ON THE CONNECTION BETWEEN THE PROBLEM OF THE EXISTENCE OF AN OPTIMAL PROPOSITIONAL PROOF SYSTEM AND THE STRUCTURE OF EASY SUBSETS OF TAUT

Zenon Sadowski
Institute of Mathematics, University of Warsaw,
Bialystok Division,
15-275 Bialystok , ul. Akademicka 2, Poland

A b s t r a c t

In this paper we develope a connection between optimal propositional proof systems and structural complexity theory - specifically there exists an optimal propositional proof system if and only if there is a suitable recursive presentation of the class of all easy (polynomial time recognizable) subsets of TAUT. As a corollary we have obtained the result that if there does not exist an optimal propositional proof system, then for every theory T there exists an easy subset of TAUT which is not T-provably easy.

1. Introduction

Research into propositional proof systems, initiated by paper [5], has been motivated by the open problem **NP** = co-**NP**?.In the course of this research the problem of the existence of an optimal propositional proof system has arisen (cf.[7]).

The inspiration to write this paper comes from two sources. The first is paper paper [7] by Krajíček and Pudlák, the second is paper [6] by Kowalczyk.Kowalczyk`s paper seems, at first glance , distant from the subject of the optimal propositional proof system at it shows that **NP**∩co-**NP** has complete sets if and only if **NP**∩co-**NP** is presentable in a suitable way.

The notion of recursive presentability of a class of sets appears useful in the problem we are interested in. It turns out, that there exists an optimal propositional proof system if and only if easy (polynomial time recognizable) subsets of TAUT are recursively presentable in the sense given in Section 4. This result reveals, and it was the main aim of the paper, a connection between two different areas of research: propositional proof systems and structural complexity theory.

The paper is organized as follows.Basic definitions are presented in Section 2. In Section 3 we give a precise definition of a family of propositional formulas which will be used in the proofs of our main results. In Section 4 our main results are stated and proved. Section 5 discusses corollaries arising from the main results of the paper. For example, it is shown that if there does not exist an optimal propositional proof system, then for every sound and axiomatizable formal system T there exists an easy subset of TAUT which is not T-provably easy. In the last section we introduce the notion of a relatively optimal propositional proof system. The introduction of this notion is motivated by the extended resolution simulation theorem proved by S.A.Cook in [3].

2. Preliminaries

The symbol Σ denotes, throughout the paper, a certain fixed finite alphabet. This alphabet contains 0 and 1, a certain adequate set of propositional connectives and some auxiliary symbols. Strings describing Turing machines, computations of Turing machines and propositional formulas considered in this paper are taken over Σ . The set of all strings over Σ is denoted by Σ^{*}.

Our model of computation is the multitape (deterministic or nondeterministic) Turing machine. Two versions of Turing machines are considered in the paper: the Turing machine acceptor and the Turing machine transducer. We will consider also polynomial time clocked (deterministic or nondeterministic) Turing machines (for exact definitions see [1]).

If $A \subset \Sigma^{*}$, then $\overline{A} = \Sigma^{*} \setminus A$. If M is a Turing machine, then L(M) denotes the set of strings accepted by M.

For any deterministic Turing machine and an input w by Time(M,w) we denote the running time of M on w. If M halts on input w, then Time(M,w) is the number of steps in the (unique) computation of M on w. If M does not halt on w, then Time(M,w) is undefined. If there exists a polynomial p such that for every w it holds, Time(M,w) $\leq$ p(|w|), we say that M runs in polynomial time.

In case, when M is a nondeterministic Turing machine Time(M,w) is defined only for w accepted by M and it denotes the number of steps in the shortest accepting computation of M on w. If there exists a polynomial p such that for every accepted w it holds, Time(M,w) $\leq$ p(|w|), we say that M runs in polynomial time. Obviously every deterministic (nondeterministic) polynomial time clocked Turing machine runs in polynomial time.

The following classes of sets (languages) are considered in the paper: **P**= { L(M): M is deterministic and runs in polynomial time } , **NP**= { L(M): M is nondeterministic and runs in polynomial time } , co-**NP**= { $\overline{L}$: L$\in$**NP** }.

The symbol TAUT denotes throughout the paper the set of all propositional tautologies over a fixed adequate set of connectives. For a string x, $|x|$ denotes the length of x .

Definition 1. (Cook, Reckhow [5]) A propositional proof system is a function $F: \Sigma^* \xrightarrow{onto}$ TAUT 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 2. (Cook, Reckhow) A propositional proof system is polynomially bounded if and only if there exists a polynomial $p(n)$ such that every tautology α has a proof of length no more than $p(|\alpha|)$ in this system.

The connection between propositional proof systems and the problem **NP** = co-**NP**? is described by the following fact, due to Cook, Reckhow [5].

Fact 1. NP = co-**NP** if and only if there exists a polynomially bounded propositional proof system.

In the set of all propositional proof systems J. Krajíček and P. Pudlák have introduced the quasiordering $\leq$.

Definition 3. For two propositional proof systems P, Q we write $P \leq Q$ if and only if there exists a polynomial p such that for every tautology α , if α has a proof of length n in Q , then α has a proof of length $\leq p(n)$ in P.

Earlier in paper [5] S.Cook and R.Reckhow introduced the notion of polynomial simulation of one system by another.

Definition 4. Propositional proof system P polynomially simulates propositional proof system Q if there exists polynomial time computable function $f: \Sigma^* \rightarrow \Sigma^*$ such that for every w , if w is a proof of α in Q , then f(w) is a proof of α in P .

Obviously if P polynomially simulates Q , then $P \leq Q$. In paper [7] the authors have introduced the notion of an optimal propositional proof system. This notion has been introduced in two versions.

Definition 5. A propositional proof system P is optimal in the first sense if for every other propositional proof system Q , it holds $P \leq Q$.

A propositional proof system P is optimal in the second sense if for every other propositional proof system Q, the system P polynomially simulates Q.

J. Krajíček and P. Pudlák have formulated in paper [7] the following problem.

Problem. Whether there exists a propositional proof system which is optimal in the first sense (in the second sense)?

The connection between this problem and the problem **NP** = co-**NP**? is described by Fact 2.

Fact 2. If an optimal propositional proof system in the first sense (in the second sense) exists , then **NP** = co-**NP** if and only if this system is polynomially bounded.

3. Formulas expressing the soundness of Turing machines.

This section will have a technical character. Propositional formulas constructed in this section will be used in the next section in the proofs of theorems 1 and 2.

For any transducer N we will denote by f_N the function computed by N. ($f_N : \Sigma^* \to \Sigma^*$)

Definition 6. A Turing machine transducer N is called sound if f_N maps Σ^* into TAUT ($f_N : \Sigma^* \to$ TAUT)

Definition 7. A Turing machine acceptor K is called sound if $L(K) \subset$ TAUT.

To any polynomial time clocked transducer M we will assign the set $A_M = \{ Sound_M^1 , Sound_M^2 , Sound_M^3 ,... \}$ of propositional formulas such that:

(1) $Sound_M^n$ can be constructed in time polynomial in n. It means, that for the set A_M there exists a transducer working in polynomial time which on input O^n the sequence of n zeros) constructs the formula $Sound_M^n$, where n = 1,2,3,...

(2) $Sound_M^n$ is a propositional tautology if and only if for every input of length n , the machine M outputs a propositional tautology.

So , for any polynomial time clocked transducer M , it holds: M is sound if and only if $A_M \subset$ TAUT.

By TR we will denote the fixed deterministic polynomial time Turing machine which accepts the pair (1) a propositional formula and (2) a truth assignment if and only if the formula is satisfied by this assignment.

Let M be any polynomial time clocked transducer. Let $n^i + i$ be its polynomial time bound and let k be the number of its worktapes. Our description of the formulas $Sound_M^n$ (where n = 1,2,3,...) will proceed in two steps. In the first step we construct the deterministic polynomial time acceptor M` which will be used later in the second step of our description.

The machine M` is constructed from M and TR as their superposition. It has two input tapes.The content of the first one (a string w_1) is treated as input to M. The content of the second one (a string w_2) is later,in the course of actions,treated as a truth assignment to propositional variables. On inputs w_1 and w_2 the machine M` first runs the transducer M on input $w_1$ and after M has finished its work, M` runs TR on inputs w_3 and w_2 , where w_3 is the output of M .If w_3 is not a propositional formula then TR halts in NO state.If w_3 is a propositional formula , say α , the machine TR treats the i-th bit of w_2 as the truth value assigned to the i-th, in order of appearance, propositional variable from α . TR halts in YES state if and only if w_2 satisfies α . We may assume that M` has k + 3 tapes and works in time $p(n) = n^{i+3}$.

Fix n natural. The second step of our construction of the formula $Sound_M^n$ is adapted from [4].

Let $\{ q_1 , q_2 , ... , q_r \}$ denote the set of states of M` and let $\{ \sigma_1 , \sigma_2 , \sigma_3 ,...,\sigma_v\}$ denote the set of its tape symbols. Now we mention the propositional variables occuring in the formula $Sound_M^n$ with their intended interpretations.

$S^l(i,j,m) = 1$ if and only if at the i-th step the j-th cell of the l-th tape of M` contains the symbol σ_m

,where $1 \le i \le p(n)$, $1 \le j \le p(n)$, $1 \le m \le v$, $1 \le l \le k + 3$.

$Q(i,l) = 1$ if and only if at the i-th step M` is at state q_l , where $1 \le i \le p(n)$, $1 \le l \le r$.

$H^l(i,j) = 1$ if and only if at the i-th step the j-th cell of the l-th tape is currently scanned by M`, where $1 \le i \le p(n)$, $1 \le j \le p(n)$, $1 \le l \le k + 3$.

Using these variables we construct propositional formulas G_1 , G_2 , G_3 , G_4 , G_5 , G_6 which express the following conditions:

G_1 : At any step M` is at a unique state.

G_2 : At any step every head of M` is scanning a unique cell of its tape.

G_3 : At any step every cell of every tape of M` contains a unique symbol.

G_4 : Every configuration of M` follows from the previous one by a legal move of M`.

G_5 : At the initial step M` is at the initial state and every head of M` is scanning the first cell of its tape. The first n cells of the first input tape do not contain blank , the remaining p(n)-n cells are blank. The first p(n) cells of the second input tape contain only the symbols 0 or 1. The first p(n) cells of each remaining tape are blank.

G_6 : Within p(n) steps M` reaches YES state.

The final formula $Sound^n_M$ is of the form $G_1 \wedge G_2 \wedge G_3 \wedge G_4 \wedge G_5 \rightarrow G_6$. It is easy to check that conditions (1) and (2) from this section hold.

4. Main result

A class of sets is recursively presentable if there exists an effective enumeration of devices for recognizing all and only members of this class (cf.[8]). In this paper we use the notions, recursive P-presentation and recursive NP-presentation which are mutations of the notion of recursive presentability.

Definition 8. By an easy subset of TAUT we mean a set A such that $A \subset$ TAUT and $A \in \mathbf{P}$ (A is polynomial time recognizable).

Definition 9. A nondeterministic Turing machine M is said to be a nondeterministic optimal algorithm for TAUT if M accepts TAUT and for every nondeterministic Turing machine M\` which accepts TAUT there exists a polynomial p such, that for every tautology α

$\mathrm{Time}(M,\alpha) \leq p(\mathrm{Time}(M`,\alpha))$.

Theorem 1. Statements (i) - (iii) are equivalent.

(i) There exists an optimal propositional proof system in the first sense.
(ii) There exists an optimal nondeterministic algorithm for TAUT.
(iii) The class of all easy subsets of TAUT possesses a recursive NP - presentation.

By the statement (iii) we mean: there exists a recursively enumerable list of nondeterministic polynomial time clocked acceptors K_1 , K_2 , K_3 ,... such that

(1) $L(K_i) \subset$ TAUT for every i
(2) For every $A \subset$ TAUT such that $A \in \mathbf{P}$ there exists i such that $A = L(K_i)$

Proof. (i)$\rightarrow$(ii) Let Opt denote an optimal propositional proof system in the first sense. A nondeterministic Turing machine M which is an optimal nondeterministic algorithm for TAUT can be described in the following way. On an input w , the machine M guesses a string v and then verifies whether v is a proof of w in the system Opt. If successful M halts in an accepting state. It can be easily checked that M accepts TAUT and for any nondeterministic Turing machine K accepting TAUT there exists a polynomial q such that for every tautology α it holds, $\mathrm{Time}(M,\alpha) \leq q(\mathrm{Time}(K,\alpha))$.

(ii)$\rightarrow$(iii) Let M be an optimal nondeterministic algorithm for TAUT. A recursive NP-presentation of all easy subsets of TAUT we will define in two steps. In the first step we define a recursively enumerable list of nondeterministic Turing machines F_1, F_2, F_3,... The machine F_i operating like the machine M accepts a string w if and only if M accepts w in no more than n^i steps, where $n = |w|$. The sequence F_1, F_2, F_3,... of nondeterministic Turing machines running in polynomial time possesses the properties (1) and (2)

(1) For every i it holds $L(F_i) \subset \mathrm{TAUT}$

(2) For every easy subset A of TAUT there exists i such that $A \subset L(F_i)$

Let M_1, M_2, M_3,... be a standard enumeration of all nondeterministic multitape Turing machines and N_1, N_2, N_3, ... be a standard enumeration of all nondeterministic polynomial time clocked Turing machines. Let the symbol $\langle .,. \rangle$ denote the standard pairing funkction on natural natural numbers. The machine N_j , where $j=\langle i,k \rangle$, is obtained by attaching to the machine M_i a standard clock which stops its computation after $n^k + k$ steps, where n is the length of the input.

In the second step of our construction of a recursive NP-presentation of all easy subsets of TAUT we define the new recursively enumerable list S_1, S_2, S_3,... of nondeterministic Turing machines working in polynomial time.The machine S_n , where $n=\langle i,j \rangle$, runs in parallel the machine K_i and the machine N_j. It accepts a string w if and only if K_i accepts w and N_j accepts w.

Finally, uniformly attaching standard polynomial clocks to the family $S_1, S_2, S_3, \ldots$ we obtain a recursively enumerable list of nondeterministic polynomial time clocked acceptors K_1, K_2, $K_3, \ldots$ The machine K_i, where $i = \langle j,k \rangle$, is obtained by attaching the shut-off clock $n^k + k$ to the machine S_j.(n is the length of the input).

Now we will prove that the sequence K_1, K_2, $K_3, \ldots$ provides a recursive NP-presentation of all easy subsets of TAUT. Let A be any fixed, easy subset of TAUT. The set A is accepted by some polynomial time deterministic Turing machine, so A is accepted by some polynomial time nondeterministic Turing machine. From the definition of polynomial time clocked acceptors we conclude that there exists k such that A is accepted by the machine N_k.Let M_0 denote the "brute force" algorithm for TAUT. Let L be the machine which chooses in a nondeterministic manner N_k or M_0 and then runs it. The machine L accepts TAUT and for every $\alpha \in A$ it holds, $\mathrm{Time}(L,\alpha) \leq |\alpha|^k + k$. From the definition of an optimal nondeterministic algorithm for TAUT and from the definition of the sequence F_1, F_2, $F_3, \ldots$ we conclude that there exists j such that $A \subset L(F_j)$. It is clear from the definition of the sequence S_1, S_2, $S_3, \ldots$ that A is accepted by the machine $S_{\langle j,k \rangle}$. It follows easily from the definition of the sequence K_1, K_2, $K_3, \ldots$ that there exists i such that $A = L(K_i)$.

(iii)$\rightarrow$(i) Let G be the machine generating the codes of the machines from the sequence $\{K_i\}_{i=1}^{\infty}$. We define the machine Opt which is the machine of the optimal propositional proof system. Strings of input of the machine Opt are divided into two groups: good and bad. A good string is of the form $w_1\$w_2\$w_3\$w_4$, where

w_1 – is a code of a polynomial time clocked transducer.

w_2 – is a code of a computation of the machine w_1.It follows from the next conditions that this computation must produce some propositional formula α as its input.

w_3 – is a code of a computation of the machine G. In the last step of this computation the output tape of the machine G contains a code of some machine K_i from the sequence $\{K_j\}_{j=1}^{\infty}$.

w_4 – is a code of a computation of the machine K_i accepting the formula $Sound_{w_1}^{n}$. The subscript n in this formula is equal to the length of the string on input of the machine w_1 in the first step of the computation w_2.

These conditions guarantee firstly, that the machine w_1 for any input of length n outputs a propositional tautology and secondly, that α is produced by a computation of w_1 with an input of length n. Hence α is a propositional tautology.

Given a good string $w_1\$w_2\$w_3\$w_4$ as input, the machine Opt outputs the tautology α from the substring w_2. Given a bad word as input the machine Opt outputs the standard tautology $p_1 \vee \sim p_1$. It is obvious that Opt always outputs a tautology.

Let P: $\Sigma^* \rightarrow$ TAUT be any propositional proof system and let α be a propositional tautology possessing a proof of length n in the system P. Let M be a polynomial time clocked transducer such that $P = f_M$. It follows from (iii) that there exists a machine K_i such that $L(K_i) = A_M$ (For exact definitions of symbols f_M and A_M see Section 3).

We define:

w_1 – a code of the machine M.

w_2 – a code of a computation of M such that the output of this computation is the tautology α and the input of this computation is a proof of α in P of length n.

w_3 – a code of acomputation of the machine G such that a code of K_i is the output produced by this computation.

w_4 – a code of the computation of K_i accepting $Sound_M^n$.

Then $|w_1|= c_1$ and $|w_3|= c_2$, where c_1 and c_2 are constants. Because M and K_i are polynomial time clocked there exists polynomials p and q such that $|w_2|\leq p(n)$ and $|w_4|\leq q(n)$. The polynomial p and the constant c_1 depend only on M. The polynomial q and the constant c_2 depend only on K_i, which is fixed and connected with M. This establishes the desired conclusion that Opt $\leq$ P.

Definition 10. A deterministic Turing machine M is said to be an almost optimal deterministic algorithm for TAUT if M accepts TAUT and for every deterministic Turing machine M\` which accepts TAUT there exists a polynomial p such that for every tautology $\alpha$ $\mathrm{Time}(M,\alpha)\leq p(\mathrm{Time}(M`,\alpha))$.

The equivalence (i)-(ii) in the next theorem is restated from [7] in order to emphasize the symetry between Theorem 1 and Theorem 2.

Theorem 2. Statements (i)-(iii) are equivalent.

(i) There exists an optimal propositional proof system in the second sense.

(ii) There exists an almost optimal deterministic algorithm for TAUT.

(iii) The class of all easy subsets of TAUT possesses a recursive P-presentation.

By the statement (iii) we mean: there exists a recursively enumerable list of deterministic polynomial time clocked acceptors $K_1,K_2,K_3,...$ such that

(1) $L(K_i)\subset$ TAUT for every i

(2) For every $A \subset$ TAUT such that $A\in\mathbf{P}$ there exists i such that $A = L(K_i)$

Proof.(i) $\to$ (ii) See [7].
(ii) $\to$ (iii) Let M be an almost optimal deterministic algorithm for TAUT. A recursive P-presentation of all easy subsets of TAUT we will define in two steps. In the first step we define a recursively enumerable list of deterministic Turing machines $F_1, F_2, F_3, \ldots$ The machine F_i operating like the machine M accepts (rejects) a string w if and only if M accepts (rejects) w in no more than n^i steps, where $n = |w|$.The sequence $F_1, F_2, F_3, \ldots$ of deterministic polynomial time Turing machines possesses the properties (1) and (2)

(1) For every i it holds $L(F_i) \subset$ TAUT
(2) For every easy subset A of TAUT there exists i such that $A \subset L(F_i)$

Let $D_1, D_2, D_3, \ldots$ be a standard enumeration of all deterministic polynomial time clocked Turing machines. In the second step of our construction we define the new recursively enumerable list $S_1, S_2, S_3, \ldots$ of deterministic Turing machines working in polynomial time. The machine S_n ,where $n = \langle i, j \rangle$, runs in parallel the machine F_i and the machine D_j . It accepts a string w if and only if F_i accepts w and D_j accepts w. It rejects a string w if and only if F_i rejects w or D_j rejects w.

Finally, uniformly attaching standard polynomial clocks to the family $\{S_i: i=1,2,3,\ldots\}$ we obtain a recursively enumerable list of deterministic polynomial time clocked acceptors $K_1, K_2, K_3, \ldots$ The machine K_i, where $i = \langle j, k \rangle$, is obtained by attaching the shut-off clock $n^k + k$ to S_j (n is the length of the input). The sequence $K_1, K_2, K_3, \ldots$ provides a recursive P-presentation of all easy subsets of TAUT.

(iii) $\to$ (i) The proof is analogous to the proof of (iii) $\to$ (i) from Theorem 1. Let w_1, w_2, w_3, w_4 denote the same as in Theorem 1. Then $w_1\$w_2\$w_3\$w_4$ is a proof of α in the propositional proof system Opt which is optimal in the second sense. Let M be any propositional proof system and let α be any propositional tautology. We define w_1, w_2, w_3, w_4 as in Theorem 1. Since every machine K_j from the P-presentation is deterministic and runs in polynomial time, Opt polynomially simulates M.

5. Corollaries and natural questions

Let T be any formal theory whose language contains the language of arithmetic, i.e. the language $\{0,1,\leq,=,+,\cdot\}$.We will not specify T in details but only assume that T is sound and the set of all theorems of T is recursively enumerable. The notation $T - \beta$ means that a first order formula β is provable in T.

Let M be a Turing machine.By "$L(M)\subset TAUT$" we denote the first order formula which expresses the soundness of M , i.e. $\forall w \in L(M)$[w is a propositional tautology].

Definition 11. A deterministic (nondeterministic) Turing machine M is T-sound if $T -$ "$L(M)\subset TAUT$"

Definition 12. A set $A \subset TAUT$ is deterministically (nondeterministically) T-tractable if there exists a deterministic (nondeterministic) polynomial time clocked acceptor M fulfilling (1)-(2)

(1) M is T-sound
(2) $L(M)= A$

Theorem 3. If there does not exist an optimal propositional proof system in the first sense, then for every theory T there exists an easy subset of TAUT which is not nondeterministically T-tractable

Proof. Suppose, on the contrary, that all easy subsets of TAUT are nondeterministically T-tractable. The machines from the set Ω_T= {N: N is a nondeterministic polynomial time clocked acceptor which is T-sound} create a recursive NP-presentation of the class of all easy subsets of TAUT. By Theorem 1 there exists an optimal propositional proof system in the first sense.Contradiction.

Theorem 4. The conditions (i)-(ii) are equivalent.

(i) There exists an optimal propositional proof system in the first sense.
(ii) There exists a theory T such that every easy subset of TAUT is nondeterministically T-tractable.

Proof.(i)$\rightarrow$(ii) Let M_1, M_2, M_3,... be the sequence of nondeterministic polynomial time clocked acceptors creating NP-presentation of all easy subsets of TAUT.The sentences {"$L(M_i)\subset$ TAUT" : i=1,2,3,...} are axioms of the desired theory T.

Theorem 5. If there does not exist an optimal propositional proof system in the second sense, then for every theory T there exists an easy subset of TAUT which is not deterministically T-tractable.

Theorem 6. The conditions (iii)-(iv) are equivalent.

(iii) There exists an optimal propositional proof system in the second sense.
(iv) There exists a theory T such that every easy subset of TAUT is deterministically T-tractable.

In connection with theorems 1 and 2 the natural question, whether the set of all sound deterministic (nondeterministic) polynomial time clocked acceptors creates the desired P-presentation (NP-presentation), occurs.The negative answer to this question is provided by Theorem 7.

Theorem 7. The set of all sound deterministic (nondeterministic) polynomial time clocked Turing acceptors is not recursively enumerable.

This theorem is an immediate consequence of Rice`s theorem (cf.[9]).

6. Relatively optimal propositional proof systems

Let T be a fixed theory satysfying the assumptions from the previous section. The question about the existence of an optimal propositional proof system we can relativize to theory T.

For any propositional proof system M we can construct the first order formula $SOUND_M(x)$ satisfying the condition (v).

(v) For every n, $SOUND_M(\bar{n})$ states that each propositional formula α which possesses a proof in M of length $\leq n$, is a propositional tautology.

The symbol $\bar{n}$ denotes the numeral associated with natural number n.

Definition 13. A propositional proof system M is T-provably sound if there exists a polynomial p such that for every n there exists a proof of $SOUND_M(\bar{n})$ in theory T of length $\leq p(n)$.

We additionally assume that T is sufficiently rich. This means that there exists at least one propositional proof system which is T-provably sound.

Definition 14. A propositional proof system Opt_T is T-provably optimal if for every T-provably sound propositional proof system Q, it holds $Opt_T \leq Q$.

Theorem 8. For every sufficiently rich theory T there exists a T-provably optimal propositional proof system.

Proof.Every proof of a propositional tautology α in the system Opt_T is of the form $w_1\$w_2\$w_3$,where

w_1 – is a code of a polynomial time clocked transducer

w_2 – is a code of a computation of the machine w_1 which produces α as its output

w_3 – is a proof of the formula $SOUND_{w_1}(\bar{n})$ in the theory T; n is the length of the string on input of the machine w_1 in the first step of the computation w_2.

It can be easily checked that $Opt_T \leq Q$ for every T-provably sound propositional proof system Q.

When T is taken to be PV, as defined by S.A.Cook [3], or to be S_2^1 as introduced by S.Buss [2], then extended resolution is a T-provably optimal propositional proof system. The proof of Theorem 8 contains recursion theoretic aspects of the proof of Cook`s ER simulation theorem [3].

References

[1] J.L.Balcazar,J.Díaz and J.Gabarró, Structural Complexity I (Springer-Verlang,Berlin,1995).

[2] S.Buss, Bounded Arithmetic,volume 3 of Studies in Proof Theory (Bibliopolis,1986).

[3] S.A.Cook, Feasibly constructive proofs and the propositional calculus, in: Proc.7th Annual ACM Symposium on Theory of Computing (1975) 83-97.

[4] S.A.Cook, The complexity of theorem proving procedures, in: Proc.3rd ACM Symposium on Theory of Computing (1971) 151-158.

[5] S.A.Cook and R.A.Reckhov, The relative efficiency of propositional proof systems, J.Symb.Log.44(1979) 36-50.

[6] W.Kowalczyk, 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(Springer,Berlin,1988) 364-369.

[7] J.Krajíček and P.Pudlák, Propositional proof systems,the consistency of first order theories and the complexity of computations, J.Symb.Log.54(1989) 1063-1079.
[8] L.H.Landweber,R.J.Lipton and E.L.Robertson, On the structure of sets in NP and other complexity classes, Theoret.Comput.Sci.15(1981) 181-200.
[9] H.G.Rice, Classes of recursively enumerable sets and their decision problems, Trans.Amer.Math.Soc. 74(1953) 358-366.