A Third-Order Bounded Arithmetic Theory for PSPACE

Alan Skelley*

Department of Computer Science, University of Toronto
10 King's College Road,
Toronto, ON M5S 3G4 Canada
skelley@acm.org

Abstract. We present a novel third-order theory W_1^1 of bounded arithmetic suitable for reasoning about PSPACE functions. This theory has the advantages of avoiding the smash function symbol and is otherwise much simpler than previous PSPACE theories. As an example we outline a proof in W_1^1 that from any configuration in the game of Hex, at least one player has a winning strategy. We then exhibit a translation of theorems of W_1^1 into families of propositional tautologies with polynomial-size proofs in BPLK (a recent propositional proof system for PSPACE and an alternative to G). This translation is clearer and more natural in several respects than the analogous ones for previous PSPACE theories.

Keywords: Bounded arithmetic, propositional proof complexity, PSPACE, quantified propositional calculus

1 Introduction

Theories of bounded arithmetic such as S_2^i and T_2^i of Buss [1] are interesting for their close ties to computational complexity. For example, the S_2 hierarchy collapses if and only if S_2 proves that the polynomial-time hierarchy collapses [3, 25, 16]. An important property of a theory is the computational complexity of functions that can be defined in it, and theories are known that correspond in this way to many natural complexity classes; see for example [7], [2], [13], [6].

Another important feature of theories of bounded arithmetic is that theorems can often be translated into families of tautologies with polynomial-sized proofs in a related propositional proof system. For example, propositional translations of theorems of Cook's equational theory of polynomial-time functions, PV, have polynomial-sized extended Frege proofs [12].

1.1 Our Results and Related Work

In his thesis [1], Buss introduced the first-order S_2 hierarchy but he also gave second-order theories U_2^1 and V_2^1 whose Σ_1^B-definable functions are exactly the

* Research supported by Canadian Natural Sciences and Engineering Research Council grant PGSB-208264-2000

classes PSPACE and EXPTIME, respectively. The ability to reason about the exponentially-large second-order objects gives the theory greatly increased power; for example, V_2^1 is otherwise identical to T_2, whose Σ_1^b-definable functions are from the polynomial hierarchy.

Now, Razborov [19] and Takeuti [24] independently showed a general method (the RSUV isomorphism) by which a first-order theory could be shown equivalent to a second-order theory: for example, the Σ_1^b-definable number functions of S_2^1 are the same as the Σ_1^B-definable *string* functions of V_1^1. Zambella [25] then gave a very elegant presentation of a second-order hierarchy $\{V^i\}$ equivalent to $\{S_2^i\}$. This second-order "viewpoint" has been adopted by other authors [8, 9] and has the advantages of greatly reducing the number of axioms required due to the absence of '#' (the smash function symbol) from the language and also simplifying the bootstrapping of the theories.

In this paper we introduce a new third-order theory called W_1^1 designed to exploit both the above uses of a higher order in bounded arithmetic: Firstly to simplify the language, presentation and bootstrapping and secondly to reason about exponentially large objects. We show that the $\Sigma_1^{\mathcal{B}}$-definable string functions of this theory are exactly those computable in polynomial space (PSPACE). Our witnessing theorem is much simpler than the analogous one for U_2^1 since it completely eliminates the complicated witnessing formulas of [1] and also uses a simpler comprehension scheme that does not necessitate adding comprehension rules to the sequent calculus.

We also discuss a recent propositional proof system, BPLK [22], corresponding to PSPACE and give a translation of theorems of W_1^1 into families of propositional tautologies with polynomial-size proofs in this new system. This translation is very much simpler than the analogous one for U_2^1 and G, the quantified propositional calculus that is the only previously studied propositional proof system for PSPACE. This latter translation is from [17] and lacks many technical details that we suspect would be very tricky if written out in full.

2 A Third-Order Language

We consider a three-sorted ("third-order") predicate calculus with free and bound variables of the first sort named $a, b, c, \ldots$ and $x, y, z, \ldots$, respectively, and free and bound variables of the second sort named $A, B, C, \ldots$ and $X, Y, Z, \ldots$, and likewise of the third sort named $\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$ and $\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \ldots$. The first sort is intended to represent natural numbers; the second, finite sets of natural numbers; and the third, finite sets of finite sets. The language $\mathcal{L}_A^3$consists of the following set of non-logical symbols: $\mathcal{L}_A^3 = \{0, 1, +, \cdot, |\cdot|_2, \in_2, \in_3, \leq, =\}$, the same as the set $\mathcal{L}_A^2$for V^1 but with the addition of the third-order membership predicate $A \in_3 \mathcal{B}$. Note in particular the absence of the smash function symbol. The expression $|X|_2$ is intended to represent the largest element of the set X. Such sets are interchangeable with finite binary strings under the following mapping, as in [9]: The set X represents the string with length $|X|_2 - 1$ whose i^{th} bit is 1 exactly when $i \in_2 X$. This map is a bijection with the exception that the string corresponding to the empty set would be undefined, so we define it to be the empty string. Third-order objects can then be thought of as sets of strings.

Number terms are defined identically as in V^1, in particular not including any reference to third-order variables. Formulas additionally may have third-order variables and quantifiers. The hierarchy $\Sigma_i^{\mathcal{B}}$ of classes of formulas in this language is analogous to Σ_i^B and Σ_i^b for second- and first-order formulas: $\Sigma_i^{\mathcal{B}}$ consists of those formulas with arbitrarily many bounded first- and second-order quantifiers, and exactly i alternations of third-order quantifiers, the outer-most being **restricted**, i.e. equivalent to an existential quantifier. We shall be concerned only with $i \in \{0,1\}$. Now, **strict** $\Sigma_1^{\mathcal{B}}$-formulas are those consisting of a single existential third-order quantifier followed by a formula with no third-order quantifiers; we shall be mainly concerned with a slightly more inclusive class of formulas called $\forall^2\Sigma_1^{\mathcal{B}}$, consisting of a single bounded universal second-order quantifier followed by a **strict** $\Sigma_1^{\mathcal{B}}$-formula. Restricting several schemes in our theory to this class will be justified in section 4 by the fact that an appropriate replacement scheme will be provable in our theory.

Note that third-order quantifiers are not bounded, and in fact there does not seem to be any way to bound them since terms cannot reference third-order variables. Fortunately, in the appropriate fragment of the theory we shall be concerned with, these variables will always be implicitly bounded.

3 The Theory W_1^1

W_1^1 is a theory over the above-defined third-order language. The axioms of W_1^1 are B1-B12 and L1,L2 of [8] (open axioms defining the function and predicate symbols in the language), (strict) $\forall^2\Sigma_1^{\mathcal{B}}$-IND, and the following two comprehension schemes $\Sigma_0^{\mathcal{B}}$-2COMP:

$$(\exists Y \leq t(\overline{x},\overline{X}))(\forall z \leq s(\overline{x},\overline{X}))[\phi(\overline{x},\overline{X},\overline{\mathcal{X}},z) \leftrightarrow Y(z)]$$

and $\Sigma_0^{\mathcal{B}}$-3COMP:

$$(\exists \mathcal{Y})(\forall Z \leq s(\overline{x},\overline{X}))[\phi(\overline{x},\overline{X},\overline{\mathcal{X}},Z) \leftrightarrow \mathcal{Y}(Z)],$$

where in each case $\phi \in \Sigma_0^{\mathcal{B}}$ subject to the restriction that neither Y nor $\mathcal{Y}$, as appropriate, occurs free in ϕ. $\mathcal{Y}(Z)$ abbreviates $X \in_3 \mathcal{Y}$, and similarly for $Y(z)$.

4 $\Sigma_1^{\mathcal{B}}$-Replacement Schemes

In this section we shall show that W_1^1 proves various replacement schemes, allowing third-order existential quantifiers to be moved past lower-order quantifiers.

First, though, it is convenient to note that adding to W_1^1 function symbols for its number- and string-valued $\Sigma_1^{\mathcal{B}}$-definable functions results in a conservative extension. The proof of the present claim is analogous to that for first-order bounded arithmetic theories in section 2.3 of [1]. In that proof, a given Σ_1^b-formula in the augmented language is shown to be equivalent to a constructed

Σ_1^b-formula in the original language, and preserves strictness of the quantifier syntax.

$W_1^1 \supset V (= \bigcup V^i)$ since all the axioms of the latter theory are in the former. W_1^1 can therefore $\Sigma_\infty^{\mathcal{B}}$-define all number- and string-valued functions of number and string arguments from the polynomial-time hierarchy. By the remarks in the previous paragraph, we can add symbols for these functions to W_1^1 and obtain a conservative extension. In particular, pairing functions such as $\langle x, y\rangle$, $\langle X, Y\rangle$ and $\langle X, y\rangle$ may be added. For a third-order variable $\mathcal{X}$ define $\mathcal{X}^{[x]}(X) \equiv \mathcal{X}(\langle x, X\rangle)$ and $\mathcal{X}^{[X]}(Y) \equiv \mathcal{X}(\langle X, Y\rangle)$, which make $\mathcal{X}$ into an array, with rows indexed by number or strings respectively, each row of which is a third-order object. Let '$\frown$' represent string concatenation and $\dot{-}$ represent limited subtraction. With this in mind, we can state the $\Sigma_1^{\mathcal{B}}$ replacement schemes:

Definition 1 ($\Sigma_1^{\mathcal{B}}$ Replacement Schemes). *$\Sigma_1^{\mathcal{B}}$-1REPL is:*

$$\forall x \leq y \exists \mathcal{X} \phi(x, y, \mathcal{X}) \leftrightarrow \exists \mathcal{X} \forall x \leq y \phi(x, y, \mathcal{X}^{[x]})$$

and $\Sigma_1^{\mathcal{B}}$-2REPL is:

$$\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X}) \leftrightarrow \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]}),$$

where in each case ϕ is a (general) $\Sigma_1^{\mathcal{B}}$-formula that may have other free variables than those indicated.

Theorem 2. *The $\Sigma_1^{\mathcal{B}}$ replacement schemes are theorems of W_1^1.*

Proof (Proof Sketch). Although the $\Sigma_1^{\mathcal{B}}$-1REPL scheme has a simpler proof, it can also be proved in the same way as the $\Sigma_1^{\mathcal{B}}$-2REPL scheme, so we sketch only a proof of the latter.

$\leftarrow$: This direction of the equivalence, namely that for $\phi(X, y, \mathcal{X}) \in \Sigma_1^{\mathcal{B}}$

$$W_1^1 \vdash \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]}) \supset \forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X})$$

is immediate.

$\rightarrow$: The existence of a proof in W_1^1 of this direction of the equivalence is itself proved by structural induction on ϕ. The base case of the induction is when ϕ is $\Sigma_0^{\mathcal{B}}$. let ψ be $\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X})$. Let $\theta(c)$ be the formula

$$\forall X \leq (y \dot{-} c) \exists \mathcal{X} \forall Y \leq c \phi(X \frown Y, y, \mathcal{X}^{[Y]}).$$

$\theta(0)$ is a simple logical consequence of ψ, and $W_1^1 \vdash \psi \wedge \theta(c) \supset \theta(c+1)$ by use of $\Sigma_0^{\mathcal{B}}$-3COMP to combine two third-order objects (coding the two arrays of third-order objects for all strings of length smaller than y starting with $X \frown 0$ and $X \frown 1$ respectively) into one third-order object coding the array for all strings of length smaller than y starting with X. Thus $W_1^1 \vdash \psi \supset \theta(y)$ by $\forall^2 \Sigma_1^{\mathcal{B}}$-IND, and clearly $W_1^1 \vdash \theta(y) \supset \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]})$. This induction, incidentally, is the only place where $\forall^2 \Sigma_1^{\mathcal{B}}$-IND, rather than strict $\Sigma_1^{\mathcal{B}}$-IND, seems to be necessary.

The induction step ($\phi \notin \Sigma_0^{\mathcal{B}}$) is proved by putting the formula in prenex form and then applying the induction hypothesis several times to manipulate the quantifiers. We omit the details. $\square$

The following is an immediate, useful corollary:

Corollary 3. *Let $\phi \in \Sigma_1^{\mathcal{B}}$. Then there exists $\psi \in$ strict $\Sigma_1^{\mathcal{B}}$ such that $W_1^1 \vdash \phi \leftrightarrow \psi$.*

5 Definability in W_1^1

We know that W_1^1 can $\Sigma_0^{\mathcal{B}}$-define all functions (of string variables) from the polynomial-time hierarchy. In fact, W_1^1 can $\Sigma_1^{\mathcal{B}}$-define all string functions computable in polynomial space:

Theorem 4. *Let $f \in PSPACE$ be of polynomial growth rate. Then there is a strict $\Sigma_1^{\mathcal{B}}$-formula ϕ such that*

1. $W_1^1 \vdash \forall X \exists Y \phi(X, Y)$
2. $W_1^1 \vdash \forall X \forall Y \forall Z(\phi(X, Y) \wedge \phi(X, Z) \supset Y =_2 Z$ *($Y =_2 Z$ may be defined as* $(|Y|_2 = |Z|_2 \wedge \forall x \leq |Y|_2(Y(x) \leftrightarrow Z(x)))))$
3. *For all strings X, $\phi(X, f(X))$ is true.*

Proof (Proof Sketch). The proof is by induction on the logarithm of the length (number of steps) of the PSPACE computation that for any initial configuration there is a unique ending configuration. In the induction step two computations of length 2^i are pieced together using $\Sigma_0^{\mathcal{B}}$-3COMP. □

5.1 Strategies in Hex

As an example, consider the game of Hex, which has recently achieved some notoriety in the form of propositional tautologies due to Buss [5], Urquhart and others. These tautologies state that a finished game of Hex has a winner and are generally provable in Frege, resolution or weaker systems, depending on the formulation. The winner can be found in logarithmic space by solving a related graph reachability problem. A related problem is to determine which player has the winning strategy from a given configuration, which is PSPACE complete [20]. A Hex configuration is easily coded as a string, compared to which a strategy is an exponential-sized object coding a map from partially filled boards to moves. Thus there is a $\Sigma_1^{\mathcal{B}}$ formula Strategy1(X) stating that there exists a strategy such that for any (game sized) sequence of moves by player 2, when player 1 responds according to his strategy then he is the winner. There is similarly a $\Sigma_1^{\mathcal{B}}$-formula Strategy2(X), and as expected,

Theorem 5.
$$W_1^1 \vdash \forall X[\mathit{Strategy1}(X) \vee \mathit{Strategy2}(X)].$$

Proof (Proof Sketch). Given a configuration X, we can define **continuations** of X as those configurations reachable from X by play. This is a simple matter of counting the numbers of added pieces of each colour, and checking that no existing pieces have been changed or removed. Then it is proved by induction on the number of remaining moves in the game that from any continuation of X, some player has a winning strategy. The base case is a reformulation of the above tautologies and is thus easily provable in W_1^1. In the induction step, from a given position Y the induction hypothesis gives a winning strategy for some player from each possible next position. If the current player can reach a winning position with a move then the strategy for the current position is amended to apply to the configuration Y by adding that move. Otherwise, a strategy for the other player is the merger of his strategies for all possible next positions. □

5.2 A Witnessing Theorem for W_1^1

To prove the converse of Theorem 4, namely that functions provably total in W_1^1 are in PSPACE, we shall use a Buss-style witnessing argument, which requires that we define an equivalent sequent calculus formulation $\mathrm{LK}^3 - W_1^1$ of W_1^1. We omit this for brevity, but it is essentially LK with the addition of second- and third-order quantifier introduction rules (replacing only free variables by quantifiers) plus the following $\forall^2\Sigma_1^{\mathcal{B}}$-IND rule:

$$\frac{\Gamma, \phi(b) \longrightarrow \phi(b+1), \Delta}{\Gamma, \phi(0) \longrightarrow \phi(t), \Delta},$$

where b appears only as indicated and $\phi \in \forall^2\Sigma_1^{\mathcal{B}}$. As initial sequents we allow all substitution instances of the axioms (other than induction) of W_1^1. Note that all rules of $\mathrm{LK}^3 - W_1^1$ are valid in W_1^1, and furthermore, $\mathrm{LK}^3 - W_1^1$ proves the induction and comprehension schemes of W_1^1. Formally, $\mathrm{LK}^3 - W_1^1$ also adopts the usual conventions concerning free and bound variables, as in [4].

The standard definition of an anchored cut in LK^3 is extended for $\mathrm{LK}^3 - W_1^1$ by allowing cuts on the descendents of principal formulas of the $\forall^2\Sigma_1^{\mathcal{B}}$-IND rule, in addition to cuts on descendents of formulas in non-logical axioms. The anchored completeness theorem for LK^3 can then be extended to $\mathrm{LK}^3 - W_1^1$ in the usual way to cope with the induction rules, as detailed in [23].

With this in mind, we can now state the witnessing theorem we wish to prove, followed by several definitions:

Theorem 6. *Suppose $W_1^1 \vdash \exists Y \phi(X,Y)$, for $\phi(X,Y) \in \Sigma_1^{\mathcal{B}}$ with all free variables displayed. Then there exists a function $f \in PSPACE$ of polynomial growth rate such that for every string X, $\phi(X, f(X))$ is true.*

Definition 7. *Let $\psi \equiv \forall X \le t \exists \mathcal{X} \phi(X, \mathcal{X}) \in \forall^2\Sigma_1^{\mathcal{B}}$, with other free variables not shown. Consider an assignment to the free variables of ψ. Then the string relation $\mathcal{A}(A,B)$* **satisfies** *ψ (with respect to the assignment to the free variables of ψ) iff for every string A of no more than t bits, $\phi(A, \{B\}(\mathcal{A}(A,B)))$ is true in the standard model, where $\{B\}(\mathcal{A}(A,B))$ denotes the unary string predicate (with argument B) obtained by fixing the first argument of relation $\mathcal{A}$ to A.*

Definition 8. *Let S be the sequent $\Gamma \longrightarrow \Delta$ such that $\Gamma \bigcup \Delta \subset \forall^2\Sigma_1^{\mathcal{B}}$, i.e.*

$$\Gamma = \{\forall A_i \le s_i \exists \mathcal{A}_i \gamma_i(A_i, \mathcal{A}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})\} \quad \text{and}$$
$$\Delta = \{\forall C_i \le t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})\},$$

with $\{\gamma_i\} \bigcup \{\delta_i\} \subset \Sigma_0^{\mathcal{B}}$*. (Leading quantifiers are written for simplicity but may be absent.)*

Then **PSPACE Oracle Witnessing Operators (POWOs)** *for* S *are operators, or type-2 predicates. For each formula from* Δ

$$\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})$$

that is not $\Sigma_0^{\mathcal{B}}$ *(and may or may not have the leading string quantifier as pictured), the POWO* f_i *is a predicate with arguments* $\{\overline{\mathcal{B}}, \overline{B}, \overline{b}\}$ *(for the free variables of the sequent),* $\{\mathcal{A}_j(A_j, X)\}$ *(for the string relations satisfying the formulas in the antecedent) and finally* $\{C_i, X\}$*, making* f_i *into a two-place string relation when the other arguments are fixed. The* f_i *must have the property that for any assignment to the free variables* $\overline{\mathcal{B}}, \overline{B}, \overline{b}$ *of* S *and string relations* $\{\mathcal{A}_j(A_j, X)\}$*, if each formula* γ_j *on the left is satisfied by the corresponding* $\mathcal{A}_j$*, then some* δ_i *on the right is satisfied by the string relation* $\{C_i, X\} f_i$*, obtained by fixing all but the last two arguments to the operator* f_i*.*

Furthermore, each f_i *is computable by an oracle Turing machine in space (including on the query tapes) polynomial in the lengths of its string and number inputs.*

Now the theorem will follow from the following lemma:

Lemma 9. *Suppose* $LK^3 - W_1^1 \vdash \Gamma \longrightarrow \Delta$*, where* $\Gamma \bigcup \Delta \subset \forall^2 \Sigma_1^{\mathcal{B}}$*. Then there exist PSPACE Oracle Witnessing Operators for* $\Gamma \longrightarrow \Delta$*.*

Proof (Proof of Theorem 6 from Lemma 9). Suppose $W_1^1 \vdash \exists Y \phi(X, Y)$, for $\phi(X, Y) \in \Sigma_1^{\mathcal{B}}$ with all free variables displayed. By Parikh's theorem, $W_1^1 \vdash \exists Y \leq t(|X|_2)\phi(X, Y)$, for some term t. By Corollary 3, $W_1^1 \vdash \phi(X, Y) \leftrightarrow \exists \mathcal{Y} \psi(X, Y, \mathcal{Y})$, for some $\psi \in \Sigma_0^{\mathcal{B}}$. Also, $W_1^1 \vdash \exists Y \leq t(|X|_2) \exists \mathcal{Y} \psi(X, Y, \mathcal{Y}) \leftrightarrow \exists \mathcal{Y} \exists Y \leq t(|X|_q) \psi(X, Y, \mathcal{Y})$. Applying the lemma to the sequent $\longrightarrow \exists \mathcal{Y} \exists Y \leq t(|X|_2) \psi(X, Y, \mathcal{Y})$, we obtain a PSPACE (in $|X|$) predicate for $\mathcal{Y}$ satisfying that sequent, and so for particular X the string Y can be obtained in PSPACE by evaluating ψ, with access to the predicate $\mathcal{Y}$, on each string of length $\leq t(|X|_2)$ in turn. It is easy to see that the computed string Y satisfies $\phi(X, Y)$ (for the same fixed X). □

All that remains is to prove the lemma:

Proof (Proof of Lemma 9). Suppose $LK^3 - W_1^1 \vdash \Gamma \longrightarrow \Delta$, where $\Gamma \bigcup \Delta \subset \forall^2 \Sigma_1^{\mathcal{B}}$, and consider an anchored proof π of this sequent. Since both the endsequent of π and every non-logical axiom of $LK^3 - W_1^1$ is $\forall^2 \Sigma_1^{\mathcal{B}}$, and since the induction rule is limited to this same class of formulas, every formula in π is $\forall^2 \Sigma_1^{\mathcal{B}}$.

We now show by induction on the number of sequents in π that POWOs exist for $\Gamma \longrightarrow \Delta$.

Base Case: The base case is that $\Gamma \longrightarrow \Delta$ is either an initial sequent of LK^3 or an instance of an axiom. The only such sequents requiring POWOs are those with a third-order quantifier in the succedent, namely an instance

$$\longrightarrow (\exists \mathcal{Y})(\forall Z \leq s(\overline{B}, \overline{b}))[\phi(\overline{\mathcal{B}}, \overline{B}, \overline{b}, Z) \leftrightarrow \mathcal{Y}(Z)]$$

of $\Sigma_0^{\mathcal{B}}$-3COMP, where $\phi \in \Sigma_0^{\mathcal{B}}$, subject to the restriction that $\mathcal{Y}$ does not occur free in ϕ. The only POWO required for this sequent is computed by the predicate

$$f(\overline{\mathcal{B}}, \overline{B}, \overline{b}, \overline{\mathcal{A}}, Z) \leftrightarrow |Z|_2 \leq s(\overline{B}, \overline{b}) \wedge \phi(\overline{\mathcal{B}}, \overline{B}, \overline{b}, Z),$$

which is in some level of the polynomial-time hierarchy, and thus certainly in PSPACE.

Induction Step: The induction step has several cases depending on which rule has been used to derive $\Gamma \longrightarrow \Delta$.

1.-8. Weakening; Contraction; Exchange, introduction of $\neg$, $\vee$ on the right and $\wedge$ on the left; Introduction of $\vee$ on the left and $\wedge$ on the right; First- or second-order $\forall$: **left** and $\exists$: **right**; First- or second-order $\forall$: **right** and $\exists$: **left**; Third-order $\exists$: **left**; and Third-order $\exists$: **right**: These cases are all easy and are omitted for brevity.

9. The **cut** rule:
The inference is

$$\frac{\Gamma \longrightarrow \phi, \Delta \qquad \Gamma, \phi \longrightarrow \Delta}{\Gamma \longrightarrow \Delta}.$$

A POWO for the conclusion proceeds in two phases: First, it evaluates its formula using the POWO from the left hypothesis, and if that POWO satisfies the formula, it emulates it. Otherwise, it emulates the POWO from the right hypothesis, and uses the POWO for ϕ from the left hypothesis to supply a value for the oracle argument. The whole procedure uses at most the sum of the space requirements of the two POWOs from the hypotheses. If any free variables are eliminated, then as before a dummy argument of the correct type is supplied to the POWOs.

10. $\forall^2 \Sigma_1^{\mathcal{B}}$-IND:
The inference is:

$$\frac{\Gamma, \phi(b) \longrightarrow \phi(b+1), \Delta}{\Gamma, \phi(0) \longrightarrow \phi(t), \Delta}.$$

The POWOs for the conclusion will iterate the construction from the previous case, as the current instance of the induction rule could be simulated by t instances of the cut rule, along with some weakenings.
More precisely, let f_ϕ be the POWO for the instance of ϕ in the succedent of the hypothesis. Let ψ be any formula in the succedent of the hypothesis (including ϕ) and f_ψ its POWO. We construct a POWO f'_ψ for ψ in the conclusion in stages:
$f_\psi^0(X, Y) \leftrightarrow f_\psi(X, Y)$.
$f_\psi^k(X, Y) \leftrightarrow (\psi(f_\psi^{k-1}) \wedge f_\psi^{k-1}(X, Y)) \vee (\neg\psi(f_\psi^{k-1}) \wedge f_\psi^{k-1}(f_\phi, X, Y))$.
f_ψ^1 checks if f_ψ satisfies ψ and if so, simulates f_ψ. If not, f_ψ^1 computes $f_\psi(f_\phi)$, that is to say, uses f_ϕ to answer queries to the oracle argument corresponding to ϕ.
f_ψ^k checks if f_ψ^{k-1} satisfies ψ and if so, simulates f_ψ^{k-1}. If not, f_ψ^k computes $f_\psi^{k-1}(f_\phi)$.

f'_ψ, then, evaluates t and computes f_ψ^t. Computing f_ψ^t requires t times the space required to compute f_ϕ plus the space requirements of f_ψ, and so only increases the space usage of POWOs by a polynomial factor.

□

6 The propositional system BPLK

In this section we review the sequent system BPLK [22], which is basically PK (i.e., the propositional fragment of LK) enhanced with the reasoning power of Boolean programs, defined below. These (Boolean programs) were introduced in [10] and are a way of specifying Boolean functions. They are something like a generalization of the technique of using new atoms to replace part of a Boolean formula, which idea is the basis of extended Frege systems. The following definition is from that paper:

Definition 10 (Cook-Soltys). *A* Boolean Program *P is specified by a finite sequence $\{f_1, ..., f_m\}$ of function symbols, where each symbol f_i has an associated arity k_i, and an associated defining equation*

$$f_i(\overline{p_i}) := A_i$$

where $\overline{p_i}$ is a list $p_1, ..., p_{k_i}$ of variables and A_i is a formula all of whose variables are among $\overline{p_i}$ and all of whose function symbols are among $f_1, ..., f_{i-1}$. In this context the definition of a formula *is expanded to allow the use of function symbols as connectives.*

The semantics are as for propositional formulas, except that when evaluating an application $f_i(\overline{\phi})$ of a function symbol, the value is defined, using the defining equation, to be $A_i(\overline{\phi})$. There is no free/bound distinction between variables in the language of Boolean programs.

An interesting property of Boolean programs from [10] that demonstrates their comparability to quantified Boolean formulas is that evaluating them is PSPACE-complete.

Definition 11 (BPLK). *The system BPLK is like the propositional system PK, but with the following changes:*

1. *In addition to sequents, a proof also includes a Boolean program that defines functions. Whenever we refer to a BPLK-proof, we shall always explicitly write it as the pair $\langle \pi, P \rangle$ of the proof (sequents) and the Boolean program defining the function symbols occurring in the sequents.*
2. *Formulas in sequents are formulas in the context of Boolean programs, as defined earlier.*
3. *If the Boolean program contains a definition of the form $f(\overline{p}) := A(\overline{p})$, the new LK rules*

$$f : \textbf{left} \quad \frac{A(\overline{\phi}), \Gamma \longrightarrow \Delta}{f(\overline{\phi}), \Gamma \longrightarrow \Delta} \qquad \textit{and} \qquad f : \textbf{right} \quad \frac{\Gamma \longrightarrow \Delta, A(\overline{\phi})}{\Gamma \longrightarrow \Delta, f(\overline{\phi})}$$

may be used, where $\overline{\phi}$ are precisely as many formulas as $\overline{p}$ are variables.

4. (**Substitution Rule**) *The new inference rule* **subst**

$$\frac{\Delta(q,\overline{p}) \longrightarrow \Gamma(q,\overline{p})}{\Delta(\phi,\overline{p}) \longrightarrow \Gamma(\phi,\overline{p})}$$

may be used, where all occurrences of q have been substituted for.

Simultaneous substitutions can be simulated with several applications of **subst**. The following is the main result of [22]:

Theorem 12. *BPLK and G are polynomially equivalent for proofs of propositional tautologies.*

7 Translation into BPLK

In this section we define a translation $||\cdot||$ of Σ_∞^B formulas in the language $\mathcal{L}_A^3$ of W_1^1 (i.e. with no third-order quantifiers) into families of propositional sequents in the language of Boolean programs. Our main result is

Theorem 13. *If $\phi(A) \in \Sigma_\infty^B$ and if $W_1^1 \vdash \phi(A)$ then BPLK has polynomial-sized proofs of the translations $||\phi||$; furthermore, these proofs are definable in S_2^1 and V^1 (or any theory defining polytime functions).*

This will follow directly from lemma 17 below. The definability of the proofs follows from the fact that they can actually be constructed in polynomial time.

First, we can extend the definitions of a Boolean Program and of a BPLK proof as follows:

Definition 14. *A* **Boolean semiprogram** *is like a Boolean program, except we allow some function symbols used in the program to be undefined ("free").*

Definition 15. *A* **BPLK-sequence** *is the same as a BPLK proof except that the requirement that all function symbols occurring in the sequence be defined by the accompanying Boolean program is dropped. Furthermore, the accompanying Boolean program is instead a Boolean semiprogram. Any undefined function symbol appearing in the sequence or the semiprogram is called "free".*

The following translation is defined for the larger class $\Sigma_0^{\mathcal{B}}$ (including free third-order variables) and is necessary for the main lemma in the proof:

Definition 16. *Let $\phi(\mathcal{A}_1,...,\mathcal{A}_j,A_1,...,A_k)$ be $\Sigma_0^{\mathcal{B}}$ in the language $\mathcal{L}_A^3$. For every $m_1,...,m_k$ (lengths of the string objects) we construct a Boolean semiprogram $P_\phi^{m_1,...,m_k}$ and a formula $||\phi||^{m_1,...,m_k}$ in the language of Boolean programs, with the atoms $\overline{\overline{p}} = (\overline{p_i}, i = 1,...,k)$, where each $\overline{p_i} = (p_{i,0},...,p_{i,m_k})$. By induction on the structure of ϕ:*

– *If ϕ is an atomic formula $s = t$, $t \leq s$ or $t \in_2 T_i$ then s and t are first-order terms with no free first-order variables and refer only to the length of strings, which is known. They can be evaluated and $||\phi||^{m_1,...,m_k}$ is a constant.*

- *The cases where ϕ is formed with a propositional connective are trivial and we omit the details.*
- *If ϕ is the atomic formula $A_i \in_3 \mathcal{A}_j$ then $||\phi||^{m_i} := g_{\mathcal{A}_j}(p_{i,0}, ..., p_{i,m_i})$. $P_\phi^{m_i} := \emptyset$. The intention is that $g_{\mathcal{A}_j}$ be a free function symbol and we shall be careful not to add a definition for any function symbol of this form to our Boolean semiprograms. Furthermore, this is the only case in the construction where a free function symbol is produced.*
- *If ϕ is $\exists x \leq t\psi(x)$ then $||\phi||^{m_1,...,m_k} := \bigvee_{n\leq \underline{t}} ||\psi(n)||^{m_1,...,m_k}$ ($\phi(n)$ is $\phi(x)[s/x]$ where s is a constant term of value n, say $\overbrace{1 + ... + 1}^{n}$). $P_\phi^{m_1,...,m_k} := P_\psi^{m_1,...,m_k}$.*
- *If ϕ is $\forall x \leq t\psi(x)$ then $||\phi||^{m_1,...,m_k} := \bigwedge_{n\leq \underline{t}} ||\psi(n)||^{m_1,...,m_k}$. $P_\phi^{m_1,...,m_k} := P_\psi^{m_1,...,m_k}$.*
- *If ϕ is $\exists X \leq t\psi(X)$ then $||\phi||^{m_1,...,m_k} := f_\phi(\overline{\overline{p}})$ and $P_\phi^{m_1,...,m_k}$ is as follows:*

$$f^l_{\phi,0}(\overline{\overline{p}}, q_0, ..., q_l) := ||\psi||^{m_1,...,m_k,l}$$

for each $l \leq \underline{t}$.

$$f^l_{\phi,i}(\overline{\overline{p}}, q_i, ..., q_l) := f^l_{\phi,i-1}(\overline{\overline{p}}, 0, q_i, ..., q_l) \vee f^l_{\phi,i-1}(\overline{\overline{p}}, 1, q_i.,,, .q_l)$$

for each $l \leq \underline{t}$ and $i \leq l + 1$.

$$f_\phi(\overline{\overline{p}}) := \bigvee_{l\leq \underline{t}} f^l_{\phi,l+1}(\overline{\overline{p}}).$$

- *The case where ϕ is $\forall X \leq t\psi(X)$ is symmetric to the previous one.*

It is clear that for fixed ϕ, the size of $||\phi||^{m_1,...,m_k}$ is polynomial in $m_1.,,, .m_k$. Whenever we talk of BPLK proofs or BPLK-sequences involving translations of this form, we shall insist that the associated Boolean (semi-)program extend the (semi-)program resulting from the translation.

The following lemma is the main lemma of the proof. In the previous section, since it is not possible to translate a general $\Sigma_1^{\mathcal{B}}$ formula into the language of BPLK, we defined POWOs and used them to witness a sequent containing third-order quantifiers. Similarly, in the lemma below we shall translate sequents with third-order quantifiers as if those third-order variables were free, and then show that BPLK can prove the existence of a function symbol witnessing the sequent in much the same way. This aspect of the statement of the lemma is greatly simplified compared to the analogous lemma in [17], where to talk about an arbitrary witness to the antecedent of the sequent, the authors stated the lemma with arbitrary formulas of the appropriate class substituted for the third-order variables.

Since formulas in the proof are not all guaranteed to be **strict** $\Sigma_1^{\mathcal{B}}$, due to the slightly more complicated induction scheme in W_1^1, the translations used in the lemma are actually translations of the equivalent form given by the replacement theorem (i.e. with the third-order quantifier moved to the front).

Lemma 17. *Let $LK^3 - W_1^1 \vdash \Gamma \longrightarrow \Delta$ where $\Gamma \bigcup \Delta \subset \forall^2 \Sigma_1^{\mathcal{B}}$, i.e.*

$$\Gamma = \{\forall A_i \leq s_i \exists \mathcal{A}_i \gamma_i(A_i, \mathcal{A}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})\} \qquad \text{and} \qquad \Delta = \{\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})\},$$

with $\{\gamma_i\} \bigcup \{\delta_i\} \subset \Sigma_0^{\mathcal{B}}$, and although we write for simplicity the initial string and third-order quantifiers for each formula, in fact for some of the formulas either the initial string quantifier or both initial quantifiers may be absent.

Then for each $m_1, ..., m_k$ and $n_1, ..., n_l$ there are function symbols $h_i^{\overline{m},\overline{n}}$ and BPLK-sequences with endsequents

$$..., ||\forall A_i \gamma_i(A_i, \mathcal{A}_i^{[A_i]}, \overline{\mathcal{B}}, \overline{B}, \underline{\overline{n}})||^{m_1,...,m_k}, ... \longrightarrow ..., ||\forall C_i \delta_i(C_i, \mathcal{C}_i^{[C_i]}, \overline{\mathcal{B}}, \overline{B}, \underline{\overline{n}})||^{m_1,...,m_k}[h_i^{\overline{m},\overline{n}}/g_{\mathcal{C}_i}], ...$$

where $h_i^{\overline{m},\overline{n}}$ are called witnessing function symbols and are not free, but may be defined in terms of free function symbols (in particular, $g_{\mathcal{A}_i}$). Furthermore, these sequences have size polynomial in $m_1, ..., m_k$ and $n_1, ..., n_l$.

The notation $...[h_i^{\overline{m},\overline{n}}/g_{\mathcal{C}_i}]$ in the succedent means that one should first perform the translation, and then substitute function symbol h_i for the free symbol $g_{\mathcal{C}_i}$ in the result.

Proof. We show the existence of the desired BPLK-sequence by induction on the number of sequents in the W_1^1 proof, in a manner very similar to the witnessing theorem of the previous section. The witnessing function symbols of the present lemma are analogous to POWOs.

Base Case: This is trivial for initial sequents and the witnessing function symbol, if required, is defined to be the constant false predicate. For translations of axioms B1-B12, L1, L2 and instances of $\Sigma_0^{\mathcal{B}}$-2COMP, it follows from the analogous result for V_1^1 and Extended Frege. For translations of instances of $\Sigma_0^{\mathcal{B}}$-3COMP, the witnessing function symbol has defining formula identical to the comprehension formula, and then the translation of the instance is proved using the introduction rule for this symbol followed by repeated substitutions and $\wedge$: **right** inferences.

Induction Step: There are cases depending on the final inference of the W_1^1 proof:

1.-5. Weakening, Exchange, introduction of $\neg$, $\vee$ on the right and $\wedge$ on the left; Contraction, introduction of $\vee$ on the left and $\wedge$ on the right; introduction of first-, second- and third- order quantifiers:
These cases are all straightforward and are omitted.

6. Cut, Induction:
The cut rule is handled by defining new witnessing function symbols for the conclusion by cases, using the witnessing function symbol for the cut formula. For induction this procedure is iterated as many times as the value of the induction bound.

For example, if the cut formula is $\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i)$, then a new witnessing function symbol h_j for $\forall C_j \leq t_j \exists \mathcal{C}_j \delta_j(C_j, \mathcal{C}_j)$ would be defined as follows, where h'_j is the witnessing function symbol for the hypothesis with the cut formula on the right, and h''_j that for the hypothesis with the cut formula on the left:

$$h_j := (||\delta_j(C_j, \mathcal{C}_j^{[C_j]})||[h'_j/g_{\mathcal{C}_j}] \wedge h_j) \vee h''_j(h_i).$$

□

7.1 Consistency and Polynomial Simulation

Now Cook [12] and later others [15], [17], [14], etc. showed that some bounded arithmetic theories can prove the consistency of related propositional proof systems, and furthermore that any proof system whose consistency can be proved in the theory can be polynomially simulated by the related proof system. For completeness we mention the analogous results for W_1^1 and BPLK.

Let BPTAUT(X) be a formula stating that the string X codes a tautological propositional formula in the language of Boolean programs, as follows: "for any assignment to the free variables, there exists a transcript of the exponential-length computation of the Boolean function symbol terms occurring in the formula such that the resulting truth-values satisfy the formula". Clearly a $\Sigma_1^{\mathcal{B}}$ formula will suffice. Let $\mathrm{Prf}_{\mathrm{BPLK}}(X, Y)$ be a $\Sigma_0^{\mathcal{B}}$ formula stating that X codes a BPLK-proof of the formula coded by Y. Then

Theorem 18.

$$W_1^1 \vdash \forall X, Y[Prf_{BPLK}(X, Y) \supset BPTAUT(Y)]$$

The formula in the theorem is called RFN(BPLK).

Proof (Proof Sketch). By induction on the length of the proof, similar to the witnessing theorem, a transcript is constructed, for each assignment, of evaluating the formula at that assignment. □

The next thing to show would be that if P is a proof system whose consistency can be proved in W_1^1, i.e. $W_1^1 \vdash \mathrm{RFN}(P)$, then BPLK polynomially simulates P. For U_2^1, what is known is actually the weaker statement that if U_2^1 proves $i-\mathrm{RFN}(P)$, which is the consistency of P for Σ_i^q formulas, then G polynomially simulates P for proofs of those formulas. An analogous statement is almost certainly true of W_1^1 and BPLK simply because BPLK polynomially simulates G, and because W_1^1 and U_2^1 are most likely related by an RSUV-style isomorphism. Of more interest is the statement for RFN(P), but this formula is likely not $\Sigma_0^{\mathcal{B}}$ for interesting proof systems (G or even BPLK, for instance), and so the usual techniques do not seem to apply due to the expressibility of formulas in the language of BPLK. A proof system with more expressive formulas, however, would be a candidate for this kind of statement. See the open problems for details.

8 Open problems

Several future directions are indicated. First of all, one motivation for the definition W_1^1 was to simplify the axioms as much as possible, yet we were unable to limit induction to strict $\Sigma_1^{\mathcal{B}}$ formulas. One problem, then, is to prove the replacement theorems of W_1^1 with this more restricted induction. There does not seem to be any good reason why this should not be possible. On the other hand, Cook and Thapen [11] have recently used KPT-like witnessing theorems to show independence of certain replacement schemes from various theories of bounded arithmetic, and their techniques may apply in this case.

Next, there are some unresolved technical issues regarding BPLK: The most pressing is to eliminate the substitution rule (analogously to in G) but at least the current proof of BPLK's p-equivalence to G seems to rely on this rule in an essential way. See [21] for details.

Another idea is to extend W_1^1 to obtain theories for higher complexity classes. For example, by analogy to V_2^1, extending the induction in W_1^1 to full induction on the strings should yield a theory for EXPTIME, but this would be inelegant to state (although a more natural formulation may exist). Nevertheless, it should be possible to obtain theories for each level of the exponential-time hierarchy in this way, and with more work, for the linear-exponential-time hierarchy and others.

Finally, the idea of having free function symbols in a BPLK proof seems quite general and suggests a direction for even stronger proof systems obtained by allowing function symbol quantifiers in a new kind of BPLK proof. Indeed, this would seem to be a modern version of the *Protothetic* of Stanisław Leśniewski [18] and would hopefully match the stronger theories envisaged in the previous paragraph.

9 Acknowledgment

Many thanks to Stephen Cook for countless helpful discussions on this topic. Thanks also to the reviewers for several important comments.

References

[1] S. Buss. *Bounded Arithmetic.* Bibliopolis, Naples, 1986.
[2] Samuel Buss, Jan Krajíček, and Gaisi Takeuti. On provably total functions in bounded arithmetic theories R_3^i, U_2^i and V_2^i. In Peter Clote and Jan Krajíček, editors, *Arithmetic, proof theory and computational complexity*, pages 116–61. Oxford University Press, Oxford, 1993.
[3] Samuel R. Buss. Relating the bounded arithmetic and polynomial time hierarchies. *Annals of Pure and Applied Logic*, 75(1–2):67–77, 12 September 1995.
[4] Samuel R. Buss, editor. *Handbook of Proof Theory.* Elsevier Science B. V., Amsterdam, 1998.
[5] Samuel R. Buss. Polynomial-size frege and resolution proofs of *st*-connectivity and hex tautologies. Typewritten manuscript, 2003.

[6] Mario Chiari and Jan Krajíček. Witnessing functions in bounded arithmetic and search problems. *The Journal of Symbolic Logic*, 63(3):1095–1115, September 1998.

[7] P. Clote and G. Takeuti. Bounded arithmetic for NC, ALogTIME, L and NL. *Annals of Pure and Applied Logic*, 56(1–3):73–117, 29 April 1992.

[8] S. Cook and A. Kolokolova. A second-order system for polytime reasoning using Grädel's theorem. In *16th Annual IEEE Symposium on Logic in Computer Science (LICS '01)*, pages 177–186, Washington - Brussels - Tokyo, June 2001. IEEE.

[9] S. A. Cook. CSC 2429S: Proof Complexity and Bounded Arithmetic. Course notes, URL: "http://www.cs.toronto.edu/∼sacook/csc2429h", Winter 2002.

[10] Stephen Cook and Michael Soltys. Boolean programs and quantified propositional proof systems. *Bulletin of the Section of Logic*, 28(3), 1999.

[11] Stephen Cook and Neil Thapen. The strength of replacement in weak arithmetic. In *LICS04*, 2004. To appear.

[12] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In *Conference Record of Seventh Annual ACM Symposium on Theory of Computing*, pages 83–97, Albuquerque, New Mexico, 5–7 May 1975.

[13] Stephen A. Cook. Relating the provable collapse of P to NC^1 and the power of logical theories. *DIMACS Series in Discrete Math. and Theoretical Computer Science*, 39, 1998.

[14] Jan Krajíček. On Frege and Extended Frege proof systems. In *P. Clote, J. Remmel (eds.): Feasible Mathematics II*, pages 284–319. Birkhäuser, Boston, 1995.

[15] Jan Krajíček and Pavel Pudlák. Quantified propositional calculi and fragments of bounded arithmetic. *Zeitschr. f. Mathematikal Logik u. Grundlagen d. Mathematik*, 36:29–46, 1990.

[16] Jan Krajíček, Pavel Pudlák, and Gaisi Takeuti. Bounded arithmetic and the polynomial hierarchy. *Annals of Pure and Applied Logic*, 52(1–2):143–153, 1991.

[17] Jan Krajíček and Gaisi Takeuti. On bounded Σ^1_1 polynomial induction. In S. R. Buss and P. J. Scott, editors, *FEASMATH: Feasible Mathematics: A Mathematical Sciences Institute Workshop*, pages 259–80. Birkhauser, 1990.

[18] Stanisław Leśniewski. Grundzüge eines neunen Systems der Grundlagen der Mathematik. *Fundamenta Mathematicae*, 14:1–81, 1929.

[19] Alexander A. Razborov. An equivalence between second order bounded domain bounded arithmetic and furst order bounded arithmetic. In Peter Clote and Jan Krajíček, editors, *Arithmetic, proof theory and computational complexity*, pages 247–77. Oxford University Press, Oxford, 1993.

[20] Stefan Reisch. Hex ist PSPACE-vollständig. *Acta Informatica*, 15:167–191, 1981.

[21] Alan Skelley. Relating the PSPACE reasoning power of Boolean programs and quantified Boolean formulas. Master's thesis, University of Toronto, 2000. Available from ECCC in the 'theses' section.

[22] Alan Skelley. Propositional PSPACE reasoning with Boolean programs versus quantified Boolean formulas. In *ICALP*, volume 3142 of *Lecture Notes in Computer Science*, pages 1163–1175. Springer, 2004.

[23] Michael Soltys. A model-theoretic proof of the completeness of LK proofs. Manuscript, available on author's web page, 1999.

[24] Gaisi Takeuti. RSUV isomorphism. In Peter Clote and Jan Krajíček, editors, *Arithmetic, proof theory and computational complexity*, pages 364–86. Oxford University Press, Oxford, 1993.

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

A Appendix

A.1 Full proof of Theorem 2

Proof. $\leftarrow$: This direction of the equivalence, namely that for $\phi(X, y, \mathcal{X}) \in \Sigma_1^{\mathcal{B}}$

$$W_1^1 \vdash \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]}) \supset \forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X})$$

is immediate.

$\rightarrow$: The existence of a proof in W_1^1 of this direction of the equivalence is itself proved by structural induction on ϕ. The base case of the induction is when ϕ is $\Sigma_0^{\mathcal{B}}$. let ψ be $\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X})$. Let $\theta(c)$ be the formula

$$\forall X \leq (y \dot{-} c) \exists \mathcal{X} \forall Y \leq c \phi(X \frown Y, y, \mathcal{X}^{[Y]}).$$

$\theta(0)$ is a simple logical consequence of ψ, and $W_1^1 \vdash \psi \wedge \theta(c) \supset \theta(c+1)$ by use of $\Sigma_0^{\mathcal{B}}$-3COMP to combine two third-order objects (coding the two arrays of third-order objects for all strings of length smaller than y starting with $X \frown 0$ and $X \frown 1$ respectively) into one third-order object coding the array for all strings of length smaller than y starting with X. Thus $W_1^1 \vdash \psi \supset \theta(y)$ by $\forall^2 \Sigma_1^{\mathcal{B}}$-IND, and clearly $W_1^1 \vdash \theta(y) \supset \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]})$.

Now let $k > 0$ and assume the present theorem holds for every member of $\Sigma_1^{\mathcal{B}}$ with fewer than k third-order quantifiers. Let $\phi \in \Sigma_1^{\mathcal{B}}$ have exactly k third-order quantifiers and assume without loss of generality that ϕ is in prenex normal form. (Every formula is provably in W_1^1 equivalent to one in prenex normal form). Then every third-order quantifier in ϕ is existential, and $\phi(X, y, \mathcal{X})$ is of the form $Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \exists \mathcal{Z} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}, X, y, \mathcal{X})$ for some n and ψ with $k-1$ existential third-order quantifiers. Each Q_i is a bounded first- or second-order quantifier and the corresponding $\tilde{a_i}$ is a variable of the appropriate sort. By several applications of the inductive hypothesis we prove

$$\begin{aligned} &Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \exists \mathcal{Z} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}, X, y, \mathcal{X}) \\ &\qquad \supset \exists \mathcal{Z} Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[\tilde{a_1}] ... [\tilde{a_n}]}, X, y, \mathcal{X}). \end{aligned} \tag{1}$$

The inductive hypothesis is not needed for those Q_i that are existential, nor in that case need we add $[\tilde{a_i}]$ to the formula on the right of the equivalence, yet it is harmless and simplifies matters to do so.

Now with $\Sigma_0^{\mathcal{B}}$-3COMP we can prove

$$\begin{aligned} &\exists \mathcal{X} \exists \mathcal{Z} Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[\tilde{a_1}] ... [\tilde{a_n}]}, X, y, \mathcal{X}) \\ &\qquad \supset \exists \mathcal{Z} Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[1][\tilde{a_1}] ... [\tilde{a_n}]}, X, y, \mathcal{Z}^{[2]}) \end{aligned} \tag{2}$$

and thus piecing together implications 1 and 2 we obtain

$$\begin{aligned} &\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X}) \supset \forall X \leq y \exists \mathcal{Z} Q_1 \tilde{a_1} ... \\ &\qquad Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[1][\tilde{a_1}] ... [\tilde{a_n}]}, X, y, \mathcal{Z}^{[2]}). \end{aligned} \tag{3}$$

We may now appeal to the inductive hypothesis once more and apply the current theorem to the right-hand side of the previous implication, which results in

$$\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X}) \supset \exists \mathcal{Z} \forall X \leq y Q_1 \tilde{a_1} ... \\ Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[X][1][\tilde{a_1}]...[\tilde{a_n}]}, X, y, \mathcal{Z}^{[X][2]}).$$

By applying $\Sigma_0^{\mathcal{B}}$-3COMP we can separate in two along the second "co-ordinate" the object $\mathcal{Z}$, quantified in the right-hand side:

$$\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X}) \supset \exists \mathcal{X} \exists \mathcal{Z} \forall X \leq y Q_1 \tilde{a_1} ... \\ Q_n \tilde{a_n} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}^{[X][\tilde{a_1}]...[\tilde{a_n}]}, X, y, \mathcal{X}^{[X]}).$$

The formula

$$\exists \mathcal{X} \forall X \leq y Q_1 \tilde{a_1} ... Q_n \tilde{a_n} \exists \mathcal{Z} \psi(\tilde{a_1}, ..., \tilde{a_n}, \mathcal{Z}, X, y, \mathcal{X}^{[X]})$$

is a logical consequence of the right-hand side of the previous implication and so we have proved

$$\forall X \leq y \exists \mathcal{X} \phi(X, y, \mathcal{X}) \supset \exists \mathcal{X} \forall X \leq y \phi(X, y, \mathcal{X}^{[X]}),$$

as required. □

A.2 Cases omitted from the proof of lemma 9

1. Weakening:
   The POWOs from the hypothesis are modified to take any extra arguments the new formula introduces (free variables or an existential third-order quantifier in the antecedent) and to ignore them. If the formula is added to the succedent and contains a third-order quantifier, a constant-false predicate taking the appropriate arguments is added as the new POWO for the conclusion.
2. Contraction:
   If the contraction occurs in the succedent on a formula ϕ with a third-order quantifier, then one less POWO is required for the conclusion. Construct a new POWO for ϕ that evaluates ϕ on each original POWO in turn (each evaluation is computable in PSPACE) and then behaves like whichever satisfies ϕ, if any. This computation requires only a constant number of bits more than the maximum of the space used by the two original POWOs.
   If the contraction occurs in the antecedent on a formula ϕ with a third-order quantifier, then all original POWOs must be modified to accept one less oracle argument. Each is modified to query the original POWO but now passing the oracle argument from ϕ twice.
3. Exchange, introduction of $\neg$, $\vee$ on the right and $\wedge$ on the left:
   These rules can neither introduce nor eliminate free variables. No third-order quantifiers are added or removed, and no formula with a third-order quantifier is changed, so the POWOs from the hypothesis are used without modification for the conclusion.

4. Introduction of $\vee$ on the left and $\wedge$ on the right:
These inferences have two hypotheses, and the principal formula is $\Sigma_0^{\mathcal{B}}$ and so needs no POWO. Any side formula that is not $\Sigma_0^{\mathcal{B}}$ will have a POWO for each hypothesis. As in the case of contraction, the POWO for such a formula in the conclusion evaluates the formula on each POWO from the hypotheses, and then simulates whichever satisfies it, if any.
5. First- or second-order $\forall$: **left** and $\exists$: **right**:
The conclusion of such an inference may have less free variables than the hypothesis. Taking for example an $\exists$: **right** inference with principal formula $\exists X \phi(X)$ with the corresponding formula in the hypothesis being $\phi(B)$ and B not free in the conclusion, all POWOs for the hypothesis will have B as an argument. If this argument is fixed to the empty string, the resulting set of POWOs will suffice for the conclusion of the inference (unless $\phi \notin \Sigma_0^{\mathcal{B}}$, addressed below). $\forall$: **left** is similar and in the first-order cases one analogously substitutes 0 for eliminated variables.
If $\phi \notin \Sigma_0^{\mathcal{B}}$ then the principal formula of the inference is $\forall A_i \leq s_i \exists \mathcal{A}_i \gamma_i(A_i, \mathcal{A}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})$ and occurs in the antecedent. In addition to the procedure above (substituting the empty string for the eliminated free string variable), the POWOs must be modified so that any query $\mathcal{A}_i(X)$ becomes $\mathcal{A}_i(\lambda, X)$, adding the empty string as an additional argument, since in the conclusion this oracle argument to the POWOs is two-place.
6. First- or second-order $\forall$: **right** and $\exists$: **left**:
As in the previous case free variables are eliminated by such inferences. However, it is not sufficient to substitute a dummy value for them as above since such a value would not witness the new quantifier properly. For example, if the new quantifier is universal on the right and the principal formula is false under some assignment, the POWOs (from the hypothesis) for the remaining formulas expect a value falsifying the principal formula. This value is found by exhaustive search, evaluating the formula on each possible value of the new quantifier (subject to the bound). The POWOs for the conclusion perform this search before querying the POWOs from the hypothesis. The extra search is clearly carried out in polynomial space.
If the principal formula is not $\Sigma_0^{\mathcal{B}}$, then it is $\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})$ and is in the succedent. In this one special case the POWO for δ_i retains the same number of arguments in the conclusion, due to the string quantifier preceding the third-order quantifier. The POWO for δ_i alone is not modified as above, but instead passes the new argument, C_i, to the POWO from the hypothesis, in place of the eliminated free variable.
7. Third-order $\exists$: **left**:
The principal formula is $\exists \mathcal{A}_i \gamma_i(A_i, \mathcal{A}_i, \overline{\mathcal{B}}, \overline{B}, \overline{b})$. All POWOs from the antecedent are modified to accept oracle argument $\mathcal{A}_i$ instead of the free third-order variable eliminated by the quantifier introduction.
8. Third-order $\exists$: **right**:
If the eigenvariable $\mathcal{B}$ occurs in the lower sequent, then the POWO for the principal formula is defined by
$$f(\overline{\mathcal{B}}, \overline{B}, \overline{b}, \overline{\mathcal{A}}, Z) \leftrightarrow \mathcal{B}(Z)$$

If not, analogously to the lower-order cases of this rule, the new quantifier is witnessed by any value and thus the POWO for the new quantifier may ignore its arguments and always return false. Furthermore, a constant-false predicate is supplied in the place of the eliminated variable as an argument to the other POWOs from the hypothesis.

A.3 Cases omitted from the proof of lemma 17

1. Weakening, Exchange, introduction of $\neg$, $\vee$ on the right and $\wedge$ on the left: These cases are all either structural rules or not applicable to formulas with third-order quantifiers and thus the same rule is applied in the BPLK proof. In the case of weakening, the conclusion may have more free variables than hypothesis. In that case new witnessing function symbols are defined to ignore the new arguments and compute the same value as the old ones, and these must be substituted for the old ones (by induction on the structure of the formula it can easily be seen that BPLK can prove each formula equivalent to one with the new function symbols instead).
2. Contraction, introduction of $\vee$ on the left and $\wedge$ on the right: The only obstacle to using the identical propositional rule is that the principal formula of a contraction inference and the side formulas of the two-hypothesis inferences have two ancestors that will in general be witnessed by different witnessing function symbols (if they occur in the succedent). The solution is to define new witnessing function symbols by cases and then for each affected formula prove that the translation witnessed by the new function symbol implies the disjunction of the translations witnessed by the two old symbols.
For example, a side formula $\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i)$ with witnessing function symbols h'_i and h''_i would have new witnessing function symbol
$$h_i := (||\delta_i(C_i, \mathcal{C}_i^{[C_i]})||[h'_i/g_{\mathcal{C}_i}] \wedge h'_i) \vee (||\delta_i(C_i, \mathcal{C}_i^{[C_i]})||[h''_i/g_{\mathcal{C}_i}] \wedge h''_i)$$
in the conclusion.
3. Introduction of a first-order quantifier: These cases are handled by the introduction of the appropriate propositional connective (disjunction or conjunction). In the case of a universal quantifier on the right or of an existential one on the left, proofs for each value of the free variable are concatenated together. In the other cases the proof for the hypothesis is first extended by weakening to add the other disjuncts (conjuncts on the left).
4. Introduction of a second-order quantifier: These cases are handled the same way as in the simulation of G by BPLK, in that essentially a big disjunction or conjunction is constructed over all values of a set of propositional variables.
Additionally, if the principal formula is $\forall C_i \leq t_i \exists \mathcal{C}_i \delta_i(C_i, \mathcal{C}_i)$, then more work is needed. First, a new witnessing function symbol is defined as follows:
$$h'_i(\overline{p}, \overline{q}) := (\overline{p} = \overline{r} \wedge h_i(\overline{q}))$$

where $\overline{r}$ are the propositional variables associated with C_i, $\overline{p}$ are precisely as numerous as $\overline{r}$ and $\overline{q}$ are the same variables as the arguments to the original h_i. Then, a derivation is inserted proving

$$||\delta_i(C_i, \mathcal{C}_i)||[h_i/g_{\mathcal{C}_i}] \longrightarrow ||\delta_i(C_i, \mathcal{C}_i^{[C_i]})||[h_i'/g_{\mathcal{C}_i}].$$

The second-order quantifier introduction is then handled as usual.

5. Introduction of a third-order quantifier:
   These cases are easy: On the left, this amounts to renaming the arguments to the witnessing function symbols (from free variables to a free function symbol) and on the right it means producing a new witnessing function symbol defined equivalent to the existing free function symbol for that variable and substituting it into the sequent.