Consequences of the Provability of $\mathbf{NP} \subseteq \mathbf{P/poly}$

Stephen Cook*
University of Toronto

Jan Krajíček †
Academy of Sciences and Charles University
Prague

Abstract

We prove the following results: (i) $\mathbf{PV}$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ iff $\mathbf{PV}$ proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$. (ii) If $\mathbf{PV}$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ then $\mathbf{PV}$ proves that the Polynomial Hierarchy collapses to the Boolean Hierarchy. (iii) $\mathbf{S}^1_2$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ iff $\mathbf{S}^1_2$ proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$. (iv) If $\mathbf{S}^1_2$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ then $\mathbf{S}^1_2$ proves that the Polynomial Hierarchy collapses to $\mathbf{P}^{\mathbf{NP}}[\log n]$. (v) If $\mathbf{S}^2_2$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ then $\mathbf{S}^2_2$ proves that the Polynomial Hierarchy collapses to $\mathbf{P}^{\mathbf{NP}}$.

Motivated by these results we introduce a new concept in proof complexity: proof systems with advice, and we make some initial observations about them.

1 Introduction

The theory $\mathbf{PV}$ [Coo75] formalizes reasoning that uses only polynomial time concepts. Similar theories (the so called bounded arithmetic theories) exist

*This research was begun while the authors were visiting the Isaac Newton Institute for Mathematical Sciences (program Logic and Algorithms), in Cambridge. It was supported in part by the Natural Sciences and Engineering Research Council of Canada

†Supported in part by grants A1019401, AV0Z10190503, MSM0021620839, 201/05/0124, and LC505.

for many other complexity classes [Coo05, CN06]. Separating theories (at appropriate levels of quantifier complexity) corresponding to two complexity classes may not separate the classes, but it could still carry great significance. In fact, the problem of separating two such theories is often closely linked with the problem of whether the distinctness of two complexity classes is consistent with a suitable bounded arithmetic theory. For example, theories **PV** and $\mathbf{S}^1_2$ are different if $\mathbf{coNP} \not\subseteq \mathbf{NP/poly}$ is consistent with **PV**. Another prominent example is this: Theory $\mathbf{S}_2$ is not finitely axiomatizable (i.e. theories $\mathbf{S}^i_2$ are all different) iff it is consistent with $\mathbf{S}_2$ that **PH** does not collapse [KPT91, Bus95, Zam96].

It is therefore interesting to study the consistency with bounded arithmetic theories of various standard conjectures separating complexity classes. A good example is the conjecture $\mathbf{P} \neq \mathbf{NP}$. Showing this is consistent with **PV** is a major open problem. In more detail, $\mathbf{P} = \mathbf{NP}$ is equivalent to the existence of a polynomial time function F such that $F(A)$ is a satisfying assignment for A whenever A is a satisfiable propositional formula. This can be formalized in the language of **PV** by

$$\forall T, A,\ SAT(T, A) \rightarrow SAT(F(A), A) \tag{1}$$

where F is a **PV** function and $SAT(T, A)$ is an open **PV** formula expressing that T is a satisfying assignment for formula A. Showing consistency of $\mathbf{P} \neq \mathbf{NP}$ with **PV** is the same as showing that (1) is not provable in **PV** for any polynomial time F. (A natural way to do this would be, for every F, construct a model of **PV** in which (1) is false.) This would mean that even if a polytime F satisfying (1) exists, its correctness could not be proved using only polynomial time concepts.

It is known that PV proves $\mathbf{P} = \mathbf{NP}$ iff **PV** proves $\mathbf{NP} = \mathbf{coNP}$ iff **PV** defines a polynomial time algorithm that assigns to a propositional formula either a satisfying assignment or an Extended Frege proof of its unsatisfiability. The same holds for S^1_2. In particular, the consistency of $\mathbf{P} \neq \mathbf{NP} \neq \mathbf{coNP}$ with these theories would follow from a super-polynomial lower bound for Extended Frege proofs. This is one of the reasons why lower bounds for Extended Frege systems would be so important (and presumably why they seem so difficult).

In this paper we are concerned with whether the conjecture $\mathbf{NP} \not\subseteq \mathbf{P/poly}$ (some **NP** problem cannot be solved by any polynomial size family of Boolean circuits) is consistent with the theories **PV**, $\mathbf{S}^1_2$, $\mathbf{S}^2_2$. A well-known consequence of $\mathbf{NP} \subseteq \mathbf{P/poly}$ is that the polynomial hierarchy **PH** collapses to the second level [KL80, KL82], i.e.

$$\mathbf{NP} \subseteq \mathbf{P/poly} \implies \mathbf{PH} = \mathbf{\Sigma}_2^p \cap \mathbf{\Pi}_2^p \ .$$

(The conclusion has been improved in various ways, including $\mathbf{PH} \subseteq \mathbf{ZPP^{NP}}$. See [Cai01] for a discussion.) Here we show that stronger collapses can be inferred from stronger assumptions of the form $\mathbf{NP} \not\subseteq \mathbf{P/poly}$ is inconsistent with various theories. In particular, if the theory is $\mathbf{PV}$, then the collapse is to the Boolean hierarchy (i.e. the bounded query hierarchy). If the theory is $\mathbf{S}_2^1$, then the collapse is to $\mathbf{P^{NP}}[O(\log n)]$ (polynomial time with $O(\log n)$ queries to an $\mathbf{NP}$ oracle). If the theory is $\mathbf{S}_2^2$, then the collapse is to $\mathbf{P^{NP}}$. In all cases, the collapses are provable in the corresponding theories.

We also show two other intriguing results: $\mathbf{PV}$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ iff $\mathbf{PV}$ proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$, and $\mathbf{S}_2^1$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ iff $\mathbf{S}_2^1$ proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$. (Here the notation $/O(1)$ refers to a constant number of advice bits, and $/O(\log n)$ refers to $O(\log n)$ advice bits.) In the final section we introduce the notion of proof systems with advice.

2 Preliminaries

We presuppose basic knowledge of bounded arithmetic (see for example [Kra95] or [Bus98]). However we formulate our results in the two-sorted setting [Coo05, CN06], where $\mathbf{PV}$ becomes $\mathbf{VPV}$, $\mathbf{S}_2^1$ becomes $\mathbf{V}^1$, and $\mathbf{S}_2^2$ becomes $\mathbf{V}^2$. In this setting lower case letters $x, y, z, m, n, ...$ range over $\mathbb{N}$, and upper case letters $A, B, C, X, Y, Z, ...$ range over finite bit strings (technically over finite subsets of $\mathbb{N}$). The two-sorted vocabulary includes the symbols $0, 1, +, \cdot, =, \leq$ of first-order arithmetic, and also the length function $|X|$, where the length of the finite set $X \subseteq \mathbb{N}$ is 1 plus the largest element of X, or 0 if X is empty.

For the two-sorted theory $\mathbf{VPV}$, the vocabulary also includes symbols for all polynomial time functions, both number-valued functions $f(\vec{x}, \vec{X})$ and string-valued functions $F(\vec{x}, \vec{X})$. (Here "polynomial time" means time polynomial in the length of the inputs, where inputs and outputs of the number sort are written in unary notation: e.g. 5 is written 11111). The axioms for $\mathbf{VPV}$ include definitions for all of these functions, based on Cobham's Theorem. Also $\mathbf{VPV}$ proves the number induction scheme for open formulas.

It is an important fact that $\mathbf{VPV}$ is a universal theory (i.e. it can be axiomatized by purely universal formulas).

Bounded quantifiers for strings have the form $\exists X \leq t\varphi$, which stands for $\exists X(|X| \leq t \wedge \varphi)$, or $\forall X \leq t\varphi$, which stands for $\forall X(|X| \leq t \rightarrow \varphi)$. Here t is a number term which does not involve X.

$\mathbf{\Sigma}_1^B$ formulas have the form $\exists X_1 \leq t_1 \cdots \exists X_k \leq t_k \varphi$ where φ has no string quantifiers, but may have bounded number quantifiers. More generally $\mathbf{\Sigma}_i^B$ formulas begin with i blocks of bounded string quantifiers, in which the first block is existential, the second is universal, etc. $\mathbf{\Pi}_i^B$ formulas are the same, except the first block is universal. Note that $\mathbf{\Sigma}_i^B$ formulas correspond to *strict* $\mathbf{\Sigma}_i^b$ formulas in the single-sorted case, because we require that all string quantifiers are in front.

The theories $\mathbf{V}^i, i \geq 1$, have the finite two-sorted vocabulary mentioned above. For these theories, $\mathbf{\Sigma}_i^B$ and $\mathbf{\Pi}_i^B$ formulas are restricted to this vocabulary, but in the context of **VPV** we allow these formulas to have the full vocabulary of **VPV**. The theory $\mathbf{V}^i$ proves the number induction scheme for $\mathbf{\Sigma}_i^B$ formulas. The theory $\mathbf{V}^i(\mathbf{VPV})$ has the vocabulary of **VPV** and proves the number induction scheme for $\mathbf{\Sigma}_i^B$ formulas over this larger vocabulary. $\mathbf{V}^i(\mathbf{VPV})$ is a conservative extension of $\mathbf{V}^i$, but $\mathbf{V}^i(\mathbf{VPV})$ is not conservative over **VPV** unless the polynomial hierarchy collapses.

An important fact is that a set of strings is in **NP** iff it is represented by a $\mathbf{\Sigma}_1^B$ formula $\varphi(X)$ with one free string variable X. This is true whether or not we allow $\mathbf{\Sigma}_1^B$ formulas to have the full vocabulary of **VPV**.

The class **P/poly** consists of all problems solvable in polynomial time with polynomial "advice". That is, for each input length n there exists a string Y_n of length $n^{O(1)}$ (the advice) such that when the polytime algorithm is supplied with Y_n in addition to the input string of length n, the algorithm obtains the correct answer. An equivalent definition of **P/poly** is the class of all problems L solvable by a family $\langle C_n, n \in \mathbb{N}\rangle$ of Boolean circuits such that C_n has size $n^{O(1)}$ and solves the restriction of L to inputs of length n.

It is straightforward to verify that **VPV** can prove the equivalence of the two definitions of **P/poly**.

To formalize the assertion $\mathbf{NP} \subseteq \mathbf{P/poly}$ as a **VPV** formula, we start by noting that **VPV** can prove the **NP**-completeness of the propositional satisfiability problem in the following sense. For every $\mathbf{\Sigma}_1^B$ formula $\varphi(X)$ there is a **VPV** string function $F_\varphi(X)$ which takes a string X to a string $F_\varphi(X)$ encoding a propositional formula such that

$$\mathbf{VPV} \ \vdash \ \varphi(X) \leftrightarrow \exists T \ SAT(T, F_\varphi(X)) \tag{2}$$

where $SAT(T, A)$ is an open **VPV** formula which holds iff truth assignment

T satisfies formula A.

Thus $\mathbf{NP} \subseteq \mathbf{P/poly}$ is equivalent in **VPV** to the assertion that the satisfiability problem can be solved by a polynomial size family of Boolean circuits. By the "self-reducibility" of satisfiability, it follows that this is again equivalent to the existence of a polynomial size family of Boolean circuits such that the n-th circuit C'_n in the family, given a satisfiable propositional formula A (coded as a bit string of length at most n) as input, outputs $C'_n(A)$, a satisfying assignment for A.

Further, **VPV** proves self-reducibility in the following sense. We define a **VPV** formula $Correct(C, n)$, asserting that the Boolean circuit C correctly solves the satisfiability problem for formulas A of length at most n, as follows:

$$Correct(C, n) \equiv \forall A \leq n,\ C(A) = 1 \leftrightarrow \exists T\ SAT(T, A)\ .$$

Then there is a **VPV** function $CIRC(C, n)$ which, given a circuit C satisfying $Correct(C, n)$ gives a Boolean circuit C' which outputs a satisfying assignment $C'(A)$ for every satisfiable formula A of length at most n. We claim that

$$\begin{aligned} &\mathbf{VPV} \vdash Correct(C, n) \rightarrow \\ &\qquad\qquad \forall T, A \leq n,\ SAT(T, A) \rightarrow SAT(CIRC(C, n)(A), A)\ . \end{aligned} \tag{3}$$

For each i, the circuit $CIRC(C, n)$ finds the i-th bit of the satisfying assignment by asking C whether A remains satisfiable when the $i - th$ variable of A is set to 1, given the values it has previously found for the first $i - 1$ variables. Then (assuming $Correct(C, n)$ and $SAT(T, A)$) **VPV** proves by induction on i that A instantiated by the first i truth values is satisfiable, according to C. The claim (3) follows.

The claim (3) justifies coding the assertion $\mathbf{NP} \subseteq \mathbf{P}/poly$ by the following $\mathbf{\Sigma}_2^B$ formula:

$$\forall n \exists C \leq t \forall A \leq n \forall T \leq n,\ SAT(T, A) \ \rightarrow\ SAT(C(A), A) \tag{4}$$

where $t = t(n)$ is a **VPV** number term (we may assume it has the form n^ℓ), $C(A)$ is a **VPV** term expressing the output of circuit C on input A, and $SAT(T, A)$ is a **VPV** open formula which asserts that truth assignment T satisfies formula A.

3 Results for Universal Theories

A useful tool for studying universal theories is the KPT witnessing theorem [KPT91]. This is a form of the Herbrand Theorem, and can be stated in a general first-order context as follows.

Theorem 3.1 (KPT)**.** *Let $\mathcal{T}$ be a universal theory over a vocabulary $\mathcal{L}$ which contains at least one constant or function symbol. Let $\varphi(x, y, z)$ be an open $\mathcal{L}$-formula and suppose $\mathcal{T}$ proves $\forall x \exists y \forall z \varphi(x, y, z)$. Then there exists a finite sequence $t_1(x), t_2(x, z_1), \ldots, t_k(x, z_1, \ldots, z_{k-1})$ of $\mathcal{L}$-terms (containing only the displayed variables) such that*

$$\begin{aligned} \mathcal{T} \vdash \forall x \forall \vec{z},\ & \varphi(x, t_1(x), z_1) \vee \varphi(x, t_2(x, z_1), z_2) \\ & \vee \ldots \vee \varphi(x, t_k(x, z_1, \ldots, z_{k-1}), z_k). \end{aligned}$$

Proof: [from [CT06]] Let $b, c_1, c_2, ...$ be a list of new constants, and let $u_1, u_2, ...$ be an enumeration of all variable-free terms built from symbols of $\mathcal{L}$ together with $b, c_1, c_2, ...$, where the only new constants in u_k are among $\{b, c_1, ..., c_{k-1}\}$. It suffices to show that

$$\mathcal{T} \cup \{\neg\varphi(b, u_1, c_1), \neg\varphi(b, u_2, c_2), \ldots, \neg\varphi(b, u_k, c_k)\}$$

is unsatisfiable for some k.

Suppose otherwise. Then by compactness

$$\mathcal{T} \cup \{\neg\varphi(b, u_1, c_1), \neg\varphi(b, u_2, c_2), ...\} \tag{5}$$

has a model M. Since $\mathcal{T}$ is universal, the substructure M' consisting of the denotations of the terms $u_1, u_2, ...$ is also a model for (5). It is easy to see that

$$M' \models \mathcal{T} + \forall y \exists z \neg\varphi(b, y, z)$$

and hence $\mathcal{T} \not\vdash \forall x \exists y \forall z \varphi(x, y, z)$. □

The following is an easy consequence of this KPT theorem in the two-sorted setting.

Corollary 3.2. *Let* **TVPV** *be* **VPV***, or any universal theory with the same vocabulary as* **VPV***. If* **TVPV** *proves (4) then there are* **VPV** *functions $C_1, \cdots, C_k$ whose values are Boolean circuits such that* **TVPV** *proves*

$$\begin{aligned} \forall n \forall A_1 \cdots A_k, T_1 \cdots T_k \le n,\ & [SAT(T_1, A_1) \ \rightarrow\ SAT(C_1(n)(A_1), A_1)] \vee \\ & [SAT(T_2, A_2) \ \rightarrow\ SAT(C_2(n, A_1, T_1)(A_2), A_2)] \vee \cdots \vee \\ & [SAT(T_k, A_k) \ \rightarrow\ SAT(C_k(n, A_1, \cdots, A_{k-1}, T_1, \cdots, T_{k-1})(A_k), A_k)]\ . \end{aligned} \tag{6}$$

Definition 3.3. *Let $k = k(n)$ be a function on natural numbers. A language $L \subseteq \{0,1\}^*$ is in the class* $\mathbf{NP}/k(n)$ *(*$\mathbf{NP}$ *with $k(n)$ bits of advice) iff there is a polynomial time relation $R(X, Y, i)$ such that for all $n \in \mathbb{N}$ there exists $i \leq 2^{k(n)}$ (the advice) so for all $X \in \{0,1\}^n$*

$$X \in L \Leftrightarrow \exists Y \leq n^{O(1)}\ R(X, Y, i)) \ . \tag{7}$$

The assertion $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$ is equivalent to saying that for every $\mathbf{\Sigma}_1^B$ formula $\varphi(X)$ there is a $\mathbf{\Sigma}_1^B$ formula $\psi(X, i)$ and a constant k such that

$$\forall n \bigvee_{i=1}^{k} \forall X,\ |X| = n \to [\neg\varphi(X) \leftrightarrow \psi(X, i)] \ . \tag{8}$$

Theorem 3.4. *Let* **TVPV** *be* **VPV***, or any universal extension of* **VPV** *with the same vocabulary. Then* **TVPV** *proves* $\mathbf{NP} \subseteq \mathbf{P/poly}$ *as in (4) iff* **TVPV** *proves* $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$ *as in (8).*

Proof: Assume that **TVPV** proves $\mathbf{NP} \subseteq \mathbf{P/poly}$. It follows from Corollary 3.2 that **TVPV** proves (6). Since **VPV** proves that the propositional unsatisfiability problem is complete for **coNP** (see (2)), it suffices to prove (8) for the case $\varphi(X)$ is $\exists T \leq |X|\ SAT(T, X)$. Let the constant k be as in (6). Given n, the advice i_n for n is the smallest number $j \leq k$ such that (6) holds for this n and all $A_1, \cdots, A_k, T_1, \cdots, T_k$, with k replaced by j. Note that **VPV** easily proves the existence of i_n, since k is a constant and only order properties of $\mathbb{N}$ are needed.

We define the $\mathbf{\Sigma}_1^B$ formula $\psi(A, i)$ in (8) to assert the existence of $A_1, \cdots A_{i-1}$ and $T_1, \cdots, T_{i-1}$ which falsify the first $i-1$ disjuncts in (6) and also falsify

$$SAT(C_i(n, A_1, \cdots, A_{i-1}, T_1, \cdots, T_{i-1})(A), A) \ .$$

To make clear the application of Corollary 3.2 we use the same symbols as in (6) but note that $A_1, \ldots, A_{i-1}$ as well as $T_1, \ldots, T_{i-1}$ are existentially quantified in $\psi(A, i)$ and hence their occurrences are not free. By the definition of the advice i_n given above, it is easy to verify in **TVPV** that if A is a propositional formula of length n, then $\psi(A, i_n)$ holds iff A is unsatisfiable.

Conversely, suppose that **TVPV** proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$ as in (8). For fixed n, define

$$\begin{aligned} U =& \{X \mid \varphi(X) \wedge |X| = n\} \\ V_i =& \{X \mid \psi(X, i) \wedge |X| = n\}, i = 1 \cdots k \ . \end{aligned}$$

Then by (8) we conclude in **TVPV** that at least one V_i is the complement of U:

$$\bigvee_i [U \cap V_i = \varnothing \wedge U \cup V_i = \{X \ : \ |X| = n\}] \ .$$

It follows that (writing $U(X)$ for $X \in U$ and $V_i(X)$ for $X \in V_i$) for any $I \subseteq \{1, \ldots, k\}$, **TVPV** proves (for $X_1, \cdots, X_k, Y$ ranging over strings of length n)

$$\bigwedge_{i \in I} U(X_i) \wedge V_i(X_i) \ \rightarrow \ \bigvee_{j \notin I} U(Y) \vee V_j(Y) \ . \tag{9}$$

This is because the conditions in the antecedent imply that V_i is not the complement of U while the condition in the succedent for j is implied if V_j is the complement.

The prenex form of this formula is $\mathbf{\Sigma}_1^B$(with additional free variables), so the Herbrand theorem provides us with polynomial time functions

$$F_I(\tilde{X}_{i_1}, \cdots, \tilde{X}_{i_\ell}, Y) \ ,$$

one for each $I = \{i_1 < \cdots < i_\ell\}$ which from witnesses $\tilde{X}_i$'s to the validity of the conjuncts in the antecedent (i.e. $\tilde{X}_i$ is $X_i \in U \cap V_i$ together with the two **NP** witnesses for the membership in the two sets) and from Y finds a $j \notin I$ and an **NP** witness of the membership of Y in either U or V_j.

Now we show how to compute U in polynomial time using polynomial size advice. For a given length n we get the following advice:

- $I = \{i \mid U \cap V_i \neq \varnothing\}$.
- For each $i \in I$ a witness $\tilde{A}_i$ to the fact that $U \cap V_i \neq \varnothing$, i.e. a string A_i of length n in the intersection and the two NP-witnesses of its membership in U and V_i.

If we want to decide whether or not $B \in U$, $|B| = n$, we simply compute

$$F_I(\tilde{A}_{i_1}, ..., \tilde{A}_{i_\ell}, B) \ .$$

The output is either a witness to $B \in U$ or a witness to $B \in V_j$ for some $j \notin I$. In the latter case, necessarily $B \notin U$ by definition of I.

□

The Boolean Hierarchy **BH** is the smallest class of sets that contains **NP** and is closed under the Boolean operations of intersection, union, and complement. It coincides with the bounded query class $\mathbf{P}^{\mathbf{NP}}[O(1)]$ of sets which can be computed in polynomial time with a constant number of queries to an **NP** oracle (see [Bei91]). Thus

$$\mathbf{NP} \subseteq \mathbf{BH} \subseteq \mathbf{P}^{\mathbf{NP}}[\log n] \subseteq \mathbf{P}^{\mathbf{NP}} \subseteq \mathbf{\Sigma}_2^p \cap \mathbf{\Pi}_2^p \subseteq \mathbf{PH}$$

where **PH** is the polynomial hierarchy and $\mathbf{P}^{\mathbf{NP}}[\log n]$ is the class of sets solvable in polynomial time by making $O(\log n)$ queries to an **NP** oracle. It is not known whether any of these inclusions is proper.

Theorem 3.5. *Let* **TVPV** *be* **VPV** *or any universal extension of* **VPV** *with the same vocabulary. If* **TVPV** *proves* $\mathbf{NP} \subseteq \mathbf{P/poly}$ *as in (4), then* **TVPV** *proves that the polynomial hierarchy collapses to the Boolean Hierarchy.*

Proof: As in the proof of Theorem 3.4, we may assume that **TVPV** proves (6). To show $\mathbf{PH} = \mathbf{BH}$ it suffices to show that $\mathbf{\Sigma}_2^p \subseteq \mathbf{BH}$. Thus suppose that $L \in \mathbf{\Sigma}_2^p$, so

$$X \in L \;\Leftrightarrow\; \exists Y \leq f(|X|)\forall Z \leq g(|X|)\varphi(X,Y,Z) \tag{10}$$

where f and g are polynomials and $\varphi(X,Y,Z)$ is an open formula of **VPV**. Since the satisfiability problem is **NP**-complete, there is a **VPV** function $F(X,Y)$ such that $F(X,Y)$ is a satisfiable propositional formula iff $\neg\forall Z \leq g(|X|)\varphi(X,Y,Z)$. Further, the equivalence is provable in **VPV** (see (2)).

Now suppose C is a circuit such that for every satisfiable propositional formula A of length at most n, $C(A)$ is a satisfying assignment for A (n will be specified later). Then for all X,Y such that $|F(X,Y)| \leq n$ we have

$$\forall Z \leq g(|X|)\varphi(X,Y,Z) \;\Leftrightarrow\; \neg SAT(C(F(X,Y)), F(X,Y))\ . \tag{11}$$

We can find a suitable circuit C by using (6) and computing the advice i_n used in the proof of Theorem 3.4. We can compute i_n by successive **NP** queries, for $j = 1, 2, ...k$

$$\exists T_1, A_1, \cdots T_j, A_j \leq n$$
$$\bigwedge_{i=1}^{j}[SAT(T_i, A_i) \wedge \neg SAT(C_i(n, A_1, \cdots, A_{i-1}, T_1, \cdots, T_{i-1})(A_i), A_i)]\ .$$

Then i_n is the smallest j such that the answer to the query is 'NO'. (**VPV** easily proves the existence of i_n because there are only a constant k number of possible choices for it.) Now one more **NP** query suffices. Thus, setting $j = i_n$, $X \in L$ iff

$$\exists Y \leq f(|X|) \exists T_1, A_1, \cdots T_{j-1}, A_{j-1} \leq n$$
$$\bigwedge_{i=1}^{j-1} [SAT(T_i, A_i) \wedge \neg SAT(C_i(n, A_1, \cdots, A_{i-1}, T_1, \cdots, T_{i-1})(A_i), A_i)]$$
$$\wedge \neg SAT(C_j(n, A_1, \cdots, A_{j-1}, T_1, \cdots, T_{j-1})(F(X,Y)), F(X,Y)) \ .$$

Finally we can set $n = h(|X|)$ for a suitable polynomial h.

It is easy to see that this argument can be formalized in **TVPV**. $\square$

It turns out that Theorem 3.5 can also be proved as an immediate consequence of Theorem 3.4 and the following complexity-theoretic result.

Theorem 3.6 (Jeřábek)**.** [1] *If* **coNP** $\subseteq$ **NP**$/O(1)$ *then the Polynomial Hierarchy collapses to the Boolean Hierarchy. The proof can be formalized in* **VPV**.

Proof: As in the proof of Theorem 3.5, to show **PH** = **BH** it suffices to show that the language L satisfying (10) is in the Boolean Hierarchy. Let $F(X, Y)$ be as in that proof.

Assume that **coNP** $\subseteq$ **NP**$/O(1)$, so UNSAT is in **NP**$/O(1)$. Then there is a $\mathbf{\Sigma}_1^B$ formula $UNS(A, i)$ and a number k such that for every length n there is advice $i \leq k$ and for every propositional formula A of length at most n,

$$A \text{ is unsatisfiable iff } UNS(A, i) \tag{12}$$

Then we claim that

$$X \in L \iff \bigvee_{i=0}^{k} [\exists Y \leq f(|X|)\ UNS(F(X,Y), i)$$
$$\wedge\ \forall A, T \leq h(|X|),\ UNS(A, i) \rightarrow \neg SAT(T, A)] \tag{13}$$

[1] We are grateful to Emil Jeřábek for supplying the proof of this theorem, in response to an open problem stated in an earlier version of this paper.

where $h(m)$ is a polynomial upper bound on $|F(X,Y)|$ for all X of length m and all Y of length at most $f(m)$. It follows from this Claim that $L \in \mathbf{BH}$, since the RHS has the form $\bigvee_i [R_i \wedge S_i]$ where R_i is in **NP** and S_i is in **coNP**.

To prove the Claim, first assume that $X \in L$. Let $n = h(|X|)$ and let $i \leq k$ be the advice such that (12) holds for all A of length at most n. Then the RHS of (13) holds by (10) and the stated property of $F(X,Y)$.

Conversely, suppose that X satisfies the RHS of (13), let i satisfy the disjunction, and let Y satisfy the existential quantifier for this i. Then the second conjunct implies that this i gives "sound advice" for $UNS(A,i)$, $|A| \leq h(|X|)$, and hence $X \in L$ by (10) and the stated property of $F(X,Y)$.

Note that **VPV** proves (13) from (10) and the properties of $F(X,Y)$ and $UNS(A,i)$. □

4 Witnessing Theorems

The notions surrounding definability of multivalued functions (which we call search problems) in bounded arithmetic were introduced in [BKT93].

A *search problem* Q_R is a multivalued function with graph $R(\vec{x},\vec{X},Z)$, so

$$Q_R(\vec{x},\vec{X}) = \{Z \mid R(\vec{x},\vec{X},Z)\} \ .$$

Here the arity of either or both of $\vec{x},\vec{X}$ may be zero. We assume here that the search problem is *total*, meaning that the set $Q_R(\vec{x},\vec{X})$ is non-empty for all $\vec{x},\vec{X}$. The search problem is a *function problem* if $|Q_R(\vec{x},\vec{X})| = 1$ for all $\vec{x},\vec{X}$.

A (single-valued) function $F(\vec{x},\vec{X})$ *solves* Q_R if

$$F(\vec{x},\vec{X}) \in Q_R(\vec{x},\vec{X})$$

for all $\vec{x},\vec{X}$. More generally, a search problem $Q_{R'}$ *solves* Q_R if $Q_{R'}$ is total and

$$Q_{R'}(\vec{x},\vec{X}) \subseteq Q_R(\vec{x},\vec{X})$$

for all $\vec{x},\vec{X}$.

We say that a search problem Q_R is $\mathbf{\Sigma}_i^B$-definable in a theory $\mathcal{T}$ if there is a $\mathbf{\Sigma}_i^B$-formula ψ_R such that

$$\psi_R(\vec{x},\vec{X},Z) \to R(\vec{x},\vec{X},Z)$$

and

$$\mathcal{T} \vdash \exists Z \psi_R(\vec{x}, \vec{X}, Z) \ .$$

For example, a search problem is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^1$ iff it is solvable by a polynomial time function.

The standard notation $\mathbf{P}^{\mathbf{\Sigma}_i^p}$ refers to the class of decision problems solvable in polynomial time by accessing an oracle for a problem in $\mathbf{\Sigma}_i^p$. $\mathbf{P}^{\mathbf{\Sigma}_i^p}[q(n)]$ is the same, except that the number of oracle queries in a computation is limited to $O(q(n))$, where n is the length of the input. We use $\mathbf{FP}^{\mathbf{\Sigma_i^P}}$ and $\mathbf{FP}^{\mathbf{\Sigma_i^P}}$[q(n)] for the classes of *search problems* solvable in the same way. [BKT93] introduced the notation $\mathbf{FP}^{\mathbf{\Sigma_i^P}}[wit, q(n)]$ for the class of search problems solvable in polynomial time by making $O(q(n))$ witness queries to an oracle for some problem in $\mathbf{\Sigma}_i^p$. Here a *witness query* returns 1 together with a witness to the query if the answer is 'YES', and returns 0 if the answer is 'NO'. (A witness to an **NP** query is a certificate which can be used to verify a positive answer in polynomial time, and for $i > 1$ a witness to a $\mathbf{\Sigma}_i^p$ query allows a positive answer to be verified in $\mathbf{\Sigma}_{i-1}^p$.) When witness queries are allowed, the machine must output a solution to the search problem no matter which witnesses to the positive queries are returned.

Note that search problems in $\mathbf{FP}^{\mathbf{\Sigma_i^P}}[wit]$ are already in $\mathbf{FP}^{\mathbf{\Sigma_i^P}}$; i.e. witness queries do not help if the number of queries is unrestricted. This is because by self reducibility, a witness for a positive query can be found using polynomially many decision queries.

In terms of two-sorted theories, the following results (among others) are known, or can be inferred from the corresponding single-sorted results. Here for $i \geq 1$ the theory $\mathbf{TV}^i$ [Coo05, CN06] is the two sorted analog of T_2^i, and $\mathbf{TV}^0$ is a finitely axiomatizable theory for polynomial time (**VPV** is a conservative extension of $\mathbf{TV}^0$) [Coo05, CN06].

Theorem 4.1. *(i) [Bus86] For $i \geq 1$, a search problem Q is $\mathbf{\Sigma}_i^B$-definable in $\mathbf{V}^i$ iff $Q \in \mathbf{FP}^{\mathbf{\Sigma_{i-1}^P}}$. For the only if direction, $\mathbf{V}^i$ proves the correctness of the witness oracle algorithm.*

(ii) [Kra93, Kra95] For $i \geq 1$, a search problem Q is $\mathbf{\Sigma}_{i+1}^B$-definable in $\mathbf{V}^i$ iff $Q \in \mathbf{FP}^{\mathbf{\Sigma_i^P}}[wit, \log n]$. For the only if direction, $\mathbf{V}^i$ proves the correctness of the witness oracle algorithm.

(iii) [Bus90, Coo05] For $i \geq 0$, a search problem Q is $\mathbf{\Sigma}_{i+1}^B$-definable in $\mathbf{TV}^i$ iff $Q \in \mathbf{FP}^{\mathbf{\Sigma_i^P}}$.

(iv) [Pol99] For $i \geq 0$, a search problem Q is $\mathbf{\Sigma}^B_{i+2}$-definable $\mathbf{TV}^i$ iff $Q \in \mathbf{FP}^{\mathbf{\Sigma^P_{i+1}}}[wit, 1]$.

The case (iv) in the above theorem follows from Theorem 54 in [Pol99], where the 'only if' direction is proved by a cut-elimination argument. We are interested in this direction for the case $i = 0$, so we give a simple proof of this case based on the KPT theorem. Since **VPV** is a conservative extension of $\mathbf{TV}^0$, it suffices to prove the theorem for **VPV**. Since both directions are interesting, we prove the 'if' direction also.

Theorem 4.2. *A search problem Q is $\mathbf{\Sigma}^B_2$-definable in* **VPV** *iff $Q \in \mathbf{FP}^{\mathbf{NP}}[wit, 1]$. For the only if direction,* **VPV** *proves the correctness of the witness oracle algorithm.*

Proof: For the direction $\Longrightarrow$, assume that $Q = Q_R$ is $\mathbf{\Sigma}^B_2$-definable in **VPV**, so

$$\mathbf{VPV} \vdash \exists Z \psi_R(\vec{x}, \vec{X}, Z) \tag{14}$$

where ψ_R is $\mathbf{\Sigma}^B_2$ and

$$\psi_R(\vec{x}, \vec{X}, Z) \rightarrow R(\vec{x}, \vec{X}, Z) \ .$$

For some open formula φ, (14) can be written

$$\mathbf{VPV} \vdash \exists Z \exists Y \forall W \varphi(\vec{x}, \vec{X}, Z, Y, W) \tag{15}$$

where all quantifiers are bounded. By Theorem 3.1 (KPT) there are **VPV** functions $F_1, \ldots, F_k$ and $G_1, \ldots, G_k$ such that (thinking $Z = F_i()$ and $Y = G_i()$, and suppressing the arguments $\vec{x}, \vec{X}$)

$$\begin{aligned} \mathbf{VPV} \vdash \varphi(F_1, G_1, W_1) \vee \varphi(F_2(W_1), G_2(W_1), W_2) \vee \ldots \\ \vee\, \varphi(F_k(W_1, \ldots, W_{k-1}), G_k(W_1, \ldots, W_{k-1}), W_k) \ . \end{aligned} \tag{16}$$

Now a polynomial time witness oracle machine with an **NP** oracle, given inputs $\vec{x}, \vec{X}$, can compute Z satisfying $\psi_R(\vec{x}, \vec{X}, Z)$ as follows. First ask the oracle whether $\exists W_1 \neg \varphi(F_1, G_1, W_1)$. If 'NO', then output F_1. If 'YES', then let W_1 be a witness, and ask the oracle whether

$$\exists W_2 \neg \varphi(F_2(W_1), G_2(W_1), W_2) \ .$$

If 'NO', then output $F_2(W_1)$. If 'YES', then let W_2 be a witness, and continue. By (16) we are guaranteed a 'NO' answer after some number $i \le k$ queries, so output $F_i(W_1, \ldots, W_{i-1})$. Then **VPV** proves that the output satisfies the quantifier $\exists Z$ in (15), and hence solves the search problem Q_R.

For the direction $\Longleftarrow$, assume that the oracle Turing machine M solves $Q(\vec{x}, \vec{X})$ in polynomial time with at most k witness queries to the **NP** language L, for some constant k. Let $Comp(\vec{x}, \vec{X}, W)$ be a $\mathbf{\Pi}_1^B$-formula asserting that W codes a halting computation of M on input $\vec{x}, \vec{X}$. Thus W codes the sequence of configurations of M on input $\vec{x}, \vec{X}$, and for each query to L it provides the answer to the query. If the answer is 'YES' it provides a witness for the query (the correctness of the witness can be checked by a $\mathbf{\Sigma}_0^B$-formula). Note that a 'NO' answer can be verified using the universal string quantifiers allowed for $\mathbf{\Pi}_1^B$-formulas.

Now define

$$\psi(\vec{x}, \vec{X}, Z) \equiv \exists W \le t,\ Comp(\vec{x}, \vec{X}, W) \wedge Out(Z, W)$$

where $Out(Z, W)$ is a $\mathbf{\Sigma}_0^B$-formula asserting that Z is the output of the computation W and $t = t(\vec{x}, \vec{X})$ is a suitable bounding term. To show that Q is $\mathbf{\Sigma}_2^B$-definable in **VPV** is suffices to show that **VPV** proves $\exists Z \psi(\vec{x}, \vec{X}, Z)$. Since it is easy to show that every computation W has an output Z satisfying $Out(Z, W)$, it suffices to show

$$\mathbf{VPV} \vdash \exists W Comp(\vec{x}, \vec{X}, W)\ .$$

To do this, recall that k is an upper bound on the number of queries made by M during any computation. We define, for $0 \le i \le k+1$, the $\mathbf{\Pi}_1^B$-formula $Comp_i(\vec{x}, \vec{X}, W)$ to assert that W codes a partial computation of M on input $\vec{x}, \vec{X}$ which is either halting, or includes at least i queries, and ends on an unanswered query. It suffices to show that for each i,

$$\mathbf{VPV} \vdash \exists W Comp_i(\vec{x}, \vec{X}, W) \tag{17}$$

because by assumption $Comp_{k+1}$ is equivalent to $Comp$, so we may replace $Comp$ by $Comp_{k+1}$.

For $i = 0$, (17) follows from the fact that **VPV** proves the existence of a computation for any polytime (nonoracle) Turing machine. It suffices to show

$$\mathbf{VPV} \vdash Comp_i(\vec{x}, \vec{X}, W) \rightarrow \exists W' Comp_{i+1}(\vec{x}, \vec{X}, W')$$

Arguing in **VPV**, assume $Comp_i(\vec{x}, \vec{X}, W)$. If W is a halting computation we are done. Otherwise the answer to the final query of W must be either 'YES' or 'NO'. If the answer is 'YES', then by definition there is a witness to the answer, and using this witness the computation can be continued until the next query. If the answer is 'NO', then again the computation can be continued until the next query. $\square$

5 Results for $\mathbf{V}^1$ and $\mathbf{V}^2$

We now apply Theorem 4.1 to infer analogs of Theorems 3.4 and 3.5 for the theories $\mathbf{V}^1$ and $\mathbf{V}^2$. We also note that Theorem 3.5 and the the only if direction of Theorem 3.4 for **VPV** follow rather easily from Theorem 4.2, using the proof method of the next theorem.

Theorem 5.1. *(i)* $\mathbf{V}^1$ *proves* $\mathbf{NP} \subseteq \mathbf{P/poly}$ *iff* $\mathbf{V}^1$ *proves* $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$.

(ii) If $\mathbf{V}^1$ *proves* $\mathbf{NP} \subseteq \mathbf{P/poly}$ *then* $\mathbf{V}^1$ *proves that the Polynomial Hierarchy collapses to* $\mathbf{P^{NP}}[\log n]$.

(iii) If $\mathbf{V}^2$ *proves* $\mathbf{NP} \subseteq \mathbf{P/poly}$ *then* $\mathbf{V}^2$ *proves that the Polynomial Hierarchy collapses to* $\mathbf{P^{NP}}$.

Proof: (i) ($\Longrightarrow$): Assume that $\mathbf{V}^1$ proves $\mathbf{NP} \subseteq \mathbf{P/poly}$ as in (4). Let $\eta(n, C)$ be the formula

$$\forall A \leq n \forall T \leq n,\ SAT(T, A) \ \rightarrow\ SAT(C(A), A)\ . \tag{18}$$

Then $\mathbf{V}^1$ proves $\exists C \eta(n, C)$, and hence the search problem Q given by

$$Q(n, C) \Leftrightarrow \eta(n, C)$$

is $\mathbf{\Sigma}_2^B$-definable in $\mathbf{V}^1$. Hence by Theorem 4.1 part (ii) for $i = 1$, Q is in $\mathbf{FP^{NP}}[wit, \log n]$, provably in $\mathbf{V}^1$. Let M be a polynomial time witness oracle Turing machine solving Q by making $O(\log n)$ witness queries to some **NP** problem, such that $\mathbf{V}^1$ proves that any circuit C output by M on input n satisfies (18). Here is how $\mathbf{V}^1$ proves the existence of a correct computation of M for each input n. We distinguish between the decision part of a witness query, whose answer is 0 or 1, and the witness part of the answer in case the decision answer is 1. Note that the sequence of queries (and their answers) may not be determined by the input n, because more than one witness answer to a positive witness query may be possible. By the $\mathbf{\Sigma}_1^B$ number-MAX principle, $\mathbf{V}^1$ proves for each input n the existence of a computation Z of M in which the sequence of 0-1 answers to the decision part of the queries is lexicographically the largest possible (here the $\mathbf{\Sigma}_1^B$ formula verifies the 1 answers and their witnesses, but does not verify the 0 answers). It follows that the 0 answers for this computation are correct, even though they have not been verified (the first wrong 0 answer would yield a lexicographically larger sequence of answers).

According to Definition 3.3, the assertion $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$ is equivalent to saying that for every $\mathbf{\Sigma}_1^B$ formula $\varphi(X)$ there is a $\mathbf{\Sigma}_1^B$ formula $\psi(X)$ and a polynomial $f(n)$ such that

$$\forall n \exists i \leq f(n), |X| = n \rightarrow [\neg\varphi(X) \leftrightarrow \psi(X, i)] \ . \tag{19}$$

As before, since the unsatisfiability problem is complete for $\mathbf{coNP}$, it suffices to show that $\mathbf{V}^1$ proves (19) for the case that $\varphi(X)$ is $\exists T \leq |X| \ SAT(T, X)$.

Given n, the advice string needed to show that a given unsatisfiable formula A of length n is indeed unsatisfiable is the lexicographically largest possible string of 0-1 query answers described above for the computation of M on input n. We define the $\mathbf{\Sigma}_1^B$ formula $\psi(A, i)$ in (19) to assert that some computation Z of M on input n, in which the query answers are those coded by i in binary, computes a circuit C such that $C(A)$ is not a satisfying assignment for A. Then $\mathbf{V}^1$ proves that C satisfies (18), and hence A is unsatisfiable.

(i) ($\Longleftarrow$): Assume that $\mathbf{V}^1$ proves $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$ as in (19). We argue that $\mathbf{V}^1(\mathbf{VPV})$ proves $\mathbf{NP} \subseteq \mathbf{P}/\mathbf{poly}$ by a slight modification of the proof of the 'if' direction of Theorem 3.4. (Note that $\mathbf{V}^1(\mathbf{VPV})$ is a conservative extension of $\mathbf{V}^1$.) The sets U and V_i are as in that proof, except that i ranges up to $O(\log n)$ rather than the constant k. The set I in (9) is specified by a string variable I listing its members. Instead of a separate function F_I for every I, we now have a single $\mathbf{VPV}$ function $F(I, \tilde{X}, Y)$, where $\tilde{X}$ is an array giving witnesses to the validity of all conjuncts in the antecedent of (9). (The existence of a polynomial time such F now follows from the Buss witnessing theorem for $\mathbf{V}^1$.) The advice required to compute U is the same as before, except longer (but still polynomial in length), since it requires information for $O(\log n)$ values of i instead of a constant k values.

(ii): Assume that $\mathbf{V}^1$ proves $\mathbf{NP} \subseteq \mathbf{P}/\mathbf{poly}$ as in (4). Let M be the witness oracle machine described in the proof of (i) ($\Longrightarrow$) above. Thus on input n, M computes a circuit C satisfying (18).

To show that $\mathbf{V}^1(\mathbf{VPV})$ proves that the Polynomial Hierarchy collapses to $\mathbf{P}^{\mathbf{NP}}[\log n]$, it suffices to show $\mathbf{\Sigma}_2^p \subseteq \mathbf{P}^{\mathbf{NP}}[\log n]$. We argue as in the proof of Theorem 3.5, and assume that $L \in \mathbf{\Sigma}_2^p$, so (10) holds. The idea is to use the circuit C solving SAT computed by M, so that (11) holds for this C. However there is a difficulty in finding C, because we are trying to show L is in $\mathbf{P}^{\mathbf{NP}}[\log n]$, so that only decision queries are allowed, but M requires witness queries to find C. We proceed as follows. To determine whether a given string X is in L, we start by asking a sequence of $O(\log n)$ **NP** decision queries to determine the lexicographically largest possible sequence S of 0-1 answers to the witness queries of the computation of M (see the proof of (i) ($\Longrightarrow$)) on input n (where n is a suitable polynomial in $|X|$). Now just one more **NP** decision query is needed. By (10) and (11), X is in L iff there exists $Y \leq f(|X|)$ and there exists a computation of M on input n such that the decision part of the answers to the witness queries of the computation are the sequence S (only positive query answers and their witnesses need be verified) such that if C is the resulting circuit computed by M then $\neg SAT(C(F(X,Y)), F(X,Y))$.

(iii): The proof is similar to the proof of (ii), but instead of part (ii) of Theorem 4.1 we use part (i): If Q is $\mathbf{\Sigma}_2^B$-definable in $\mathbf{V}^2$ then Q is in $\mathbf{FP}^{\mathbf{NP}}$. $\square$

Just as Theorem 3.5 follows from Theorem 3.4 and Theorem 3.6, an alternative proof of Theorem 5.1 (ii) (except possibly the provability of the conclusion) can be obtained from the conclusion $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$ of Theorem 5.1 (i) and the following complexity-theoretic result.

Theorem 5.2. *If* $\mathbf{coNP} \subseteq \mathbf{NP}/O(\log n)$ *then the Polynomial Hierarchy collapses to* $\mathbf{P}^{\mathbf{NP}}[\log n]$.

Proof: We argue as in the proof of Theorem 3.6, except now the formula $UNS(A,i)$ needs $O(\log n)$ bits of advice instead of constant advice. Thus the constant k in (13) becomes a function $k(n) = n^{O(1)}$ (where $n = |X|$). From this we see that L can be computed in polynomial time with polynomially many *parallel* queries to an **NP** oracle. From results in [BH91] it follows that $O(\log n)$ *serial* queries suffice, so $L \in \mathbf{P}^{\mathbf{NP}}[\log n]$ as required. $\square$

6 Proof Systems with Advice

In this section we introduce **propositional proof systems with advice**, a new concept in proof complexity suggested by results from the earlier sections.

Before we give formal definitions let us explain one possible motivation in detail. It is well-known that the provability of a $\mathbf{\Pi}_1^B$ formula φ in **VPV** implies the existence of polynomial size Extended Frege **EF** proofs of the propositional translations $\langle\varphi\rangle_n$ of φ (in fact, the proofs can be constructed by a p-time algorithm). This goes back to [Coo75] (a background for this can be also found in [Kra95, CN06]).

Using this relation between **VPV** and **EF** one can show that the provability of $\mathbf{NP} = \mathbf{coNP}$ in **VPV** would imply that **EF** is polynomially bounded, i.e. that it proves all tautologies by proofs of polynomial size. Let us repeat the argument for the reader's benefit and a later reference. The assumption that **VPV** proves $\mathbf{NP} = \mathbf{coNP}$ implies (via the Herbrand theorem) that for some p-time function F **VPV** proves

$$SAT(T, A) \to SAT(F(A), A) \ . \tag{20}$$

Hence the propositional formulas translating this implication for all lengths

$$\langle SAT(T, A)\rangle_{n,m}(p, q) \to \langle SAT(F(A), A)\rangle_m(q) \tag{21}$$

have p-size **EF** proofs. Here p, q are tuples of atoms of lengths n and m respectively, representing the strings T, A.

Now let $B(p)$ be any tautology with n atoms. Combine the instance of (21) with $q := a$ where a is an m-tuple of bits representing the formula $\neg B(p)$ with a short proof of (by a straightforward evaluation)

$$\neg\langle SAT(F(A), A)\rangle_m(a)$$

to deduce

$$\neg\langle SAT(T, A)\rangle_{n,m}(p, a) \ .$$

Then, using a general and shortly provable fact

$$\neg\langle SAT(T, A)\rangle(p, a) \to B(p) \tag{22}$$

deduce formula $B(p)$.

As a corollary we get that proving any one lower bound for **EF** (i.e. proving that **EF** is not "super") implies unprovability of (20) for all possible functions F and hence the consistency of $\mathbf{NP} \neq \mathbf{coNP}$ with **VPV** .

What we are after in this section is definition of a "proof system" $\mathbf{EF}/O(1)$, **EF** with finite advice, that would play a similar role for the non-uniform case $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$ considered in Section 3. That is, the provability of the inclusion $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$ would imply that $\mathbf{EF}/O(1)$ is polynomially bounded. In particular, a lower bound for proofs in $\mathbf{EF}/O(1)$ of any one particular sequence of tautologies would imply that $\mathbf{coNP} \not\subseteq \mathbf{NP}/O(1)$ is consistent with **VPV** .

Definition 6.1. *Let $k = k(n)$ be a function on natural numbers. A propositional proof system with k bits of advice, abbreviated a pps/k system, is a binary relation $Q(X, Y)$ such that*

$$X \in TAUT \quad \textit{iff} \;\; \exists Y Q(X, Y)$$

and $Q(X, Y)$ is computable in time polynomial in $(|X| + |Y|)$ with $k(n)$ bits of advice, where $n = |X|$ and the advice depends only on n.

Here $TAUT$ is the set of propositional tautologies in the DeMorgan language.

The restriction that the advice being used in computing $Q(X, Y)$ depends only on n is motivated by Definition 3.3, where the advice allowed to determine whether $X \in L$ depends only on the length of X. Also, by allowing the advice to depend only on n we keep in line with the general idea that complexity of proofs is measured in terms of the length of the formula being proved.

In the classical Cook-Reckhow setting we can equivalently define proof systems as functions. It turns out that the straightforward definition (Part 1 of the next definition) is not necessarily equivalent with Definition 6.1 in the presence of advice, and one needs to define a variant of the usual concept (Part 2 in the next definition).

Definition 6.2. Part 1. *Let $k = k(n)$ be a function on natural numbers. A functional proof system with k bits of advice is a function $P(Y)$ from the set of all strings (in some finite alphabet of size at least 2) whose range is exactly the set $TAUT$, and such that P is computable by a polynomial time algorithm using $k(n)$ bits of advice on inputs of length n.*

Part 2. *A length-determined functional proof system with k bits of advice, abbreviated a ldpps/k system, is the same as a functional proof system P, except there is a function $\ell(m)$ such that $P(Y)$ has length $\ell(m)$ for all Y of length m, and further the advice allowed to compute $P(Y)$ is $k(\ell(m))$ bits and depends only on $\ell(m)$ (and not otherwise on m).*

When discussing functional proof systems at the same time as proof systems according to Definition 6.1 we sometimes refer to the latter as relational proof systems.

Lemma 6.3. *Relational proof systems and length-determined functional proof systems are p-equivalent concepts. (In the case of relational systems Q, we interpret the pair (X, Y) to be a proof of X iff $Q(X, Y)$ holds.)*

Proof: Given a ldpps/k system P we define $Q(X, Y)$ by the condition $X = P(Y)$. Then Q is a p-equivalent relational system with k bits of advice.

Conversely given a relational system Q with k bits of advice, we want to define a p-equivalent ldpps/k system P'. To define $P'(Z)$, the idea is to interpret Z to be an ordered pair (X, Y) and define $P'(Z) = X$. However in order to make P' length-determined we need to use a pairing function with the property that the length of the pair (X, Y) determines the lengths of both X and Y. This can be done by using a standard pairing function $\langle n, m\rangle$ on natural numbers (for example, $\langle n, m\rangle := (1/2)(n+m)(n+m+1)$), and defining the ordered pair (X, Y) for $|X| = n$ and $|Y| = m$ to be the usual ordered pair padded by some canonical symbol so the result has length $\langle n, m\rangle$. The resulting ldpps/k system P' p-simulates $Q(X, Y)$ because the padded pair (X, Y) has length bounded by a polynomial in the sum of the lengths of X and Y. $\square$

In general it seems difficult to turn a functional proof system P with advice into a p-equivalent length-determined system. However this can be done when P is a classical Cook-Reckhow proof system (i.e. $k = 0$).

Lemma 6.4. *For $k = 0$ (no advice) every functional pps/k system P is p-equivalent to some ldpps/k system P'*

Proof: Given P, let Q be as in the first part of the proof of Lemma 6.3: i.e. $Q(X, Y)$ iff $X = P(Y)$. Then Q is an equivalent relational system even if P is not length-determined, because $k = 0$. Now let P' be defined as in the second part of the proof of the Lemma. $\square$

Next we relate pps/k systems to Definition 3.3.

Theorem 6.5. *For every function $k(n)$, $TAUT \in \mathbf{NP}/k(n)$ iff some pps/$k(n)$ system is polynomially bounded.*

Proof: This is immediate from Definitions 3.3 and 6.1. $\square$

Before we turn to the question of how to define an extension of a particular classical proof system by allowing it to use advice, we make one general observation showing that functional proof systems with advice are in principle quite powerful, at least if we do not require that they be length-determined. The second part of the theorem has been suggested to us by a question of P.Pudlák.

Theorem 6.6. *There is a functional proof system P with one bit of advice that p-simulates all classical Cook-Reckhow proof systems.*

In fact, P simulates all functional pps/$O(\log n)$ systems (i.e. it is optimal in this class). The simulation of a functional pps/$k(n)$ system (for $k(n) \leq O(\log(n))$) by P is computed by a polynomial time algorithm using $k(n)$ bits of advice.

Proof: First a convention. Polynomial time Turing machines are assumed to have an explicit clock limiting the time of the computation. Furthermore, if a machine has a description encoded by natural number Q then its time bound is at most n^Q (this can be arranged by enlarging the description of Q in some canonical way if necessary).

We define the functional pps/1 system P to operate as follows. Upon receiving input w of length m it interprets m is an ordered quadruple $\langle m_1, m_2, m_3, m_4 \rangle$ of numbers and string w as being the concatenation of strings $w_1, \ldots, w_5$ with $|w_i| = m_i$ for $i = 1, \cdots, 4$ and $|w_5| = \langle m_1, m_2, m_3, m_4 \rangle - (m_1+m_2+m_3+m_4)$. Then P proceeds as follows:

1. It discards w_4 and w_5 (these are just strings of junk symbols that boost the input length.)
2. It interprets m_3 as an algorithm Q.
3. It writes down number m_2 in binary; we shall denote the resulting string $a(m_2)$ to avoid a confusion between strings and their lengths. The length of $a(m_2)$ is thus $\lceil \log(m_2) \rceil$.
4. It checks that m, the length of the input, satisfies $m \geq m_1^Q$.

5. If any of the conditions above fails, P halts and produces some default tautology, e.g. 1.

   Otherwise P gets the one bit of advice: The advice equals 1 iff

   - Q encodes an algorithm that uses $\lceil \log(m_2) \rceil$ bits of advice on inputs of length m_1.
   - Q using advice $a(m_2)$ is sound on inputs of length m_1 (i.e. it outputs only tautologies).

6. If the advice is 0, P outputs the default tautology 1. Otherwise it simulates the run of Q on input w_1 and advice $a(m_2)$ and outputs the output of this simulation (a tautology).

Note that this advice bit depends only on m, the input length. (We remark that the bit can be computed in **coNP**.)

Claim 1: *Algorithm P runs in polynomial time and on every input produces a tautology.*

It is clear that P produces a tautology because it produces either 1 or an output of a sound (on given input length) proof system. The biggest contribution to the run time of P is the simulation of Q which can be done in the square of the time of Q, which is bounded by $|w_1|^Q \leq m$.

Claim 2: *The functional pps/1 system P simulates any functional pps/$k(n)$ if $k(n) \leq O(\log n)$. The simulation is computed by a p-time function using $k(n)$ bits of advice (i.e. it is not a p-simulation if $k \neq 0$).*

In particular, P p-simulates all Cook-Reckhow proof systems.

Given a functional pps/$k(n)$ system Q with $k(n) \leq O(\log n)$ we must define its simulation f by P. On input v of length n the value $f(v)$ is computed as follows:

1. Put $w_1 := v$.

2. Let a_n be the advice string of length $k(n)$ that Q uses on inputs of length n (here the simulation function needs advice, if Q uses it).

   Put w_2 to be the string of m_2 ones, where m_2 is the natural number encoded by a_n. Note that $m_2 \leq 2^{k(n)} \leq n^{O(1)}$.

3. Put $m_3 = Q$ (this is a constant) and put w_3 to be the string of m_3 zeros.

4. Put $m_4 = n^Q$ and put w_4 to be a string of m_4 ones. (This guarantees that item 4 in the definition of P is satisfied, namely $m \geq m_1^Q$.)

5. Output $f(v) := w$, where w is the concatenation of w_1, w_2, w_3, w_4 followed by $\langle m_1, m_2, m_3, m_4 \rangle - (m_1 + m_2 + m_3 + m_4)$ zeros (so that the length of w determines $m_1, \ldots, m_4$).

□

Now we turn to the question how to sensibly define an extension of a classical proof system Q by $k := k(n)$ bits of advice, a system that we would denote Q/k. Although the definition can be given for a general "sufficiently strong" proof system Q (see the remarks after Definition 6.7) we shall concentrate here on **EF** only. Motivated by the proof of Theorem 3.4, the idea behind the definition is that the advice will provide **EF** with a form of witnessing of its $\mathbf{\Sigma}_2^B$ consequences. By virtue of Theorem 3.1 the $\forall\mathbf{\Sigma}_2^B$ consequences of **VPV** are witnessed by disjunctions in the KPT form. As we want to define $\mathbf{EF}/k$ in an elementary way without a reference to **VPV** or bounded arithmetic in general, we consider directly the **EF**-provability of the (propositional translations of the) KPT disjunctions.

Let $\varphi(X, Y, Z)$ be an open formula in the language of **VPV**. Fixing bounds $t(n)$ and $s(n)$ on the lengths of Y and Z in terms of $n := |X|$ respectively, symbol $\langle\varphi\rangle_{n,t(n),s(n)}$ denotes the propositional translation of φ for inputs of the respective lengths. We shall assume that the bounds on the lengths of Y and Z are implicit in φ and we shall skip an explicit reference to $t(n)$ and $s(n)$ in the notation.

Consider a propositional disjunction of the form

$$\langle\varphi\rangle_n(p, C_1(p), q^1) \vee \langle\varphi\rangle_n(p, C_2(p, q^1), q^2) \vee \ldots \\ \vee \langle\varphi\rangle_n(p, C_\ell(p, q^1, \ldots, q^{\ell-1}), q^\ell) \quad (23)$$

where p is an n-tuple of propositional atoms, $q^1, \ldots, q^\ell$ are $s(n)$-tuples of atoms (all distinct), and $C_1, \ldots, C_\ell$ are circuits with $t(n)$ output bits and with the indicated inputs. In fact, there are other atoms - the so called extension variables - present in the disjunction that we shall not show explicitly and we shall discuss them only now. The propositional formulas translate **VPV** formulas (in particular, p-time relations represented by canonical circuits) and also talk about circuits $C_1, \ldots, C_\ell$. For propositional logic, circuits are represented by a sequence of extension variables e, one for each subcircuit, together with their defining relations. Let $Ext(e)$ be the conjunction of all these defining relation for all variables in the tuple e corresponding to all (sub)circuits occurring in the propositional formula. Then the actual form of the propositional translation is

$$Ext(e) \ \rightarrow \ D(p, q^1, \ldots, q^\ell, e)$$

where D is the disjunction above (showing e explicitly). However, in order to avoid extensive notation, we do not display the extension variables e and formula $Ext(e)$ in the notation in Definition 6.7.

Call any disjunction such as above **an** (n, ℓ)**-disjunction from** φ.

Definition 6.7. *Let a function $k := k(n)$ be given. Put $\ell(n) := 2^{k(n)}$.*

An extension of **EF** *by $k(n)$ advice bits is determined by an open* **VPV** *formula $\varphi(X, Y, Z)$ and a polynomial time function F that upon receiving inputs $1^{(n)}$ and $1^{\ell(n)}$, outputs an* **EF***-proof of an $(n, \ell(n))$-disjunction from φ. Denote by $\mathbf{EF}_{\varphi,F}$ the system determined by φ and F.*

The advice used by $\mathbf{EF}_{\varphi,F}$ specifies, for each n, the minimal $i \leq \ell(n)$ such that already the disjunction $B_{n,i}$ of the first i disjuncts in the (n, ℓ)-disjunction computed by F is a tautology.

A proof of formula A in $\mathbf{EF}_{\varphi,F}$ is a triple $(1^{(n)}, 1^{\ell(n)}, W)$ where $n = |A|$, W is a **EF***-proof of*

$$B'_{n,i} \rightarrow A$$

and $B'_{n,i}$ is any simple substitution instance of $B_{n,i}$, i.e. only substitutions for (some) atoms of constants or other atoms are allowed.

The role of function F is to replace an intended logic assumption that the $\mathbf{\Sigma}_2^B$-formula $\exists Y \forall Z \varphi(X, Y, Z)$ is provable in **VPV**; it is known that the latter implies the existence of F. The dependence of F on $1^{\ell(n)}$ is in order to allow for superpolynomial size proofs.

Although we have formulated the definition only for **EF**, analogously one can define Q/k for any "sufficiently strong" classical proof system Q. The qualification sufficiently strong would mean that (1) Q shortly proves the uniqueness of a computation of a circuit on an input (p-simulation of resolution by Q suffices for this), and (2) that it is closed under a feasible weakening rule: a Q-proof of formula $B \to A$ can by constructed by a p-time algorithm given a Q-proof of A and a formula B.

We shall call any system $\mathbf{EF}_{\varphi,F}$ an "$\mathbf{EF}/k(n)$ proof system".

At the beginning of this section we outlined a proof that if **VPV** proves **NP** = **coNP**, then **EF** is a polynomially bounded proof system. The following theorem shows that under a weaker assumption, some $\mathbf{EF}/O(1)$ proof system is polynomially bounded. This theorem is stated for the theory **VPV**, although it is clear that an analogous result holds also for the theory $\mathbf{V}^1$ and $O(\log n)$ advice bits.

Theorem 6.8. *Assume that* **VPV** *proves that* $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$. *Then some* $\mathbf{EF}/O(1)$ *proof system has polynomial size proofs of all tautologies.*

Proof: From the hypothesis of the theorem, we conclude from the "if" direction of Theorem 3.4 that **VPV** proves $\mathbf{NP} \subseteq \mathbf{P/poly}$; i.e. **VPV** proves the $\mathbf{\Sigma}_2^B$ formula (4). Now from the proof of the "only if" direction we can obtain the desired polynomially bounded $\mathbf{EF}/O(1)$ system.

To add detail to the second part of the argument, we can rewrite the formula (4) in the form $\forall X \exists Y \forall Z \varphi(X, Y, Z)$, where φ is an open **VPV** formula, and we have replaced n by $|X|$, C by Y, and coded the pair (A, T) by Z. By applying the KPT Theorem to this we see that **VPV** proves a version of (6), and the propositional translation of this is a family of (n, ℓ) disjunctions (23), where $\ell = 2^{\lceil \log_2 k \rceil}$ and k is the constant in (6). Let F in Definition 6.7 be the polynomial time function which gives **EF** proofs of the (n, ℓ) disjunctions.

To obtain an $\mathbf{EF}/O(1)$ proof of a formula A, let $n = |\neg A|$, let $B_{n,i}$ be as in Definition 6.7, and let $B'_{n,i}$ be the result of substituting the n-tuple of bits coding $\neg A$ for p in (23) and suitable atoms r and constants for the other atoms of $B_{n,i}$ so that **EF** shortly proves

$$B'_{n,i} \to \neg \langle SAT(T_i, A_i) \rangle (r, a)$$

Now we use the fact mentioned as (22) in the beginning of this section to conclude **EF** shortly proves $B'_{n,i} \to A$.

To fix the slight glitch that we want $n = |A|$ rather than $n = |\neg A|$, simply replace A by $\neg A$ in the matrix of (4). □

Open Questions: Does Theorem 3.5 have a converse as in Theorem 3.4: If **VPV** proves **PH** = **BH** can we conclude **VPV** proves $\mathbf{NP} \subseteq \mathbf{P/poly}$?

Is there a converse to either of Theorems 3.6 or 5.2? For example, does $\mathbf{PH} = \mathbf{BH}$ imply $\mathbf{coNP} \subseteq \mathbf{NP}/O(1)$, possibly with the additional assumption that $\mathbf{NP} \subseteq \mathbf{P}/\mathbf{poly}$?

How strong are the finite advice extensions of classical proof systems like resolution or **EF** in terms of simulation?

References

[Bei91] Richard Beigel. Bounded Queries to SAT and the Boolean Hierarchy. *Theoretical Computer Science*, 84(2):199–223, 1991.

[BH91] Samuel Buss and Louise Hay. On Truth-Table Reducibility to SAT. *Information and Computation*, 91(1):86–102, 1991.

[BKT93] Samuel Buss, Jan Krajíček, and Gaisi Takeuti. On Provably Total Functions in Bounded Arithmetic Theories R^i_3, U^i_2, and V^i_2. In Peter Clote and Jan Krajíček, editors, *Arithmetic, Proof Theory and Computational Complexity*, pages 116–61. Oxford University Press, 1993.

[Bus86] Samuel Buss. *Bounded Arithmetic.* Bibliopolis, 1986.

[Bus90] Samuel Buss. Axiomatizations and conservation results for fragments of bounded arithmetic. In *Logic and Computation, Proceedings of a Workshop held at Carnegie Mellon University*, pages 57–84. AMS Contemporary Mathematics (106), 1990.

[Bus95] Samuel Buss. Relating the bounded arithmetic and polynomial time hierarchies. *Annals of Pure and Applied Logic*, 75:67–77, 1995.

[Bus98] Samuel Buss. First–Order Proof Theory of Arithmetic. In S. Buss, editor, *Handbook of Proof Theory*, pages 79–147. Elsevier, 1998. Available on line at `www.math.ucsd.edu/~sbuss/ResearchWeb/HandbookProofTheory/`.

[Cai01] Jin-Yi Cai. $\mathbf{S}^p_2 \subseteq \mathbf{ZPP}^{\mathbf{NP}}$. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS 2001)*, pages 620–628, 2001.

[CN06] Stephen Cook and Phuong Nguyen. Foundations of Proof Complexity: Bounded Arithmetic and Propositional Translations. unpublished manuscript `http://www.cs.toronto.edu/~sacook/`, 2006.

[Coo75] Stephen Cook. Feasibly constructive proofs and the propositional calculus. *Proceedings of the 7th Annual ACM Symposium on Theory of computing*, pages 83–97, 1975.

[Coo05] Stephen Cook. Theories for Complexity Classes and Their Propositional Translations. In Jan Krajíček, editor, *Complexity of computations and proofs*, pages 175–227. Quaderni di Matematica, 2005.

[CT06] Stephen Cook and Neil Thapen. The strength of replacement in weak arithmetic. *ACM Trans. on Computational Logic (TOCL)*, 7(4), 2006.

[KL80] R. M. Karp and R. J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proc. 12th Annual ACM Symposium on Theory of Computing*, pages 302–309, 1980.

[KL82] R. M. Karp and R. J. Lipton. Turing machines that take advice. *Enseignement Mathematique*, 30:255–273, 1982.

[KPT91] Jan Krajíček, Pavel Pudlák, and Gaisi Takeuti. Bounded arithmetic and the polynomial hierarchy. *Annals of Pure and Applied Logic*, 52:143–53, 1991.

[Kra93] Jan Krajíček. Fragments of bounded arithmetic and bounded query classes. *Trans. AMS*, 338(2):587–98, 1993.

[Kra95] Jan Krajíček. *Bounded Arithmetic, Propositional Logic and Computational Complexity*. Cambridge University Press, 1995.

[Pol99] Chris Pollett. Structure and Definability in General Bounded Arithmetic Theories. *Annals of Pure and Applied Logic*, 100:189–245, 1999.

[Zam96] D. Zambella. Notes on polynomially bounded arithmetic. *Journal of Symbolic Logic*, 61(3):942–966, 1996.