FRAGMENTS OF BOUNDED ARITHMETIC
AND BOUNDED QUERY CLASSES

JAN KRAJÍČEK

ABSTRACT. We characterize functions and predicates $\Sigma_{i+1}^b$-definable in $S^i_2$. In particular, predicates $\Sigma_{i+1}^b$-definable in $S^i_2$ are precisely those in bounded query class $P^{\Sigma^p_i}[O(\log n)]$ (which equals to Log Space $^{\Sigma^p_i}$ by [B-H,W]). This implies that $S^i_2 \neq T^i_2$ unless $P^{\Sigma^p_i}[O(\log n)] = \Delta^p_{i+1}$. Further we construct oracle $A$ such that for all $i \geq 1$: $P^{\Sigma^p_i(A)}[O(\log n)] \neq \Delta^p_{i+1}(A)$. It follows that $S^i_2(\alpha) \neq T^i_2(\alpha)$ for all $i \geq 1$. Techniques used come from proof theory and boolean complexity.

Bounded arithmetic, a subtheory of Peano arithmetic with induction axioms only for bounded formulas, was introduced in [Pa]. Later several other systems were considered, varying in their language or underlying logic, or restricting induction axioms even to a subclass of bounded formulas. Bounded arithmetic is relevant to topics like nonstandard models of arithmetic, interpretability of theories, computational complexity and complexity of propositional logic$^1$.

Fragments of bounded arithmetic in which we are interested here are theories $S^i_2$ and $T^i_2$, subsystems of theory $S_2$ introduced in [B1]. The language of these theories consists of symbols: 0, 1, +, ·, ≤, =, $[\frac{x}{2}]$, $|x|$ ($= \lceil \log_2(x+1) \rceil$) and $x\#y$ ($\approx 2^{|x| \cdot |y|}$). Both theories contain 32 universal axioms BASIC defining most elementary properties of functions represented in the language. $T^i_2$ is axiomatized over BASIC by an induction axiom scheme IND:

$$A(0) \& \forall x(A(x) \rightarrow A(x + 1)) \rightarrow \forall xA(x)$$

restricted to bounded $\Sigma^b_i$-formulas $A$, while in $S^i_2$ the induction axioms are replaced by seemingly weaker scheme LIND:

$$A(0) \& Ax(A(x) \rightarrow A(x + 1)) \rightarrow \forall xA(|x|)$$

restricted also to $\Sigma^b_i$-formulas.

It holds that $S^i_2 \subseteq T^i_2 \subseteq S^{i+1}_2$ for $i \geq 1$ and $S_2 = \cup S^i_2 = \cup T^i_2$. All $S^i_2$ and $T^i_2$ are finitely axiomatizable and thus the important open question whether $S_2$ is finitely axiomatizable reduces to a question whether $S_2 = S^i_2$ or $S_2 = T^i_2$ for

*Received by the editors February 21, 1991.
1980 Mathematics Subject Classification (1985 Revision). Primary 03F30; Secondary 03F05, 03F50.

$^1$A survey text covering most parts of bounded arithmetic (and containing also bibliographical and historical information) is in monograph [H-P].

©1993 American Mathematical Society
0002-9947/93 $1.00 + .25 per page

587
some \( i \geq 1 \). This naturally leads to attempts to show that actually \( S_2^i \neq T_2^i \) and \( T_2^i \neq S_2^{i+1} \) for all \( i \geq 1 \).

The relationship between \( T_2^i \) and \( S_2^{i+1} \) is better understood than the relationship between \( S_2^i \) and \( T_2^i \). In [B2] it is proved that \( S_2^{i+1} \) is \( \forall \Sigma_{i+1}^b \)-conservative over \( T_2^i \) while in [K-P-T] it was shown that \( T_2^i \neq S_2^{i+1} \) provided that \( \Sigma_{i+2}^b \neq \Pi_{i+2}^p \). As \( S_2^{i+1} \) can be \( \forall \Sigma_{i+2}^b \)-axiomatized these two results seem to furnish rather complete understandings of the relation of \( T_2^i \) to \( S_2^{i+1} \) (provided that the polynomial-time hierarchy \( PH \) does not collapse).

About the relation of \( S_2^i \) to \( T_2^i \) considerably less is known. Conservativity of \( T_2^i \) over \( S_2^i \) was in [K-P and K-T] equivalently restated as certain combinatorial proof-theoretic problems but neither of them was solved. Problem whether \( S_2^i \) and \( T_2^j \) are equivalent was in [P] reduced to a problem in complexity theory but for rather unusual mode of computation: interactive computations with counterexamples, see also [K] for another presentation. A hierarchy theorem for such computations was proved in [K-P-S] but unfortunately not strong enough to separate \( S_2^i \) from \( T_2^j \). Also a relation of this problem about counterexample computations to standard conjectures in complexity theory is unknown at present.

The main objective of this paper is to show that \( S_2^1 = T_2^1 \) would imply that \( P^{E_T}[O(\log n)] = \Delta^p_{i+1} \). Here \( P^{E_T}[O(\log n)] \) is (a straightforward generalization of) a class introduced in [Kre], cf. [W]. It consists of those languages recognizable by a polynomial-time oracle machine quering a \( \Sigma^p_i \)-oracle at most \( O(\log n) \)-times, \( n \) the length of an input. \( \Delta^p_{i+1} \) is the familiar class of languages recognizable by polynomial-time oracle machines quering a \( \Sigma^p_i \)-oracle with no restriction (other than the obvious polynomial one) on the number of queries.

The problem whether \( P^{E_T}[O(\log n)] = \Delta^p_2 \) seems to be quite extensively studied, cf. [Kre, B-H, and W]; the case \( i > 1 \) was considered in [W]. In particular, the class \( P^{E_T}[O(\log n)] \) was in [B-H and W] equivalently characterized in many different ways, most notably as the class of predicates log-space Turing reducible or truth-table reducible (via formulas or circuits) to SAT, or as predicates computable by polynomial-time \( \Sigma^p_i \)-oracle machines which are allowed only one round of parallel queries, or as the class of predicates definable by \( \Sigma^b_2 \cap \Pi^b_2 \)-formulas (i.e. formulas whose syntactic form puts them simultaneously to \( \Sigma^b_2 \) and \( \Pi^b_2 \)).

The arguments from [B-H and W] readily generalize to any oracle of the form \( \Sigma^p_i (A) \) in place of \( \Sigma^p_i \), and in particular to \( \Sigma^p_i (A) \). This gives completely analogical characterizations of the classes \( P^{E_T}[O(\log n)] \).

Although the conjecture that \( P^{E_T}[O(\log n)] \neq \Delta^p_2 \) appears to be closer to standard conjectures about \( PH \) than is the conjecture about counterexample computations needed for separation of \( S_2^1 \) from \( T_2^1 \) (see [P and K-P-S]), no such reduction is in fact known. In particular, it is an open problem whether any \( P^{E_T}[O(\log n)] = \Delta^p_{i+1} \) would imply the collapse of \( PH \). (In [Kre] it is observed—for \( i = 1 \)—that such an equality for classes of function instead of predicates would imply \( P = NP \), and \( \Delta^p_i = \Sigma^p_i \) for general \( i \geq 1 \). Unfortunately, this does not seem to be relevant at all to the case with predicates.)

However, we construct oracle \( A \) separating \( P^{E_T}[O(\log n)] \) from \( \Delta^p_{i+1}(A) \)
for all $i \geq 1$. The existence of such an oracle implies that theories $S_2^i(\alpha)$ and $T_2^i(\alpha)$ are different for all $i \geq 1$. Such oracle for $i = 1$ was constructed in \[B-H\]. That $S_2^1(\alpha) \neq T_2^1(\alpha)$ and $S_2^2(\alpha) \neq T_2^2(\alpha)$ was already proved by other means in \[P and K\], and by Buss (unpublished).

1. Modified computations with oracles

We first give the definitions for the case of $\Sigma^p_1$-oracles which generalizes easily to $\Sigma^p_2$-oracles.

(1.1) Let $M$ be a polynomial-time oracle machine and $A(u) = \exists v B(u, v)$ a $\Sigma^p_1$-oracle, where $B$ is a polynomial-time predicate. We shall always assume that a polynomial time bound is a part of the specification of $M$ and a polynomial bound to $v$, $|v| \leq |u|^k$, is a part of $B$.

An $\alpha(M, A, t(n))$-computation is a computation obtained by the following modification of $\Delta^p_2$-computations. On input $x$ of length $n$, $M$ computes querying oracle $A$ with the restriction that there are at most $t(n)$ oracle queries in the computation, but with the addition that if the oracle returns affirmative answer to a query $[A(u)]$ it also provides $M$ with a witness to it, i.e. with some $v$ such that $B(u, v)$. The witness is provided in the same computational step.

Clearly there might be more $\alpha(M, A, t(n))$-computations on a given input as the oracle might have several options to choose witnesses from.

(1.2) A function $f: \omega \rightarrow \omega$ is $\alpha(M, A, t(n))$-computable iff for any $x$ all $\alpha(M, A, t(n))$-computations on $x$ output $f(x)$. A predicate is a function assuming only values 0, 1.

(1.3) Proposition. Given machine $M$ and oracle $A$ as in (1.1), and a constant $c$, the following is provable in $S_2^1$:

“For arbitrary $x$ there exists an $\alpha(M, A, c \cdot \log(n))$-computation on $x$.”

Proof. We may assume that both $M$ and $B$ are defined by $\Delta^p_1$-formulas. Let $n^k$ be the time bound of $M$. Consider formula $\psi$:

$\psi(a, h, w) :=$

(a) “$w = (w_1, \ldots , w_t)$ is a computation of length $t \leq |a|^k$ on input $a$”, and

(b) “$h$ is a sequence $\langle (i_1, j_1), \ldots , (i_r, j_r) \rangle$ for some $r \leq c \cdot ||a||$ such that $i_1 < i_2 < \cdots < i_r \leq t$ and $j_1, \ldots , j_r = 0, 1$ (we think of $h$ as coding oracle answers in steps $i_1, \ldots , i_r$),” and

(c) “$w$ correctly follows oracle answers coded in $h$ and all oracle queries are answered in $h$”, and

(d) “whenever $[A(u)?]$ is the query in step $i_s$ ($s \leq r$) and $j_s = 1$ then $w_{j_s}$ codes a witness $v_s$ such that $B(u_s, v_s)$ is true”.

Clearly formula $\psi$ is $\Delta^p_1$ in $S_2^1$.

Claim. $S_2^1$ proves formula

“$\exists$ maximal $m = (j_1, \ldots , j_r) \exists h, w$; ” $h$ is of the form $\langle (i_1, j_1), \ldots , (i_r, j_r) \rangle \& \psi(a, h, w)”.

(Observe that maximal $m$ means the same as lexicographically maximal 0-1 sequence $(j_1, \ldots , j_r)$.)
Proof of the claim. Denote by $\Psi(a, m)$ formula

$$\exists h, w; \text{"} h \text{ is of the form } ((i_1, j_1), \ldots, (i_r, j_r))$$

where $m = (j_1, \ldots, j_r)$ and $\psi(a, h, w)$.

Clearly $\Psi$ is $\Sigma^p_1$ in $S^1_2$. As $m$ is implicitly sharply bounded:

$$m \leq 2^r \leq 2^c \cdot ||a|| \leq |a|^c,$$

the existence of maximal $m$ s.t. $\Psi(a, m)$ follows by $\Sigma^p_1$-LIND.

To conclude the proof of the proposition observe that in $h, w$ witnessing $\Psi(a, m)$ for the maximal $m$ all negative oracle-answers (and therefore all answers as the affirmative ones are witnessed) must be correct. Otherwise a 0 in $m$ could be changed to 1 leaving the earlier bits unchanged and setting the later bits to 0, and thus increasing $m$. Therefore $w$ is a wanted $\alpha(M, A, c \cdot \log(n))$-computation on $a$. □

(1.4) Remark. Analogically, $\alpha(M, A, t(n))$-computations exist for every input provably in $S^1_2 + \forall x \exists y; ||y|| > \tau(|x|)$ (such $y$'s are needed to code $h$'s).

For $t(n) = \log(n)\epsilon$ this is $S^1_3$.

(1.5) $\beta(M, A, t(n))$-computations are defined as $\alpha(M, A, t(n))$-computations with the change that a witness to a positive oracle-answer is provided only in the last query of the computation and not otherwise.

(1.6) Proposition. For any $M, A$, and $t(n)$ as in (1.1) there are machine $M'$ and $\Sigma^p_1$-oracle $A'$ such that for every input $x$ it holds: the set of outputs of $\beta(M', A', t(n) + 1)$-computations on input $x$ is nonempty and is included in the set of outputs of $\alpha(M, A, t(n))$-computations on $x$.

Proof. Machine $M'$ by binary search constructs maximal 0-1 sequence $m = (j_1, \ldots, j_r)$ such that $\Psi(x, m)$. This requires $|m| = r \leq t(n)$ queries to oracle $A_1(u) := \exists v \Psi(x, u \wedge v)$.

Having such maximal $m$, $M'$ asks $[\Psi(x, m)]$. The answer must be affirmative and a witness to it contains a correct $\alpha(M, A, t(n))$-computation $w$ on $x$, therefore also the output of $w$.

Oracle $A'$ is composed of $A_1$ and $\Psi$. □

(1.7) Corollary. If a function $f: \omega \to \omega$ is $\alpha(M, A, t(n))$-computable for some $M, A, t(n)$ as in (1.1), it is also $\beta(M', A', t(n) + 1)$-computable for some $M', A'$. □

(1.8) Proposition. The class of predicates which are $\alpha(M, A, c \cdot \log(n))$-computable for some $M, A$ as in (1.1) and $c < \omega$ equals the class $P^{\Sigma^p_1}[O(\log n)]$.

Proof. $\alpha(M, A, c \cdot \log(n))$-computability of $P^{\Sigma^p_1}[O(\log n)]$-predicates is trivial.

Assume now that predicate $P(x)$ is $\alpha(M, A, c \cdot \log(n))$-computable and so—by (1.7)—also $\beta(M', A', c \cdot \log(n) + 1)$-computable. In the computation of $M'$ change the last query—see the proof of (1.6)—to:

$$[\Psi(x, m) \& \text{"} w \text{ witnessing } \Psi(x, m) \text{ outputs } 1"]?$$

and do not require a witness to it. Clearly affirmative answer to this query is equivalent to the validity of $P(x)$. □

(1.9) Generalization to $i > 1$. Clearly all preceding definitions and propositions generalize to $i > 1$: consider $\alpha^i$- and $\beta^i$-computations which differ
from $\alpha$- and $\beta$-computations in that we allow $A$ to be a $\Sigma^0_p$-oracle. Then $B$

is required to be $\Delta^0_p$-predicate.

In particular, (1.3) generalizes to "$S^2_2$ proves that $\alpha'(M, A, c \cdot \log(n))$-computations exist on all inputs" and (1.8) gives equivalence between $P^{\Sigma^0}[O(\log n)]$ and the class of $\alpha'(M, A, c \cdot \log(n))$-computable predicates, $c < \omega$.

2. WITNESSING $S^2_2$-PROOFS

This section aims at proving the following proposition.

(2.1) **Theorem.** For $i \geq 1$, a predicate is $\Sigma^b_{i+1}$-definable in $S^2_2$ iff it belongs to class $P^{\Sigma^0}[O(\log n)]$.

**Proof.** The if-part follows from (1.3), (1.8) and (1.9). Therefore it remains only to prove the only if-part of the theorem. This is done by a witnessing type argument.

Let $\psi(x, y)$ be a $\Sigma^b_{i+1}$-formula such that for all $x < \omega$ either $\psi(x, 0)$ or

$\psi(x, 1)$ holds but not both, and assume that $S^2_2$ proves $\forall x \exists y ; \psi(x, y) \land y \leq 1$.

We want to show that the predicate $\psi(x, 1)$ is in $P^{\Sigma^0}[O(\log n)]$.

Adding possibly to the language some polynomial-time functions (coding and decoding sequences) we may assume, by cut elimination, that we have an $S^2_2$-proof $d$ of the sequent $\vdash \exists y \psi(a, y)$ in which every sequent has the form

$\Gamma_0 \rightarrow \Delta_0$ where

(i) $\Gamma_1, \Gamma_2$ are cedents of $\Sigma^b_i$- and $\Pi^b_i$-formulas,

(ii) $\Delta_1$ is a cedent:

$\exists y_1 \theta_1(b, y_1), \ldots, \exists y_r \theta_r(b, y_r)$ and $\Delta_2$ is a cedent:

$\exists z_1 \eta_1(b, z_1), \ldots, \exists z_s \eta_s(b, z_s)$, where $\theta_j$'s and $\eta_j$'s are $\Pi^b_i$-formulas and

bounds to $y_j$'s and $z_j$'s are part of $\theta_j$'s and $\eta_j$'s respectively.

We say that $u$ is a witness to $\Gamma_1, \Delta_1$ for parameters $b$ if $u$ has the form

$u = \langle b, y_1, \ldots, y_r \rangle$ and conjunction $\Box \Delta_1(b) \land \Box \theta_j(b, y_j)$ is true.

We say that $v$ is a witness to $\Gamma_2, \Delta_2$ for parameters $b$ if $v$ has the form

$v = \langle b, z_1, \ldots, z_s \rangle$ and disjunction $\bigvee \Delta_2(b) v \lor \Box \theta_j(b, y_j)$ is true.

**Claim.** For every sequent in $d$ of the above form there is a polynomial-time oracle machine $M$, a $\Sigma^0_p$-oracle $A$, and a constant $c < \omega$ such that: if $u$ is a witness of $\Gamma_1, \Delta_1$ for parameters $b$ and $v$ is an output of any $\alpha'(M, A, c \cdot \log(n))$-computation on $u$ then $v$ is a witness of $\Gamma_2, \Delta_2$ for parameters $b$.

**Proof of the claim.** The proof of the claim goes by induction on the number of sequents in $d$ above the sequent, distinguishing several cases according to the type of the inference giving the sequent. We treat only two nontrivial cases:

$\exists \leq$: left and $\Sigma^b_i$-LIND (see [B1, K], or [P] or other witnessing arguments).

$\exists \leq$: left case. We consider two subcases according to the complexity of the principal formula of the inference. If the principal formula is $\Sigma^b_i$ but not $\Sigma^b_i$ then the machine remains (essentially) the same: only a parameter becomes a bounded variable and hence a part of the witness $u$.

Assume now that a $\Sigma^b_i$-formula $\exists t \xi(b, t)$ was inferred from $\xi(b, b_0), b_0$ not among $\overline{b}$. Assume $M$ witnesses the upper sequent in the sense of the claim. Construct new machine $M'$: on input $u' = \langle b, \ldots \rangle$ it first asks a query
If the answer is negative, \( M' \) outputs 0 and stops (\( u' \) is not a witness of \( \Gamma_1, \Delta_1 \)). If the answer is affirmative then \( M' \) is also provided with a witness \( t \) to it, i.e. \( \xi(\vec{b}, t) \) is true. Then \( M' \) forms \( u := (\vec{b} \sim t, \ldots) \) and runs as \( M \) on input \( u \).

**\( \Sigma^b_i \)-LIND case.** Assume the inference is of the form

\[
\begin{align*}
\xi(b_0) & \rightarrow \xi(b_0 + 1) \\
\xi(0) & \rightarrow \xi(|t(\vec{b})|)
\end{align*}
\]

omitting the side formulas. We may also assume that \( b_0 \) is not among \( \vec{b} \). Let \( M \) be a machine witnessing the upper sequent.

Machine \( M' \) on input \( u' = (\vec{b}, \ldots) \) first computes value \( w = |t(\vec{b})| \) and asks \([\xi(w)]?\). If the answer is affirmative it outputs 0 and stops (any \( v' \) is a witness to the succedent). If the answer is negative it asks \([\xi(0)]?\). If the answer to this query is negative, it outputs 0 and stops.

In the case that the answers to \([\xi(w)]?\) and \([\xi(0)]?\) were negative resp. affirmative, \( M' \) finds by binary search \( t < w \) such that: \( \xi(t) \) holds but \( \xi(t + 1) \) does not; this takes \( \log(w) = O(\log(\log(|u'|))) = O(\log n) \) queries. Having such \( t \), \( M' \) forms \( u = (\vec{b} \sim t, \ldots) \) and runs as \( M \) on input \( u \). Any output \( v \) is a witness to the succedent of the upper sequent but as \( \xi(t + 1) \) fails it is also a witness to the succedent of the lower sequent.

This proves the claim.

Clearly, the claim together with (1.8) and (1.9) completes the proof of the theorem. \( \square \)

**Remark.** Similar witnessing theorem remains true even if \( S^i_2 \) is extended by a certain version of induction for \( \Sigma^b_{i+1} \)-formulas arising in a connection with second order bounded arithmetic, offering thus (with (1.4)) a conservation result. This will be considered elsewhere.

(2.2) **Corollary.** Let \( i \geq 1 \) and assume \( S^i_2 = T^i_2 \). Then

\[
P^{\Sigma^b_i}[O(\log n)] = \Delta^p_{i+1}.
\]

**Proof.** By [B2] every \( \Delta^p_{i+1} \)-predicate is \( \Sigma^b_{i+1} \)-definable in \( T^i_2 \). This with (2.1) implies the corollary. \( \square \)

(2.3) **Corollary.** Assume there is an oracle \( A \) such that

\[
P^{\Sigma^b(A)}[O(\log n)] \neq \Delta^p_{i+1}(A)
\]

for all \( i \geq 1 \). Then \( S^i_2(\alpha) \neq T^i_2(\alpha) \) for all \( i \geq 1 \).

**Proof.** The proof of Theorem (2.1) relativizes as does also a proof in [B2] characterizing \( \Sigma^b_{i+1} \)-definable functions of \( T^i_2 \). Therefore (2.2) relativizes too. \( \square \)

### 3. A construction of an oracle

In this section we construct oracle \( A \) separating \( P^{\Sigma^b_i}[O(\log n)] \) from \( \Delta^p_{i+1}(A) \) for all \( i \geq 1 \). For \( i = 1 \) such oracle was constructed in [B-H] and we shall later, in (3.12), make use of that construction.
(3.1) **Theorem.** There exists oracle $A$ such that for every $i \geq 1$ it holds that

$$P^\Sigma_i(A)[O(\log n)] \neq \Delta^p_{i+1}(A).$$

(3.2) The proof of the theorem occupies the rest of the paper and is summarized in (3.13). Methodologically we follow a construction of an oracle separating the levels of the polynomial hierarchy as presented in [H1], following [S]. The strategy is the following.

We define predicates $\Psi^\alpha_i(x)$ contained always in $\Delta^p_{i+1}(\alpha)$, a straightforward generalization of ODDMAXSAT problem. From a characterization of $P^\Sigma_i[O(\log n)]$ as tt-reducible to $\Sigma^p_i(\alpha)$ in [B-H, W] we deduce that containment of $\Psi^\alpha_i$ in $P^\Sigma_i[O(\log n)]$ would imply that corresponding boolean functions (deciding truth-value of $\Psi^\alpha_i(m)$ for $m$ fixed and $\alpha$ variable) are computable by boolean circuits of certain type. Utilizing a switching lemma we then show that this is impossible. (Predicates $\Psi^\alpha_i$ are defined in a way allowing a direct use of a switching lemma as formulated and proved in [HI, 2].) This will imply that all tt-reducibilities to $\Sigma^p_i(\alpha)$ can be diagonalized and alternating this diagonalization for all $i \geq 1$ will give the required oracle.

(3.3) For $i \geq 1$ define formulas

(a) $\psi_1(x, y_1) := y_1 = 0 \lor \alpha((i, x, y_1)),$
(b) $\psi_2(x, y_1) := y_1 = 0 \lor \forall y_2 < x \cdot \log(x); \alpha((i, x, y_1, y_2)),$
(c) $\psi_i(x, y_1) := y_1 = 0 \lor \forall y_2 < x \exists y_3 < x \cdots \exists y_{i-1} < x$ 

\[ Q_i y_i < \sqrt{\frac{i \cdot x \cdot \log(x)}{2}}; \alpha((i, x, y_1, \ldots, y_i)). \]

Thus $\psi_i$ is a $\Pi^b_{i-1}(\alpha)$-formula. Consider predicate

$$\Psi^\alpha_i(x) := \"\text{maximal } y_1 < x \text{ satisfying } \psi_i(x, y_1) \text{ is odd}\".$$ 

(3.4) **Lemma.** Predicate $\Psi^\alpha_i(x)$ is in $\Delta^p_{i+1}(A)$ for all $i \geq 1$ and $A \subseteq \omega$. \(\square\)

(3.5) Now we define depth $i-1$ boolean circuits $\tilde{\psi}_i(m, u)$ with input variables $x_u,y_2,\ldots,y_{i-1},t$ for every choice of $y_2, \ldots, y_{i-1} < m$ and $t < \sqrt{\frac{i \cdot m \cdot \log(m)}{2}}$ computing the truth value of $\psi_i(m, u)$ for every $A \subseteq \omega$ under evaluation of variables 

$$x_u, y_2, \ldots, y_{i-1}, t = 1 \iff (i, m, u, y_2, \ldots, y_{i-1}, t) \in A.$$ 

Precise definition of circuits $\tilde{\psi}_i(m, u)$ is by induction

(i) circuit $G_0(u)$ is just variable $x_u$,
(ii) circuit $G_{k+1}(u)$ is conjunction $\bigwedge_{v < m} G_k^*(v)$ with variables $x_u,v_1,\ldots,v_k$ replaced by $x_u,u,v_1,\ldots,v_k$, where $G_k^*(v)$ is $G_k(v)$ with AND’s replaced by OR’s and vice versa,
(iii) $\tilde{\psi}_i(m, u)$ is $G_{i-2}(u)$ with variables $x_u,y_2,\ldots,y_{i-1}$ replaced by conjunction for $i$ even respectively by disjunction for $i$ odd of variables

$$x_u, y_2, \ldots, y_{i-1}, t, \quad t < \sqrt{\frac{i \cdot m \cdot \log(m)}{2}}.$$
Circuit $C_i^m$ is a disjunction of $\frac{m!}{2}$ conjunctions:

$$\psi_i(m, u) \land \bigwedge_{u < v < m} \neg \psi_i(m, v),$$

one for each odd $u < m$. Clearly $C_i^m$ computes $\Psi_i^A(m)$ for every $A \subset \omega$.

(3.6) $(B_j)_j$ is a partition of variables of $C_i^m$ consisting of $m'^{i-1}$ classes

$$\left\{ x_{y_1, \ldots, y_{i-1}, l} \mid l < \frac{i \cdot m \cdot \log(m)}{2} \right\}$$

for every choice of $y_1, \ldots, y_{i-1} < m$. So these are classes entering a gate at level 1 of $C_i^m$.

$R_+^*$, for $0 < q < 1$, is a probability space of restrictions $\rho$ (i.e. maps of variables into $\{0, 1, *\}$) defined by

(i) with probability $q$: $s_j = *$, and $s_j = 0$ with probability $1 - q$,

(ii) for every variable $x \in B_j$, with probability $q$: $\rho(x) = s_j$, and with probability $1 - q$: $\rho(x) = 1$.

Space $R_-^*$ is defined analogically, interchanging the roles of 0 and 1 in the definition of $R_+^*$ (see [H1, 2] for more details).

For restriction $\rho$ from $R_+^*$, $g(\rho)$ is a restriction and renaming of variables defined as follows: For all $B_j$ with $s_j = *$, $g(\rho)$ gives value 1 to all $x_{y_1, \ldots, y_i} \in B_j$ given value * by $\rho$ except one, say the one with minimal last index $y_i$, to which $g(\rho)$ assigns new name $x_{y_1, \ldots, y_{i-1}}$. If $\rho$ is from $R_-^*$, $g(\rho)$ is defined identically using 0 instead of 1.

Finally, if $G$ is a circuit with variables among those of $C_i^m$ then $g(p) \downarrow g(p)$ denotes a boolean function with variables $x_{y_1, \ldots, y_{i-1}}$ computed by $G$ after applying to it successively $\rho$ and $g(\rho)$.

(3.7) Lemma (Hastad). Fix $q := \sqrt{\frac{2 \cdot i \cdot \log(m)}{m}}$. Then it holds.

(a) Let $G$ be a depth 2 subcircuit of $C_i^m$, so $G$ is either an OR of AND’s of size $\leq \frac{i \cdot m \cdot \log(m)}{2}$ or an AND of OR’s of size $\leq \frac{i \cdot m \cdot \log(m)}{2}$. Then for a random restriction $\rho$ from $R_+^*$ in the former case or from $R_-^*$ in the latter one the probability that $(G \upharpoonright \rho) \upharpoonright g(\rho)$ is an OR (resp. an AND) of at least $\frac{i-1}{i \cdot m \cdot \log(m)}$ different variables is at least $1 - \frac{1}{3} m^{-i+1}$.

(b) For $i \geq 3$ and $m$ sufficiently large and $\rho$ random from $R_+^*$ if $i$ is even or from $R_-^*$ if $i$ is odd it holds: with probability at least $\frac{2}{3}$ circuit $(C_i^m \upharpoonright \rho) \upharpoonright g(\rho)$ contains $C_i^{m-1}$, i.e. for some renaming $\kappa$ of variables

$$(C_i^m \upharpoonright \rho) \upharpoonright g(\rho) \upharpoonright \kappa = C^{m-1}_i.$$  

(c) For $i = 2$ and $\rho$ from $R_+^*$ random, circuit $(C_2^m \upharpoonright \rho) \upharpoonright g(\rho)$ contains with probability at least $\frac{1}{3}$ circuit $C_1^n$, for $n = \sqrt{\frac{m \cdot \log(m)}{2}}$.

Proof. This is Hastad's lemma broken into parts which we will later need separately. For completeness we outline the proof, for details see [H1, 2].

(a) Assume $G$ is an OR of AND’s and $\rho$ is from $R_+^*$. An AND gate corresponds to a class $B_j$ of variables and takes value $s_j$ with probability at
least

\[ 1 - (1 - q)^{|B_j|} = 1 - \left(1 - \sqrt{\frac{2 \cdot i \cdot \log(m)}{m}}\right)^{\frac{i \cdot m^* \cdot \log(m)}{2}} > 1 - \frac{1}{6} e^{-i \cdot \log(m)} = 1 - \frac{1}{6} m^{-i}. \]

So with probability at least \(1 - \frac{1}{6} m^{-i+1}\) this is true for all \(m\) AND's in \(G\).

Expected number of AND's assigned \(s_j\) and not 0 (in the definition of \(\rho\)) is

\[ m \cdot q = \sqrt{2 \cdot i \cdot m \cdot \log(m)} \]

and we can get with probability \(\geq 1 - \frac{1}{6} m^{-i}\) at least \(\frac{(i-1) \cdot m^* \cdot \log(m)}{2}\) \(s_j\)'s assigned.

Thus with probability at least \(1 - \frac{1}{3} m^{-i+1}\) \((G \upharpoonright \rho) \upharpoonright g(\rho)\) is an OR of at least \(\frac{(i-1) \cdot m^* \cdot \log(m)}{2}\) variables.

(b) There is \(m'^{-2}\) different subcircuits \(G\) of depth 2 in \(C_i^m\). Thus with probability at least \(1 - \frac{1}{3} m^{-1}\) \(\geq \frac{2}{3}\) all of them are restricted as required in (a). Hence additional renaming \(\kappa\) produces \(C_{i-1}^m\).

(c) If \(i = 2\), \(\psi_i(m, u)\) are just AND's of size at most \(\sqrt{m \cdot \log(m)}\) corresponding to classes \(B_j\), and there is \(m\) different of them. Thus, by (a), with probability at least \(\frac{5}{6}\) they all take value \(s_j\) which is, again with probability at least \(\frac{5}{6}\), equal to \(*\) for at least \(\sqrt{m \cdot \log(m)}\) of them. □

(3.8) A boolean circuit is \(\Sigma_{i,m}^{S,t}\) if it has depth \(i + 1\) with top gate OR, with at most \(S\) gates in levels \(2, 3, \ldots, i + 1\), bottom gates have arity at most \(t\) and variables are those of \(C_i^m\).

A tt-reducibility \(D = (f; E_1, \ldots, E_r)\) of type \((i, m, k)\) is a boolean function \(f(w_1, \ldots, w_r)\) in \(r \leq \log(m)^k\) variables together with a list of \(r \Sigma_{i,m}^{S,t}\)-circuits \(E_1, \ldots, E_r\), where \(S = 2^{\log(m)^k}, t = \log(m)^k\).

\(D\) naturally computes a boolean function on variables of \(C_i^m\): first evaluates \(w_j := E_j\) and then \(f\) on \(w_j\)’s.

(3.9) The following switching lemma is crucial. For the proof we refer to [H1, 2].

Lemma (Hastad). Let \(G\) be an AND of OR's of size \(\leq t\) of variables of \(C_i^m\) and \(\rho\) a random restriction from \(R_\rho^+ \cup R_\rho^-\). Then probability that \((G \upharpoonright \rho) \upharpoonright g(\rho)\) cannot be written as an OR of AND's of size \(s\) is bounded by \((6 \cdot q \cdot t)^s\).

The same probability is for converting an OR of AND's into an AND of OR's. □

(3.10) Lemma. Let \(D\) be a tt-reducibility of type \((i, m, k)\) and \(\rho\) a random restriction from \(R_\rho^+ \cup R_\rho^-\) with \(q := \sqrt{2 \cdot i \cdot \log(m)}\).

Then with probability at least \(\frac{1}{3}\),

\[(D \upharpoonright \rho) \upharpoonright g(\rho) = (f; (E_1 \upharpoonright \rho) \upharpoonright g(\rho), \ldots, (E_r \upharpoonright \rho) \upharpoonright g(\rho))\]

is a tt-reducibility of type \((i - 1, m, k)\).
Proof. Lemma (3.9) with \( s = t = \log(m)^k \) gives probability of a failure to convert one depth 2 subcircuit of any \( E_j \) at most

\[
(6 \cdot q \cdot t)^s = \left( 6 \cdot \sqrt[2]{\frac{2 \cdot i \cdot \log(m)}{m}} \cdot \log(m)^k \right)^{\log(m)^k},
\]

which can be made smaller than any \( 2^{-h} \cdot \log(m)^k \) increasing \( m \) sufficiently.

There is at most \( 2^\log(m)^k \) such subcircuits so taking \( h = 2 \) makes probability of a failure to convert any of them at most \( 2^{-\log(m)^k} \cdot \frac{1}{2} \). When all such subcircuits are converted, they can be merged with gates at level 3. □

(3.11) Lemma. Assume that there is a tt-reducibility \( D_i \) of type \((i, m, k)\) computing \( \Psi_i^A(m) \) for every \( A \subseteq \omega \). Then there is a tt-reducibility \( D_1 \) of type \((1, m, k)\) computing \( \Psi_1^B(\sqrt{(m \cdot \log(m))}/2) \) for every \( B \subseteq \omega \).

Proof. \( \Psi_i^A(m) \) is computed by \( C_i^m \). By Lemmas (3.7) and (3.10) (and \( q \) as there) a random restriction \( \rho \) from \( R_i^+ \) if \( i \) is even or from \( R_i^- \) if \( i \) is odd converts simultaneously \( C_i^m \) into \( C_{i-1}^m \) and \( D_i \) into \( D_{i-1} \) of type \((i-1, m, k)\) with probability at least \( \frac{1}{6} \). Therefore there exists such a restriction \( \rho \). Clearly \((C_i^m \upharpoonright \rho) \upharpoonright g(\rho)\) and \((D_i \upharpoonright \rho) \upharpoonright g(\rho)\) compute the same predicate.

Applying this \((i-1)\)-times, clause (c) of (3.7) in the last application, gives the statement. □

(3.12) Now we complete the chain of reductions by a lemma which is essentially an oracle construction from \([B-H]\).

Lemma. Let \( k \) be arbitrary. Then for \( m \) sufficiently large there is no tt-reducibility \( D \) of type \((1, m, k)\) computing \( \Psi_1^A(\sqrt{(m \cdot \log(m))}/2) \) for every \( A \subseteq \omega \).

Proof. Let \( D = (f; E_1, \ldots, E_r) \) be type \((1, m, k)\) tt-reducibility and denote circuit \( C_i^n \) for \( n = \sqrt{(m \cdot \log(m))}/2 \) by \( C \). In successive steps we shall construct sets \( A_i^+, A_i^- \) and \( I_s \) satisfying

(a) \( A_i^+ \cap A_i^- = \emptyset \) and both contain only numbers \( < \sqrt{(m \cdot \log(m))}/2 \),
(b) \( |A_i^+| \leq s, |A_i^+ \cup A_i^-| \leq s \cdot \log(m)^k \),
(c) at least half of numbers \( \leq \max(A_i^+) \) belong to \( A_i^- \cup A_i^+ \),
(d) \( I_s \subseteq \{1, \ldots, r\}, |I_s| = s \),
(e) for every \( B \subseteq \omega \) such that \( A_i^+ \subseteq B \) and \( A_i^- \cap B = \emptyset \), and every \( j \in I_s \) it holds: \( E_j^B = 1 \).

Initiate \( A_0^+ := A_0^- := I_0 := \emptyset \).

Step \( s + 1 \). Assume we have sets \( A_s^+, A_s^-, I_s \) satisfying the above conditions. Put \( B := A_s^+ \); therefore \( E_j^B = 1 \) for all \( j \in I_s \). Consider three cases

(1) \( D^B = 1 \) but \( \max B \) is even or \( D^B = 0 \) but \( \max B \) is odd. Then STOP.
(2) \( D^B = 1 \) and \( \max B = \max A_s^+ \) is odd. Take set

\[ S = \{ x < 2^{\log(m)^k} \mid \max A_s^+ \cdot x, x \text{ is even}, x \notin A_s^- \} \]

\( S \) is nonempty by conditions (a), (b), and (c). There are two possible subcases:
(2a) We can add some \( x \in S \) to \( B \) to form \( B' := B \cup \{x\} \), such that \( DB' = DB = 1 \). Then put \( A^+_{s+1} := A^+_s \cup \{x\} \), \( A^-_{s+1} := A^-_s \) and STOP.

(2b) Not (2a). Take \( x := \min S \) and form \( A^+_{s+1} := A^+_s \cup \{x\} \). As \( D \) changes value some \( E_{j_0} \) for \( j_0 \notin I_s \) had to become true. Take an AND of \( E_{j_0} \) (containing \( x \)) which becomes true and add indices of all variables negatively occurring in it to \( A^-_{s+1} \) to form \( A^-_{s+1} \) (note that none of them is in \( A^+_s \)). Put \( I_{s+1} := I_s \cup \{j_0\} \) and GO TO STEP \((s+2)\).

Note that \( A^+_{s+1}, A^-_{s+1} \) satisfy the conditions (a)-(e); in particular, (c) holds as we have chosen for \( x \) the minimal element of \( S \).

(3) \( DB = 0 \) and max \( A^+_s \) is even. Take set
\[ S = \{x < 2^{\log(m)k} \mid \max A^+_s < x, x \text{ odd}, x \notin A^-_s \}, \]
and proceed analogically with case (2).

If we do not stop at step \( s \), necessarily \( I_s \) is a proper subset of \( I_{s+1} \). Therefore we stop in at most \( r \leq \log(m)k \) steps. Take \( A := A^+_s \) for final \( s \). Clearly \( DA \) does not agree with \( CA \).

(3.13) **Proof of Theorem (3.1).** We construct oracle \( A \) such that for all \( i \geq 1 \), \( \Psi_i^A(x) \) is not in \( \leq^P_i \left( \Sigma^P_i(A) \right) \). Let \((M_j)\) enumerate all polynomial-time machines. Considering successively all pairs \((i, j)\) we shall build \( A \) in stages assuring that \( M_j \) does not provide a tt-reducibility of \( \Psi_i^A(x) \) to \( \Sigma^P_i(A) \).

Let \( A_s \) be an approximation to \( A \) constructed in first \( s \) stages and let \((i, j)\) be the first pair not yet considered. Choose \( m = m_{s+1} \) so large that all numbers considered up to now are small w.r.t. \( m \). \( M_j \) outputs on input \( m \) a boolean function \( f(w_1, \ldots, w_r) \) and queries \( z_1, \ldots, z_r \) to a (canonical complete one) \( \Sigma^P_i(A) \)-oracle (we do not have to worry how \( f \) is presented). A query \( z \) to the \( \Sigma^P_i(\alpha) \)-oracle naturally correspond to an evaluation of a \( \Sigma_{i,m}^{\log(m)} \)-circuit on variables corresponding to atomic statements \( \" \alpha e a \" \), where \( \alpha = 2^{\log(m)k} \), \( k \) a constant. We first evaluate variables corresponding to \( \alpha \) according to \( A_s \) and then set equal to 0 all those for which \( \alpha \) is not of the form \( \langle i, m, y_1, \ldots, y_l \rangle \), as these are the only variables on which truth-value of \( \Psi_i^A(m) \) depends.

This leaves us with a tt-reducibility of type \((i, m, k)\) and by Lemmas (3.11) and (3.12) no such reducibility computes \( \Psi_i^A(m) \) correctly for all \( \alpha \). Define \( A_{s+1} \supset A_s \) in such a way that the tt-reducibility fails, i.e. \( M_j \) fails too. Then proceed to the next pair \((i, j)\).

This completes the proof of the theorem.

(3.14) Combining Lemma (2.3) and Theorem (3.1) gives

**Corollary.** \( S^i_2(\alpha) \neq T^i_2(\alpha) \) for all \( i \geq 1 \).

**Acknowledgment**

A part of this work was performed while I was enjoying the hospitality of S. Buss and the University of California at San Diego during the workshop Proof Theory, Fragments of Arithmetic and Complexity organized as a part of a joint project of NSF and ČSAV.
References


Mathematical Institute, Žitná 25, Praha 1, 115 67, Czech Republic
E-mail address: krajicek@csearn.bitnet