acceptance criterion Q) is given by

$$L_Q(N) = \{x \in \Sigma^* \mid Q(N, x)\}.$$

Finally, for a promise predicate R and an acceptance criterion Q, we define the promise class

$$\mathcal{C}_{Q,R} = \{L_Q(N) \mid N \text{ is an } R\text{-machine}\}$$

and call (Q, R) a *defining pair* for $\mathcal{C}_{Q,R}$. Notice that a promise class $\mathcal{C}_{Q,R}$ cannot be empty as we assume that some R-machine exists.

In order to obtain completeness results, this setting is still too unrestricted. Therefore, we often restrict our consideration to pairs (Q, R) that fulfill the following two conditions A1 and A2. Basically, A1 demands the existence of a universal NPTM (with respect to Q and R) and A2 requires that $\mathcal{C}_{Q,R}$ is closed under many-one reducibility (in a constructive way).

A1: There is an NPTM U such that the following two conditions hold for all NPTMs N, all $x \in \Sigma^*$, and all $s \geq \text{time}_N(x)$.

1. $Q(N, x) \iff Q(U, \langle N, x, 0^s \rangle)$.
2. $R(N, x) \implies R(U, \langle N, x, 0^s \rangle)$.

A2: There is a binary operation $\circ$ mapping an NPTM N and a polynomial-time transducer M to an NPTM $N \circ M$ such that the following three conditions hold for all NPTMs N, all PTMs M, and all $x \in \Sigma^*$ (f_M denotes the function computed by M).

1. $Q(N, f_M(x)) \iff Q(N \circ M, x)$.
2. $R(N, f_M(x)) \implies R(N \circ M, x)$.
3. The set $\{N \circ M' \mid M'$ is a polynomial-time transducer $\}$ belongs to $\mathcal{P}$.

It is easy to verify that both the conditions A1 and A2 hold in the leaf-language as well as in the tree-structure approach. Notice that in the present context, A2 is not really a restriction. In fact, if one wants to examine whether a class $\mathcal{C}_{Q,R}$ has a complete language, then it suffices to study the $\leq_m^p$-closure of $\mathcal{C}_{Q,R}$. If a pair (Q, R) does not fulfill property A2, then it can be replaced by a defining pair (Q', R') for the $\leq_m^p$-closure of $\mathcal{C}_{Q,R}$ which fulfills A2. Likewise, it is easily seen that for any promise class $\mathcal{C}$ which is closed under $\leq_m^p$ there is a defining pair which fulfills A2.

Using this notion, promise classes like $\mathcal{NP} \cap$ co-$\mathcal{NP}$, $\mathcal{UP}$, $\mathcal{BPP}$, $\mathcal{RP}$, $\mathcal{ZPP}$, $\mathcal{AM}$ and $\mathcal{MA}$ can be characterized in a natural way by defining corresponding pairs (Q, R) which fulfill A1 and A2.

Next we introduce the concept of a test set which is central to our approach.

Definition 4 *Let $\mathcal{C}$ be a promise class, and let (Q, R) be a defining pair for $\mathcal{C}$. Then a set $T \subseteq \Sigma^*$ is called a (Q, R)-*test set for $\mathcal{C}$ *if the following two conditions are fulfilled.*

- *If $\langle N, x, 0^s \rangle \in T$, then s bounds the running time of N on input x and $R(N, x)$ holds.*
- *For any $L \in \mathcal{C}$ there is an NPTM N that accepts L, i.e., $L_Q(N) = L$, and that passes test T, i.e., there is a polynomial p such that for all inputs $x \in \Sigma^*$, $\langle N, x, 0^{p(|x|)} \rangle \in T$.*

Thus, any element $\langle N, x, 0^s \rangle$ belonging to a (Q, R)-test set T serves as an assertion that N behaves well (according to R) on input x. Very informally, the intuitive idea behind the notion of a test set T is that we can obtain a complete language for $\mathcal{C}_{Q,R}$, provided that we can enable an R-machine to decide T (see the proofs of Theorems 7 and 9 for details). In order to make this intuition work we additionally need the following notion.

Definition 5 *Let $\mathcal{A}$ be a class of languages and (Q, R) be a defining pair for a promise class $\mathcal{C}$. We say that $\mathcal{A}$-*assertions are useful for (Q, R)*, if for any language $B \in \mathcal{A}$ and any NPTM N the following holds: if N obeys R for any $x \in B$ then there is a language $C \in \mathcal{C}$ such that*

$$C \cap B = L_Q(N) \cap B.$$

Occasionally, when it is clear from the context which defining pair (Q, R) we associate with a promise class $\mathcal{C}$ then we just say that *$\mathcal{A}$-assertions are useful for $\mathcal{C}$*; similarly, we sometimes call a (Q, R)-test set simply a *test set for $\mathcal{C}$*.

We next show that $\mathcal{P}$-assertion are useful for a promise class, provided that it is closed under $\leq_m^p$-reducibility.

Lemma 6 *If a defining pair (Q, R) fulfills A2 then $\mathcal{P}$-assertions are useful for (Q, R).*

Proof. Let $B \in \mathcal{P}$ and let N be an NPTM such that $R(N, x)$ holds for all $x \in B$. We can assume that B is nonempty (otherwise the statement is true as we assume that $\mathcal{C}_{Q,R}$ is nonempty). Let M be a PTM that computes the $\mathcal{FP}$ function f defined as $f(x) = x$, if $x \in B$, and $f(x) = y$ otherwise, where y is a fixed string in B. Since $R(N, f(x))$ holds for all x, we conclude by assumption A2.2 that $N' = N \circ M$ is an R-machine, implying that $C = L_Q(N') \in \mathcal{C}_{Q,R}$. By the definition of f and by A2.1 it follows that for all $x \in B$, $Q(N, x) \iff Q(N', x)$. Hence, $C \cap B = L_Q(N) \cap B$. ∎

Now we are ready to prove our main theorem.

Theorem 7 *Let $\mathcal{C}$ be a promise class and let (Q, R) be a defining pair for $\mathcal{C}$ which fulfills A1 and A2. Then the following conditions are equivalent.*

1. *$\mathcal{C}$ has a (Q, R)-test set with a p-optimal proof system.*
2. *$\mathcal{C}$ has a many-one complete set.*
3. *There is a set $S \in \mathcal{P}$ that contains only (encodings of) R-machines and for which $\mathcal{C} = \{L_Q(N) \mid N \in S\}$.*
4. *$\mathcal{C}$ has a (Q, R)-test set in $\mathcal{P}$.*

Proof. 1 $\Longrightarrow$ 2. Let T be a (Q, R)-test set for $\mathcal{C}$ which has a p-optimal proof system h. Remember that for every fixed NPTM N that passes test T, there is a polynomial p such that the language

$$T_N = \{\langle N, x, 0^{p(|x|)} \rangle \mid x \in \Sigma^*\}$$

is a subset of T. Hence it follows that there is a proof system g for T with the property that for all $x \in \Sigma^*$,

$$g(1x) = \langle N, x, 0^{p(|x|)} \rangle.$$

Since h is a p-optimal proof system for T, there is a function $t \in \mathcal{FP}$ such that for every $x \in \Sigma^*$, $h(t(1x)) = \langle N, x, 0^{p(|x|)} \rangle$. Thus, $L_Q(N)$ is easily seen to reduce to the set

$$A = \{\langle N', x, 0^s, w \rangle \mid x \in L_Q(N') \wedge h(w) = \langle N', x, 0^s \rangle\}$$

via the reduction

$$f_N : x \mapsto \langle N, x, 0^{p(|x|)}, t(1x) \rangle.$$

Now, let $B = \{\langle N', x, 0^s, w \rangle \mid h(w) = \langle N', x, 0^s \rangle\}$. Notice that the reductions f_N defined above map only to elements in B. Therefore, any language C with the property that $A = C \cap B$ is hard for $\mathcal{C}$. We show that such a language C exists in the class $\mathcal{C}$. Let U be a universal NPTM according to A1 and let $U' = U \circ M$, where M is a transducer computing the projection that maps $\langle a, b, c, d \rangle$ to $\langle a, b, c \rangle$. Observe that (by A1 and A2) U' obeys R for all $y \in B$ and that $Q(U', \langle N, x, 0^s, w \rangle) \iff Q(N, x)$. Since by Lemma 6, $\mathcal{P}$-assertions are useful for (Q, R), and since $B \in \mathcal{P}$, it follows that there is a language $C \in \mathcal{C}$ with the property that $C \cap B = A \cap B = A$.

2 $\Longrightarrow$ 3. Let C be a many-one complete set for $\mathcal{C}$ and let N_C be an R-machine with $C = L_Q(N_C)$. Since C is complete for $\mathcal{C}$, any language L in $\mathcal{C}$ can be decided by an R-machine of the form $N_C \circ M$, where M is a polynomial-time transducer computing the reduction from L to C. (Notice that A2.2 implies that $N_C \circ M$ is an R-machine and that A2.1 implies that $L_Q(N_C \circ M) = L$). Thus, due to A2, the set

$$S = \{N_C \circ M \mid M \text{ is a PTM}\}$$

has the properties required for condition *3*.

3 $\Longrightarrow$ *4*. It suffices to observe that the set

$$T = \{\langle N, x, 0^s\rangle \mid N \in S, \text{ and } s \geq p_N(|x|)\}$$

is a (Q, R)-test set in $\mathcal{P}$. (Remember our assumption that a polynomial time bound p_N is attached to any NPTM N).

4 $\Longrightarrow$ *1*. This implication follows immediately from Lemma 3. ■

To obtain similar equivalences for the case of optimal proof systems, it suffices to require that the promise class under consideration can use $\mathcal{NP}$-assertions. However, the following equivalence even holds without this assumption.

Theorem 8 *Let $\mathcal{C}$ be a promise class and (Q, R) be a defining pair for $\mathcal{C}$. Then the following two conditions are equivalent.*

1. *$\mathcal{C}$ has a (Q, R)-test set with an optimal proof system.*
2. *$\mathcal{C}$ has a (Q, R)-test set in $\mathcal{NP}$.*

Proof. By Lemma 3, *2* implies *1*. For the opposite implication assume that we have a (Q, R)-test set T for $\mathcal{C}$ and an optimal proof system h for T. Let

$$T' = \{\langle N, x, 0^{s+t}\rangle \mid \exists w \in \Sigma^{\leq s}\colon\ h(w) = \langle N, x, 0^t\rangle\}.$$

Clearly, $T' \in \mathcal{NP}$. To show that T' is a (Q, R)-test set for $\mathcal{C}$, we prove that each machine N which passes T also passes T'. If N passes T, then there is some polynomial p bounding the running time of N and having the property that for all $x \in \Sigma^*$, $\langle N, x, 0^{p(|x|)}\rangle \in T$. It is easy to define a proof system g for $T_N = \{\langle N, x, 0^{p(|x|)}\rangle \mid x \in \Sigma^*\}$ such that for any $x \in \Sigma^*$, $g(1x) = \langle N, x, 0^{p(|x|)}\rangle$. As h simulates g, there is a polynomial q such that for all x, $\langle N, x, 0^{p(|x|)}\rangle$ has an h-proof of size $q(|x|)$. Thus for any x, $\langle N, x, 0^{p(|x|)+q(|x|)}\rangle \in T'$. But this means that N passes test T'. ■

Theorem 9 *Let $\mathcal{C}$ be a promise class and let (Q, R) be a defining pair for $\mathcal{C}$ which fulfills A1. If $\mathcal{NP}$-assertions are useful for (Q, R), then 1 implies 2.*

1. *$\mathcal{C}$ has a (Q, R)-test set with an optimal proof system.*
2. *$\mathcal{C}$ has a many-one complete set.*

Proof. Let T' be a (Q, R)-test set for $\mathcal{C}$ which has an optimal proof system. By Theorem 8, there is a (Q, R)-test set T for $\mathcal{C}$ which is in $\mathcal{NP}$. Consider the set

$$A \quad = \quad \{\langle N, x, 0^s\rangle \in T \mid x \in L_Q(N)\}.$$

Notice that $A = L_Q(U) \cap T$ where U is a universal NPTM given by A1. Notice also that by A1.2, U obeys R on any $y \in T$. As $\mathcal{NP}$-assertions are useful for (Q, R), there is a language $C \in \mathcal{C}$ such that $C \cap T = L_Q(U) \cap T = A$. We now show that C is complete for $\mathcal{C}$. Let L be a set in $\mathcal{C}$ and let N be an NPTM with $L = L_Q(N)$ which passes test T with respect to a polynomial p. Then the mapping $x \mapsto \langle N, x, 0^{p(|x|)}\rangle$ reduces L to C (as well as to A). ■

Notice that the two conditions 1 and 2 of Theorem 9 are equivalent if we additionally require that (Q, R) fulfills A2. This follows from the fact stated in Theorem 7 that the existence of a complete language implies the existence of a test set in $\mathcal{P}$.

5 Applications

In this section we derive complete problems for various promise classes from the assumption that certain languages have (p-)optimal proof systems. We start by sketching how defining pairs (Q, R) (i.e. machine models) for $\mathcal{NP} \cap$ *co*-$\mathcal{NP}$ and $\Sigma_2^p \cap \Pi_2^p$ can be defined.

A machine model for $\mathcal{NP} \cap$ *co*-$\mathcal{NP}$ can be obtained by combining two $\mathcal{NP}$-machines N_1 and N_2 which accept complementary languages into a machine N that in the first nondeterministic step, branches left to N_1 and right to N_2. So, for $\mathcal{NP} \cap$ *co*-$\mathcal{NP}$ the promise $R(N, x)$ states that on input x, either N_1 or N_2 accepts but not both. $Q(N, x)$ holds if N_1 has an accepting path on input x.

A machine model for $\Sigma_2^p \cap \Pi_2^p$ can be obtained in a similar way by combining two Σ_2^p-machines which accept complementary languages (Σ_2^p-machines may be defined syntactically or by the tree-structure of an NPTM). So the promise $R(N, x)$ holds when N branches in the first nondeterministic step to two Σ_2^p-computations, where the left one is accepting if and only if the right one is rejecting. Here $Q(N, x)$ holds when N_1 is accepting x in a Σ_2^p-way.

Notice that in both cases the defining pairs fulfill A1 and A2.

Proposition 10

- *$\mathcal{NP} \cap$ co-$\mathcal{NP}$ has a test set in* D^p.
- *$\Sigma_2^p \cap \Pi_2^p$ has a test set which is $\leq_m^p$-reducible to* $\mathrm{SAT}_2 \times \mathrm{TAUT}_2$.

Proof. Let $R(N, x)$ be the promise, described above, i.e., $R(N, x)$ holds if and only if N behaves like an $\mathcal{NP} \cap$ *co*-$\mathcal{NP}$-machine on input x. Clearly, the set

$$T = \{\langle N, x, 0^s\rangle \mid R(N, x) \text{ and } s \geq p_N(|x|)\}$$

is a test set for $\mathcal{NP} \cap$ *co*-$\mathcal{NP}$. Observe that $R(N, x)$ holds if there exists an accepting path α of N on input x, and if there is no pair β_1, β_2 of accepting paths of N on input x such that in the first nondeterministic step β_1 branches left and β_2 branches right. This shows that $T \in \mathrm{D}^p$. The statement for $\Sigma_2^p \cap \Pi_2^p$ can be proved in a similar way. ■

We now show that $\mathcal{NP}$ assertions are useful for $\Sigma_2^p \cap \Pi_2^p$ (considering the machine model defined above). Hence, in order to get a many-one complete set for $\Sigma_2^p \cap \Pi_2^p$, it suffices to find a test set that has an optimal proof system.

Proposition 11 *$\mathcal{NP}$ assertions are useful for $\Sigma_2^p \cap \Pi_2^p$.*

Proof. Let $B \in \mathcal{NP}$ and let N be an NPTM behaving on any input $x \in B$ like a $\Sigma_2^p \cap \Pi_2^p$-machine. Let A be the set of strings accepted by the left Σ_2^p-computation of N, and let C be the set of strings accepted by the right Σ_2^p computation of N. Thus, all $x \in B$ fulfill the promise that $x \in A$ if and only if $x \notin C$, implying that $A \cap B = \overline{C} \cap B$. Notice that $A = L_Q(N)$. Therefore it suffices to show that $A \cap B \in \Sigma_2^p \cap \Pi_2^p$. Since $A \in \Sigma_2^p$ and since $\overline{C} \in \Pi_2^p$, it follows that $A \cap B \in \Sigma_2^p$ and that $A \cap B = \overline{C} \cap B \in \Pi_2^p$. ∎

Actually, the proof shows that even $\Sigma_2^p \cap \Pi_2^p$-assertions are useful for $\Sigma_2^p \cap \Pi_2^p$. As D^p is the $\leq_m^p$-closure of $\mathrm{SAT} \times \mathrm{TAUT}$, Lemma 1 and Lemma 2 imply that any set in D^p has a p-optimal proof system provided that TAUT as well as SAT have p-optimal proof systems. Hence, we can use Theorems 7 and 9 to get the following implications.

Theorem 12

- *If* TAUT *and* SAT *have p-optimal proof systems, then $\mathcal{NP} \cap$ co-$\mathcal{NP}$ has a $\leq_m^p$-complete set.*
- *If* TAUT_2 *and* SAT_2 *have optimal proof systems, then $\Sigma_2^p \cap \Pi_2^p$ has a $\leq_m^p$-complete set.*

In a similar way it can be shown that for any $k \geq 2$, $\Pi_k^p \cap \Sigma_k^p$ has a $\leq_m^p$-complete set provided that TAUT_k and SAT_k have optimal proof systems.

Test sets for probabilistic classes

We show now that the probabilistic complexity classes $\mathcal{BPP}$, $\mathcal{RP}$, and $\mathcal{ZPP}$ have test sets in Σ_2^p as well as in Π_2^p. To this end we recall the definitions and basic properties of hashing that we need. Sipser [Sip83] used universal hashing, originally invented by Carter and Wegman [CW79], to estimate (probabilistically) the size of a finite set X of strings.

A linear hash function h from Σ^m to Σ^k is given by a boolean (k, m)-matrix (a_{ij}) and maps any string $x = x_1 \ldots x_m$ to a string $y = y_1 \ldots y_k$, where y_i is the inner product $a_i \cdot x = \sum_{j=1}^{m} a_{ij} x_j$ (mod 2) of the i-th row a_i and x.

Let $X \subseteq \Sigma^m$ and let h be a linear hash function from Σ^m to Σ^k. Then we say that h *hashes* X if for all pairs of different strings $x, y \in X$, $h(x) \neq h(y)$. More generally, if H is a family $(h_1, \ldots, h_s)$ of linear hash functions from Σ^m to Σ^k, then we say that H *hashes* X if for every $x \in X$ there is some i, $1 \leq i \leq s$, such that $h_i(x) \neq h_i(y)$, for all $y \in X - \{x\}$.

Note that the predicate "H hashes X" can be decided in co-$\mathcal{NP}$ provided that membership in X can be tested in $\mathcal{P}$. We denote the set of all families $H = (h_1, \ldots, h_k)$ of k linear hash functions from Σ^m to Σ^k by $\mathcal{H}(k, m)$.

As observed by Sipser, the size of a set $X \subseteq \Sigma^m$ can be estimated by checking for which values of k it is hashable by some hash family $H \in \mathcal{H}(k, m)$.

Lemma 13 [Sip83] *No hash family $H \in \mathcal{H}(k, m)$ can hash a set $X \subseteq \Sigma^m$ of cardinality $|X| > k2^k$. Furthermore, if $|X| \leq 2^k$, then some hash family $H \in \mathcal{H}(k, m)$ hashes X.*

The next two lemmas make use of Stockmeyer's refinement of the hashing technique [Sto85]. Their proofs are straightforward (see, e.g., [Köb95]).

Lemma 14 *Let $X \subseteq \{0,1\}^l$ and let $m = 1 + 72l$ and $k = 1 + m(l-2)$ be integers. Then the following implications hold.*

- *If there exists a hash family $H \in \mathcal{H}(k, lm)$ that hashes X^m, then $|X| \leq 2^l/3$.*
- *If $|X| \leq 2^l/4$, then some hash family $H \in \mathcal{H}(k, lm)$ hashes X^m.*

Lemma 15 *Let $X \subseteq \{0,1\}^l$ and let $m = 1 + 512l$ and $k = 1 + \lceil m(l + 1 - \log 3) \rceil$ be integers. Then the following implications hold.*

- *If there exists a hash family $H \in \mathcal{H}(k, lm)$ that hashes X^m, then $|X| \leq 2^l \cdot 3/4$.*
- *If $|X| \leq 2^l \cdot 2/3$, then some hash family $H \in \mathcal{H}(k, lm)$ hashes the set X^m.*

Proposition 16 *$\mathcal{BPP}$, $\mathcal{RP}$, $\mathcal{ZPP}$, and $\mathcal{MA}$ have test sets in Σ_2^p as well as in Π_2^p. $\mathcal{AM}$ has a test set in Π_2^p.*

Proof. Consider the following set B containing only descriptions of NPTMs that have bounded error probability on a given input x (when interpreted as probabilistic machines).

$$B = \{\langle N, x, 0^l \rangle \mid l = p_N(|x|) \text{ and for } m = 1 + 72l \text{ and } k = 1 + m(l-2) \text{ there is a hash family } H \in \mathcal{H}(k, lm) \text{ that hashes either } \mathrm{Acc}(N, l, x) \text{ or } \mathrm{Rej}(N, l, x)\}.$$

Recall that for any NPTM N, p_N is the polynomial time bound associated with N. The set $\mathrm{Acc}(N, l, x)$ (resp., $\mathrm{Rej}(N, l, x)$) contains all sequences $r \in \{0,1\}^l$ of random bits using which $N(x)$ accepts (rejects, respectively) after at most l probabilistic moves.

Clearly, $B \in \Sigma_2^p$ and by using Lemma 14 (as well as the probability amplification property of $\mathcal{BPP}$) it is easy to see that B is a test set for $\mathcal{BPP}$. Next we show that $\mathcal{BPP}$ has a test set in Π_2^p. In fact, consider the set

$C = \{\langle N, x, 0^l\rangle \mid l = p_N(|x|)$ and for $m = 1 + 512l$ and $k = 1 + \lceil m(l + 1 - \log 3)\rceil$ there is no hash family $H \in \mathcal{H}(k, lm)$ that hashes both $\mathrm{Acc}(N, l, x)$ and $\mathrm{Rej}(N, l, x)\}$

which belongs to Π_2^p. By Lemma 15 it is easy to see that C is a test set for $\mathcal{BPP}$. Moreover, it is not hard to adapt the above argument to get suitable test sets for $\mathcal{RP}$, $\mathcal{ZPP}$, and $\mathcal{MA}$, and to get a test set for $\mathcal{AM}$ in Π_2^p ∎

It is easily seen that $\mathcal{MA}$ as well as $\mathcal{AM}$ can use $\mathcal{NP}$-assertions. Therefore we obtain

Theorem 17 *If there exists a p-optimal (or, for the case of $\mathcal{MA}$, an optimal) proof system for SAT_2 or for TAUT_2, then $\mathcal{BPP}$, $\mathcal{RP}$, $\mathcal{ZPP}$, and $\mathcal{MA}$ have many-one complete sets. If TAUT_2 has an optimal proof system then $\mathcal{AM}$ has a many-one complete set.*

6 Completeness results for function classes

The results in Section 4 can be translated in a straightforward way to promise function classes. Due to the limited space we just give a brief sketch. The definition of a promise R for function classes is the same as for languages, whereas the acceptance criterion Q is replaced by a function S mapping each pair (N, x) consisting of an NPTM N and a string x to some string $S(N, x)$. The function $F_S(N) : \Sigma^* \to \Sigma^*$ computed by N (when applying S) is given by

$$F_S(N)(x) = S(N, x).$$

R and S together define the function class

$$\mathcal{F}_{S,R} = \{F_S(N) \mid N \text{ is an } R\text{-machine}\}.$$

Conditions A1 and A2 translate to the corresponding conditions A1′ and A2′ for function classes. We just have to replace A1.1 and A2.1 by

A1′.1: $S(N, x) = S(U, \langle N, x, 0^s\rangle)$ and

A2′.1: $S(N, f_M(x)) = S(N \circ M, x)$

respectively. We use the following notion of many-one reducibility for functions: $g \leq_m^p h$ if there is a function $f \in \mathcal{FP}$ such that for any $x \in \Sigma^*$, $h(f(x)) = g(x)$. Notice that this notion is closely related to the notion of p-simulation (although g and h need not belong to $\mathcal{FP}$).

It is also straightforward to translated the definition of a test set. The notion of usefulness for a defining pair (S, R) for a function class $\mathcal{F}_{S,R}$ reads as follows. $\mathcal{A}$-assertions are called useful for (S, R) if for any language $B \in \mathcal{A}$ and any NPTM N the following holds: if $R(N, x)$ for any $x \in B$ then there is a function $f \in \mathcal{F}_{S,R}$ such that for all $x \in B$, $f(x) = S(N, x)$.

Theorems 7, 8, and 9 also translate to promise function classes. We first give the translation of the main equivalence of Theorem 7.

Theorem 18 *Let $\mathcal{F}$ be a promise function class and let (S, R) be a defining pair for $\mathcal{F}$ which fulfills A1′ and A2′. Then the following conditions are equivalent.*

1. *$\mathcal{F}$ has a (S, R)-test set with a p-optimal proof system.*
2. *$\mathcal{F}$ has a many-one complete set.*

Translating Theorem 9 to the functional setting yields the following sufficient condition for the existence of many-one complete functions.

Theorem 19 *Let $\mathcal{F}$ be a promise function class and let (S, R) be a defining pair for $\mathcal{F}$ which fulfills A1′. If $\mathcal{NP}$-assertions are useful for (S, R), then 1 implies 2:*

1. *$\mathcal{F}$ has a (S, R)-test set with an optimal proof system.*
2. *$\mathcal{F}$ has a many-one complete function.*

We now apply this result and show that the class of disjoint $\mathcal{NP}$-pairs has a strongly many-one complete pair if TAUT has an optimal proof system. We associate to each disjoint $\mathcal{NP}$-pair (A, B) a function $f_{(A,B)}$ as follows. For all $x \in \Sigma^*$,

$$f_{(A,B)}(x) = \begin{cases} 0 & x \in A, \\ 1 & x \in B, \\ \lambda & \text{otherwise.} \end{cases}$$

In a sense, the class of disjoint $\mathcal{NP}$-pairs corresponds to the function class of all those functions. This class can be defined as a promise class in the following way. An NPTM N is an R-machine if in the first nondeterministic step it branches to two $\mathcal{NP}$-machines N_1 and N_2 which accept disjoint languages. Therefore $R(N, x)$ holds if there is no pair α_1, α_2 of accepting paths of N on input x such that in the first nondeterministic step α_1 branches left and α_2 branches right. If there is an accepting path branching left, the value of $S(N, x)$ is 0. Otherwise, $S(N, x) = 1$ if there is an accepting path branching right, and $S(N, x) = \lambda$ if there isn't any accepting path. This defines the class $\mathcal{F}_{S,R}$ which has a $\leq_m^p$-complete function if and only if there is a strongly many-one complete disjoint $\mathcal{NP}$-pair. It is easy to see that $\mathcal{F}_{S,R}$ has a test set in co-$\mathcal{NP}$ and that $\mathcal{NP}$-assertions are useful for $\mathcal{F}_{S,R}$. Therefore, Theorem 19 gives us the following consequence.

Theorem 20 *If TAUT has an optimal proof system, then there is a pair that is strongly many-one complete for the class of all disjoint $\mathcal{NP}$-pairs.*

7 When do optimal proof systems exist?

In this section we investigate conditions which imply the existence of (p-)optimal proof systems. We first make an observation which allows us to infer the existence of a p-optimal proof system for a recursively enumerable set L, provided that there are complete functions for certain promise function classes. Secondly, we will see that collapses of tally sets at the double exponential-time level imply the existence of (p-)optimal proof systems for TAUT_k. This improves a result in [MT98].

For any recursively enumerable set $L \notin \{\emptyset, \Sigma^*\}$, the function class $\mathcal{PS}_L = \{f \in \mathcal{FP} \mid f(\Sigma^*) \subseteq L\}$ is the $\leq^p_m$-closure of the class $\{f \in \mathcal{FP} \mid f(\Sigma^*) = L\}$ that consists of all proof systems for L. Clearly, $\mathcal{PS}_L$ has a $\leq^p_m$-complete function if and only if there is a p-optimal proof system for L. Furthermore the class $\mathcal{PS}_L$ is easily described as a promise function class by the following defining pair (S, R) which fulfills A1′ and A2′.

1. $R(N, x)$ holds if on input x, N only makes deterministic moves and halts with some string $v \in L$ on its tape.
2. $S(N, x) = v$ where v is the string produced by N on input x on its leftmost nondeterministic computation.

It is also possible to describe this class using the tree structure approach for promise classes. Here, the idea is to allow only special trees (called combs in [Bor94]) which represent polynomial-time computable functions.

An (S, R)-test set is given by the set

$$T_L = \{\langle N, x, 0^s\rangle \mid R(N, x) \text{ and } s = p_N(|x|)\}.$$

It is easy to see that $T_L \leq^p_m L$ via a PTM M which on input $\langle N, x, 0^s\rangle$ simulates N for at most s steps as follows. If N does only perform deterministic moves, then M behaves exactly as N. Otherwise, M outputs a fixed string $y \notin L$. Notice that also $L \leq^p_m T_L$ via $f : x \mapsto \langle N_{\mathrm{id}}, x, 0^{p(|x|)}\rangle$, where N_{id} is an NPTM computing the identity mapping and p is the polynomial time bound of N_{id}; implying that $T_L \equiv^p_m L$.

Combining these observations with Theorem 18 we quite easily obtain the following theorem.

Theorem 21 *Let L be a recursively enumerable language, where $\emptyset \neq L \neq \Sigma^*$. Then the following statements are equivalent.*

1. *There is a p-optimal proof system for L.*
2. *Every promise function class $\mathcal{F}$ which has a defining pair (S, R) fulfilling A1′ and A2′, and further possesses an (S, R)-test set which is $\leq^p_m$-reducible to L has a $\leq^p_m$-complete function.*
3. *$\mathcal{PS}_L$ has a $\leq^p_m$-complete function.*

It would be interesting to know whether a similar theorem holds when considering just language classes instead of function classes.

We now give a sufficient condition for the existence of a (p-)optimal proof system for TAUT_k. Let us call a set T *super-tally* (in symbols: $T \in$ *Super-Tally*) if T is a subset of $\{0^{2^{2^n}} \mid n \geq 0\}$.

Proposition 22 *If any super-tally set in $\mathcal{NP}$ is already in $\mathcal{P}$, then* TAUT *has a p-optimal proof system.*

Proof. We assume some standard enumeration M_1, M_2, M_3, ... of (encodings of) deterministic Turing transducers with binary input alphabet such that for a given triple $\langle M_i, x, 0^k\rangle$, up to k steps of the computation of M_i on input x can be efficiently simulated. Let $i(k)$ denote the largest exponent i such that 2^i divides k and consider the language

$$T = \{\, 0^{2^{2^k}} \mid \text{on any input, } M_{i(k)} \text{ either stops after at most } 2^{2^k} \text{ steps and outputs some tautology or } M_{i(k)} \text{ runs for more than } 2^{2^k} \text{ steps}\,\}.$$

In order to decide whether a given string $0^n = 0^{2^{2^k}}$ belongs to T, it suffices to simulate $M_{i(k)}$ on any input of length at most $n+1$ for at most $n+1$ steps. This shows that $\overline{T} \cap \{\, 0^{2^{2^k}} \mid k \geq 0\,\} \in \mathcal{NP}$ and thus, by the assumption that any super-tally set in $\mathcal{NP}$ is already in $\mathcal{P}$, T can be decided in $\mathcal{P}$. We claim that the following transducer computes a p-optimal proof system h for TAUT.

input $\langle 0^n, w\rangle$
if $0^n \in T$ **then**
  determine k such that $n = 2^{2^k}$
  if $M_{i(k)}$ stops on input w in at most n steps **then**
    output $M_{i(k)}(w)$ and halt;
output some fixed tautology x_0 and halt.

Since, as is not hard to see, $h(\Sigma^*) \subseteq \mathrm{TAUT}$ and $h \in \mathcal{FP}$, it only remains to show that h is p-optimal. Let g be any proof system for TAUT, computed by some deterministic Turing transducer M_i in time bounded by some polynomial p. Then any g-proof w can be translated into an h-proof by the mapping $w \mapsto \langle 0^{2^{2^k}}, w\rangle$, where k is the smallest integer $k_j = (2j+1)2^i$, $j \geq 0$, such that $p(|w|) \leq 2^{2^{k_j}}$. Since

$$2^{2^{k_{j+1}}} \quad = \quad \left(2^{2^{k_j}}\right)^c,$$

where $c = 2^{2 \cdot 2^i}$, it follows that $2^{2^k} < p(|w|)^c$, implying that the translation $w \mapsto \langle 0^{2^{2^k}}, w\rangle$ is computable in polynomial time. ∎

It is interesting to note that the above proof still goes through if we define a set T to be super-tally if it is a subset of $\{0^{c^{c^k}} \mid k \geq 0\}$ where $c \geq 2$ is an arbitrary integer constant. However, in our proof, we cannot allow T to be any sparser. To see why, let us just try to replace the function $j \mapsto 2^{2^{(2j+1)2^i}}$ by some other function $f(j)$ (where f as well as the constant c below might depend on i). This would guarantee that the new set T is a subset of $\{0^{f(j)} \mid j \geq 0\}$. On the other hand, a necessary condition for the proof to work is that for some constant c, $f(j+1) \leq f(j)^c$, implying that $f(j) \leq f(0)^{c^j}$.

A similar proof shows that there is an optimal proof system for TAUT, provided that any super-tally set in $\mathcal{NP}$ is also in *co*-$\mathcal{NP}$.

Let $\mathcal{EE} = \text{DTIME}(2^{O(2^n)})$, $\mathcal{EEE} = \text{DTIME}(2^{O(2^{2^n})})$ and let $\mathcal{NEE}$, $\mathcal{NEEE}$ be their nondeterministic counterparts. Using a technique in the style of [Boo74] it is easy to see that each tally language in $\mathcal{EE}$ ($\mathcal{NEE}$) translates to a super-tally language in $\mathcal{P}$ (respectively $\mathcal{NP}$) and vice versa. Thus a collapse of tally sets at the $\mathcal{EE}$-level (as, e.g., $\mathcal{NEE} \cap \textit{Tally} \subseteq \mathcal{EE}$ or $\mathcal{NEE} \cap \textit{Tally} \subseteq$ *co*-$\mathcal{NEE}$) corresponds to a collapse for super-tally sets at the $\mathcal{P}$-level (i.e., $\mathcal{NP} \cap \textit{Super-Tally} \subseteq \mathcal{P}$ and $\mathcal{NP} \cap \textit{Super-Tally} \subseteq$ *co*-$\mathcal{NP}$, respectively). As a consequence we can state the following corollary.

Corollary 23

- *If* $\mathcal{NEE} \cap \text{Tally} \subseteq \mathcal{EE}$ *then* TAUT *has a p-optimal proof system.*
- *If* $\mathcal{NEE} \cap \text{Tally} \subseteq$ *co*-$\mathcal{NEE}$ *then* TAUT *has an optimal proof system.*

Surprisingly, it seems hard to improve the above corollary further to the triple exponential time level. In fact, it is stated in [BDG98] that there is a relatized world in which $\mathcal{NEEE} = \mathcal{EEE}$ but TAUT does not have an optimal proof system.

Finally, a generalization of Proposition 22 to any level of the polynomial time hierarchy yields the following sufficient conditions for the existence of a (p-)optimal proof systems for TAUT_k.

Theorem 24

- *If any super-tally set in* Π_k^p *is already in* $\mathcal{P}$*, then* TAUT_k *has a p-optimal proof system.*
- *If any super-tally set in* Π_k^p *is already in* $\mathcal{NP}$ *then,* TAUT_k *has an optimal proof system.*

References

[BCS92] D. P. BOVET, P. CRESCENZI, AND R. SILVESTRI. A uniform approach to define complexity classes. *Theoretical Computer Science*, **104**:263–283, 1992.

[BDG90] J. L. BALCÁZAR, J. DÍAZ, AND J. GABARRÓ. *Structural Complexity II*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1990.

[BDG95] J. L. BALCÁZAR, J. DÍAZ, AND J. GABARRÓ. *Structural Complexity I*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, second edition, 1995.

[BDG98] S. BEN-DAVID AND A. GRINGAUZE. On the existence of optimal proof systems and oracle-relativized propositional logic. Manuscript, 1998

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

[Bor94] B. BORCHERT. *Predicate Classes, Promise Classes, and the Acceptance Power of Regular Languages*. PhD thesis, Ruprecht-Karls-Universität Heidelberg, 1994.

[CR79] S. A. COOK AND R. A. RECKHOW. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, **44**(1):36–50, 1979.

[CW79] J. L. CARTER AND M. N. WEGMAN. Universal classes of hash functions. *Journal of Computer and System Sciences*, **18**:143–154, 1979.

[Köb95] J. KÖBLER. Lowness-Eigenschaften und Erlernbarkeit von Booleschen Schaltkreisklassen. Habilitation Thesis, Universität Ulm, 1995.

[KP89] J. KRAJÍČEK AND P. PUDLÁK. Propositional proof systems, the consistency of first order theories and the complexity of computations. *The Journal of Symbolic Logic*, **54**(3):1063–1079, 1989.

[MT98] J. MESSNER AND J. TORÁN. Optimal proof systems for propositional logic and complete sets. In *Proc. 15th Symposium on Theoretical Aspects of Computer Science '98*, Lecture Notes in Computer Science #1373, 477–487. Springer-Verlag, 1998.

[Pud86] P. PUDLÁK. On the length of proofs of finitistic consistency statements in first order theories. In J. B. Paris et al., editor, *Logic Colloquium'84*, 165–196, 1986.

[Raz94] A. A. RAZBOROV. On provably disjoint NP-pairs. Technical Report RS-94-36, Basic Research in Computer Science Center, Aarhus, 1994.

[Sad97] Z. SADOWSKI. On an optimal quantified propositional proof system and a complete language for NP ∩ co-NP. In *Proc. 11th International Symposium on Fundamentals of Computing Theory*, Lecture Notes in Computer Science #1279, 423–428. Springer-Verlag, 1997.

[Sip83] M. SIPSER. A complexity theoretic approach to randomness. In *Proc. 15th ACM Symposium on Theory of Computing*, 330–335. ACM Press, 1983.

[Sto85] L. STOCKMEYER. On approximation algorithms for #P. *SIAM Journal on Computing*, **14**(4):849–861, 1985.

[Ver93] N. K. VERESHCHAGIN. Relativizable and non-relativizable theorems in the polynomial theory of algorithms. *Izvestija Rossijskoj Akademii Nauk*, **57**:51–90, 1993. In Russian.