On an optimal quantified propositional proof system and a complete language for NP $\cap$ co-NP

Zenon Sadowski

Institute of Mathematics,
University of Warsaw, Białystok Division
15-257 Białystok, ul. Akademicka 2, Poland
e-mail: `sadowski@math.uw.bialystok.pl`

Abstract. In this paper we prove that if there exists an optimal quantified propositional proof system then there exists a complete language for **NP** $\cap$ **co-NP**.

1 Introduction

There is a lot of activity around the complexity of propositional proof systems (see [7, 12]). Research into propositional proof systems is motivated by the open problem **NP=co-NP?**

Research into quantified propositional proof systems is not so popular. The study of quantified propositional proof systems may be related to the problem **NP=PSPACE?** Some deep results about connections between quantified propositional proof systems and bounded arithmetic are contained in [8].

We propose to study the problem of the existence of an optimal quantified propositional proof system. The similar problem for propositional proof systems has been studied in [9].

It is not known whether complete languages exist for **NP** $\cap$ **co-NP** and Sipser has shown in [10] that there are relativizations so that **NP** $\cap$ **co-NP** has no complete languages (see also [4, 6]).

The problem whether **NP** $\cap$ **co-NP** has complete languages seems, at first glance, distant from the subject of the optimal quantified propositional proof system. In our paper we bring together these two areas of research and we show that the existence of an optimal quantified propositional proof system implies the existence of a complete language for **NP** $\cap$ **co-NP**.

The inspiration to write this paper comes from [6].

2 Preliminaries

The symbol Σ denotes, throughout the paper, a certain fixed finite alphabet. This alphabet contains 0 and 1, the set of propositional connectives $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ and some auxiliary symbols. Strings describing Turing machines and propositional formulas considered in this paper are taken over Σ. The set of all strings over Σ is denoted by $\Sigma^{\star}$. If M is a Turing machine, then $L(M)$ denotes the set of all strings accepted by M. For a string x, $|x|$ denotes the length of x.

Let p_1, p_2, p_3, ... be the set of propositional variables.

Definition 1. The class of quantified propositional formulas is the smallest class which is defined by (a)–(c):

(a) Every propositional variable is a quantified propositional formula.
(b) If α_1 and α_2 are quantified propositional formulas then $\neg\alpha_1$, $\alpha_1 \wedge \alpha_2$, $\alpha_1 \vee \alpha_2$, $\alpha_1 \rightarrow \alpha_2$, $\alpha_1 \leftrightarrow \alpha_2$ are quantified propositional formulas.
(c) If α is a quantified propositional formula then $(\exists p_i)(\alpha)$ and $(\forall p_i)(\alpha)$ are quantified propositional formulas.

The semantical meaning of $(\exists p_i)(\alpha)$ is $\alpha(0) \vee \alpha(1)$ and of $(\forall p_i)(\alpha)$ is $\alpha(0) \wedge \alpha(1)$, where 0 and 1 are constants falsity and truth and $\alpha(0)$ is the formula obtained by substituting 0 for every free occurrence of p_i in α. A quantified propositional formula α is a tautological quantified propositional formula if every truth assignment makes α true. The set of all tautological quantified propositional formulas we denote throughout the paper by the symbol $QTAUT$.

Fact 1. *(cf. [1, 5, 11]) $QTAUT$ is* **PSPACE**-*complete.*

Definition 2. (cf. [3, 8]) A quantified propositional proof system is a function $F : \Sigma^\star \xrightarrow{onto} QTAUT$ computable by a deterministic Turing machine in time bounded by a polynomial in the length of the input.
A string w such that $F(w) = \alpha$ we call a proof of a formula α.

Definition 3. (cf. [3]) A quantified propositional proof system is polynomially bounded if and only if there exists a polynomial $p(n)$ such that every $\alpha \in QTAUT$ has a proof of length no more than $p(|\alpha|)$ in this system.

Fact 2. **NP=PSPACE** *if and only if there exists a polynomially bounded quantified propositional proof system.*

Proof. Cook and Reckhow proved that **NP=co-NP** if and only if there exists a polynomially bounded propositional proof system (see [3]). The proof of Fact 2 is analogous to their proof.

Definition 4. (cf. [3]) Quantified propositional proof system P polynomially simulates quantified propositional proof system Q if there exists polynomial time computable function $f : \Sigma^\star \longrightarrow \Sigma^\star$ such that for every w, if w is a proof of α in Q, then $f(w)$ is a proof of α in P.

Definition 5. (cf. [9]) Quantified propositional proof system P is optimal if for every other quantified propositional proof system Q, the system P polynomially simulates Q.

Problem. (cf. [9]) Whether there exists a quantified propositional proof system which is optimal?

Fact 3.
If an optimal quantified propositional proof system exists then **NP=PSPACE** *if and only if this system is polynomially bounded.*

3 Formulas expressing the strong nondeterminism of Turing machines

Definition 6. A Turing machine M is called a YES-NO nondeterministic Turing machine if M is nondeterministic and possesses two distinguished states, the accepting state YES, and the rejecting state NO.

Definition 7. A YES-NO nondeterministic Turing machine M is called a strong nondeterministic Turing machine if the following condition holds: For any input x, either some computation path of M accepts x and no computation path of M rejects x, or some computation path of M rejects x and no computation path of M accepts x.

Definition 8. A YES-NO nondeterministic Turing machine M is called n-strong nondeterministic Turing machine if conditions (i) - (ii) hold:

(i) For any input x of length n there exists a computation path of M accepting x or there exists a computation path of M rejecting x.
(ii) For any input x of length n there does not exist simultaneously a computation path of M accepting x and a computation path of M rejecting x.

If M is a nondeterministic YES-NO machine and k is a natural number then the symbol M^k will denote the machine obtained by attaching the shut-off clock $n^k + k$ to the machine M (n is the length of the input). The construction of M^k is carried out in such a way that M^k is a one-tape Turing machine and the clock stops computations in a state which is neither rejecting nor accepting. So, for any string w, the following equivalences hold: M^k accepts w if and only if M accepts w in no more than $|w|^k + k$ steps. M^k rejects w if and only if M rejects w in no more than $|w|^k + k$ steps.

We can assume that n^{2k} is the polynomial which bounds the running time of M^k. To any YES-NO nondeterministic machine M and any natural number k we will assign the set $B_{M,k} = \{SN^1_{M,k}, SN^2_{M,k}, SN^3_{M,k}, SN^4_{M,k}, ...\}$ of quantified propositional formulas such that:

(1) $SN^n_{M,k}$ can be constructed in time polynomial in n. It means, that for the set $B_{M,k}$ there exists a transducer working in polynomial time which on input 0^n (the sequence of n zeros) constructs the formula $SN^n_{M,k}$, where $n = 1, 2, 3, ...$
(2) $SN^n_{M,k}$ is a tautological quantified propositional formula if and only if the machine M^k is n-strong nondeterministic.

Fix n natural. Our construction of the formula $SN^n_{M,k}$ is adapted from Cook's proof that SAT is **NP**-complete (cf. [2]). The formula $SN^n_{M,k}$ is of the form $\alpha \wedge \neg\beta$, where α is a quantified propositional formula and β is a propositional formula. The formula α evaluates to "true" if and only if for any input w of length n there exists a computation of M^k accepting w or there exists a computation of M^k rejecting w. The formula β is satisfiable if there exists a string w of length n such that M^k simultaneously accepts w and rejects w.

Now we describe the construction of the formula $SN^n_{M,k}$ more precisely.

Let $\{q_1, q_2, q_3, ..., q_r\}$ denote the set of states of M^k and let $\{\sigma_1, \sigma_2, \sigma_3, ..., \sigma_v\}$ denote the set of its tape symbols. The propositional variables occuring in the formula $SN^n_{M,k}$ are divided into three groups: the variables with the superscript 1, the variables with the superscript 2 and the variables with the superscript 3. The variables which differ only in their superscripts have the same intended interpretation.

Now we describe the intended interpretations of these variables.
$S^i(j,l,m) = 1$ if and only if at the j-th step the l-th cell of the tape M^k contains the symbol σ_m, where $1 \leq l \leq n^{2k}, 1 \leq j \leq n^{2k}, 1 \leq m \leq v, i = 1,2,3$.
$Q^i(j,l) = 1$ if and only if at the j-th step M^k is at state q_l, where $1 \leq j \leq n^{2k}, 1 \leq l \leq r, i = 1,2,3$.
$H^i(j,l) = 1$ if and only if at the j-th step the l-th cell of the tape is currently scanned by M^k, where $1 \leq j \leq n^{2k}, 1 \leq l \leq n^{2k}, i = 1,2,3$.

Using these variables we construct propositional formulas G_0, G^1_1, G^2_1, G^3_1, G^1_2, G^2_2, G^3_2, G^1_3, G^2_3, G^3_3, G^1_4, G^2_4, G^3_4, G^1_5, G^2_5, G^3_5, G^1_6, G^2_7, G^3_8. If a formula from this list possesses a superscript then only variables with the same superscript may appear in this formula. These formulas express the following conditions:
G^1_1, G^2_1, G^3_1: At any step M^k is at a unique state.
G^1_2, G^2_2, G^3_2: At any step the head of M^k is scanning a unique cell of the tape.
G^1_3, G^2_3, G^3_3: At any step every cell of M^k contains a unique symbol.
G^1_4, G^2_4, G^3_4: Every configuration of M^k follows from the previous one by a legal move of M^k.
G^1_5, G^2_5, G^3_5: At the initial step M^k is at the initial state and the head of M^k is scanning the first cell of the tape. The first n cells do not contain blank, the remaining $n^{2k} - n$ cells are blank.
G^1_6, G^2_6: Within n^{2k} steps M^k reaches YES state.
G^1_7, G^3_7: Within n^{2k} steps M^k reaches NO state.
The formula G_0 is of the form

$$\prod_{1 \leq j \leq n, 1 \leq m \leq v} (S^2(1,j,m) \leftrightarrow S^3(1,j,m))$$

Let $\vec{x}$ be the "vectorial" boolean variable consisting of the following variables $S^1(1,l,m)$ where $1 \leq l \leq n, 1 \leq m \leq v$. Any valuation to $\vec{x}$ describes a certain input string of the length n.

Let $\vec{y}$ be the "vectorial" variable consisting of the following variables:
$S^1(j,l,m)$ where $2 \leq j \leq n^{2k}, 1 \leq l \leq n^{2k}, 1 \leq m \leq v$; $S^1(1,l,m)$ where $n+1 \leq l \leq n^{2k}, 1 \leq m \leq v$; $Q^1(j,l)$ where $1 \leq j \leq n^{2k}, 1 \leq l \leq r$; $H^1(j,l)$ where $1 \leq j \leq n^{2k}, 1 \leq l \leq n^{2k}$. The final formula $SN^n_{M,k}$ is of the form $\alpha \wedge \neg\beta$ where
$\alpha = (\forall \vec{x})(\exists \vec{y})(G^1_1 \wedge G^1_2 \wedge G^1_3 \wedge G^1_4 \wedge G^1_5 \wedge (G^1_6 \vee G^1_7))$
$\beta = G_0 \wedge (G^2_1 \wedge G^2_2 \wedge G^2_3 \wedge G^2_4 \wedge G^2_5 \wedge G^2_6) \wedge (G^3_1 \wedge G^3_2 \wedge G^3_3 \wedge G^3_4 \wedge G^3_5 \wedge G^3_7)$

It is easy to check that conditions (1) and (2) from this section hold.

4 Result

Definition 9.
A language L is said to be complete for $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ if $L \in \mathbf{NP} \cap \mathbf{co\text{-}NP}$ and any language L' in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is many-one polynomial time reducible to L.

The following lemma will be useful in the proof of Theorem 1.

Lemma 4. *(see [1]) A language L is in* $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ *if and only if L is accepted in polynomial time by a strong nondeterministic Turing machine.*

Theorem 1. *If there exists an optimal quantified propositional proof system then there exists a complete language for* $\mathbf{NP} \cap \mathbf{co\text{-}NP}$*.*

Proof. If there exists an optimal quantified propositional proof system then sequences of tautological quantified propositional formulas expressing the strong nondeterminism of Turing machines possess uniformly short proofs in this system. This is the key idea of the proof. Let s be the exponent of the polynomial n^s which is the running time bound of the machine of the optimal quantified propositional proof system. Let us define $L \subseteq \Sigma^\star$ in the following way:

$$L = \{(M, k, x, SN^n_{M,k}, Proof, 0^{n^{2k+s}}) : x \in L(M^k)\} \ \text{ where :}$$

M is a nondeterministic YES-NO Turing machine, k is a natural number, x is a string such that $n = |x|$, $SN^n_{M,k}$ is the quantified propositional formula expressing n-strong nondeterminism of M^k, $Proof$ is a proof of the formula $SN^n_{M,k}$ in the optimal quantified propositional proof system, $0^{n^{2k+s}}$ is the sequence of zeros (padding).

We claim that L is the desired $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ complete language.

(i) $L \in \mathbf{NP}$. It is assured by the following Turing machine K working in polynomial time.
  1. On input w the machine deterministically checks whether w is in the form $(M, k, x, SN^n_{M,k}, Proof, 0^{n^{2k+s}})$. If so then K comes to point 2, otherwise K rejects w.
  2. K runs M^k on input x. If M^k accepts x then K accepts w.

(ii) $L \in \mathbf{co\text{-}NP}$. It is assured by the following Turing machine T working in polynomial time.
  1. T deterministically checks whether an input w is in the form specified by the definition of L. If so then T comes to point 2, otherwise T halts in an acccepting state.
  2. T runs M^k on input x. If M^k rejects x then T accepts w.

(iii) L is a complete language for $\mathbf{NP} \cap \mathbf{co\text{-}NP}$. Let L' be any language in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$. Let M be a strong nondeterministic Turing machine working in polynomial time and accepting L'. Fix k natural and sufficiently large. Then M^k is a strong nondeterministic YES-NO Turing machine and M^k accepts L'. We claim that the function $f : \Sigma^\star \longrightarrow \Sigma^\star$ defined by

$$f(x) = (M, k, x, SN^n_{M,k}, Proof, 0^{n^{2k+s}})$$

is the polynomial time reduction of L' to L. The natural number n from the definition of f is the length of x. The string $Proof$ is a proof of the formula $SN^n_{M,k}$ in the optimal quantified propositional proof system.

Elements of the set $\{SN^1_{M,k}, SN^2_{M,k}, SN^3_{M,k}, SN^4_{M,k}, \ldots\}$ can be constructed in polynomial time (in n) so there exists a quantified propositional proof system in which 0^n is the proof of the tautological quantified propositional formula $SN^n_{M,k}$, where $n = 1, 2, 3, \ldots$ From the definition of the optimal quantified propositional proof system it follows that there exists a deterministic Turing machine working in polynomial time which on input 0^n constructs the proof of the tautological formula $SN^n_{M,k}$ in the optimal quantified propositional proof system. Hence, f is polynomial time computable.

References

1. Balcazar J.L., Díaz J. and Gabarró J. (1995) *Structural complexity I,* Springer.
2. Cook S.A. (1971) *The complexity of theorem proving procedures,* in: *Proc. 3rd ACM Symposium on Theory of Computing,* pp. 151–158.
3. Cook S.A. and Reckhow R.A. (1979) *The relative efficiency of propositional proof systems,* J. Symbolic Logic, 44, pp. 36–50.
4. Hartmanis J., Immerman N. (1985) *On complete problems for* **NP** $\cap$ **co-NP***,* in: *Proc. 12th Internat. Coll. on Automata, Languages and Programming,* LNCS 194, pp. 250–259, Springer.
5. Hopcroft J. and Ullman J. (1979) *Introduction to Automata Theory, Languages and Computation,* Addison-Wesley.
6. Kowalczyk W. (1984) *Some connections between presentability of complexity classes and the power of formal systems of reasoning,* in: *Proc. Mathematical Foundations of Computer Science,* Lecture Notes in Computer Science 176, pp. 364–369, Springer.
7. Krajiček J. (1995) *Bounded Arithmetic, Propositional Logic, and Complexity Theory* Cambridge University Press.
8. Krajiček J. and Pudlák P. (1990) *Quantified propositional calculi and fragments of bounded arithmetic,* Zeitschr. f. math. Logik und Grundlagen d. Math. 36, pp.29–46.
9. Krajiček J. and Pudlák P. (1989) *Propositional proof systems, the consistency of first order theories and the complexity of computations,* J. Symbolic Logic, 54, pp. 1063–1079.
10. Sipser M. (1982) *On relativizations and the existence of complete sets,* in: *Proc. Internat. Coll. on Automata, Languages and Programming,* LNCS 140, pp. 523–531, Springer.
11. Stockmeyer L. (1977) *The polynomial-time hierarchy,* Theor. Comp. Sci. 3, pp. 1–22.
12. Urquhart A. (1995) *The complexity of propositional proofs,* Bull. Symb. Log. 1, pp. 425–467.