The Equivalence of Theories that Characterize **ALogTime**

Phuong Nguyen
University of Toronto
pnguyen@cs.toronto.edu

October 12, 2007

Abstract

A number of theories have been developed to characterize **ALogTime** (or uniform $\mathbf{NC}^1$, or just $\mathbf{NC}^1$), the class of languages accepted by alternating logtime Turing machines, in the same way that Buss's theory $\mathbf{S}^1_2$ characterizes polytime functions. Among these, $\mathbf{ALV}'$ (by Clote) is particularly interesting because it is developed based on Barrington's theorem that the word problem for the permutation group S_5 is complete for **ALogTime**. On the other hand, **ALV** (by Clote), $\mathbf{T}^0\mathbf{NC}^0$ (by Clote and Takeuti) as well as Arai's theory $\mathbf{AID} + \mathbf{\Sigma}^B_0\text{-}\mathbf{CA}$ and its two-sorted version $\mathbf{VNC}^1$ (by Cook and Morioka) are based on the circuit characterization of **ALogTime**. While the last three theories have been known to be equivalent, their relationship to $\mathbf{ALV}'$ has been an open problem. Here we show that $\mathbf{ALV}'$ is indeed equivalent to the other theories.

1 Introduction

The class **ALogTime** consists of languages computable by alternating Turing machines (with random access to inputs) in time logarithmic of the input length. It can be also defined as the class of languages accepted by uniform Boolean circuits with constant fan-ins and logarithmic depth (uniform $\mathbf{NC}^1$, or just $\mathbf{NC}^1$). Here the notion of uniformity is robust: we can use either U_E-uniformity (i.e., the circuits are described by *extended connection language* [Ruz81]), DLOGTIME-uniformity, or **FO**-uniformity [BIS90].

A number of first-order logical theories have been proposed for characterizing $\mathbf{NC}^1$ in the style that Buss's theories $\mathbf{S}^i_2$ [Bus86] characterize the polynomial time hierarchy. These include the single-sorted theory **ALV** [Clo90a], $\mathbf{ALV}'$ [Clo93], $\mathbf{T}^0\mathbf{NC}^0$ [CT95], $\mathbf{AID} + \mathbf{\Sigma}^B_0\text{-}\mathbf{CA}$ [Ara00], T_1 [Pit00] as well as the two-sorted theory $\mathbf{VNC}^1$ [CM05, NC05]. Except for $\mathbf{ALV}'$, the theories are developed using intuitively the fact that the Formula Value problem is complete for $\mathbf{NC}^1$ [Bus87]; it is straightforward to show that they are equivalent. The theory $\mathbf{ALV}'$, on the other hand, is developed using the fact that the word problem for the group S_5 is complete for $\mathbf{NC}^1$ [Bar89]. Since $\mathbf{ALV}'$ is an equational theory, we will use its quantified version **QALV** [Coo98].

All these theories are "minimal" in the sense that they each have a conservative extension which is a universal theory over the functions in $\mathbf{NC}^1$. (**QALV** is already a universal theory over functions in $\mathbf{NC}^1$.) This fact suggests that **QALV** is equivalent to the other theories; however, a proof of this has not been established. We fill this gap by showing that **QALV** is RSUV "isomorphic" [Tak93, Raz93] to $\mathbf{VNC}^1$.

Other (less elegant) theories that characterize $\mathbf{NC}^1$ also include the first-order theories $\mathbf{TNC}^0$ and $\mathbf{T}^\star\mathbf{NC}^0$ [CT95] and second-order theory $\mathbf{A}^{\log}$ [CT92]. The first two use the notion of *essentially sharply bounded* (esb) formulas *in a theory*; the difference between these two is that $\mathbf{TNC}^0$ contains an inference rule called *esb-bounded successive nomination*, while $\mathbf{T}^\star\mathbf{NC}^0$ contains the function *tree* which evaluates a Boolean circuit with bounded fan-in and logarithmic depth. It might be possible to carry out our formalization in the corresponding theories to show that $\mathbf{TNC}^0$ and $\mathbf{T}^\star\mathbf{NC}^0$ are equivalent, but we will not go into further details here.

The second-order theory $\mathbf{A}^{\log}$ has a complicated definition based on alternating logtime Turing machines. It might turn out that it is RSUV isomorphic to some of the first-order theories discussed above. Similarly, Pitt's theory T_1 can be shown equivalent to other theories. However, in this paper we will focus our attention on the theories mentioned in the abstract.

1.1 Overview of the Proofs

The theory $\mathbf{ALV}'$ is an equational single-sorted theory defined using a recursion operation called Bounded Recursion on Notation (BRN). We will define a corresponding operation for two-sorted functions called Bounded Number Recursion (BNR), and use it to define the two-sorted universal theory **VALV**. The vocabulary of **VALV** is defined inductively using $\mathbf{AC}^0$-reduction and BNR to include all $\mathbf{NC}^1$ functions; the axioms of **VALV** are based on $\mathbf{AC}^0$-reduction and the newly defined operation. It is straightforward to show that **VALV** and $\mathbf{ALV}'$ are RSUV isomorphic. The main task remains is to show that **VALV** is a conservative extension of $\mathbf{VNC}^1$.

The proof that the functions of **VALV** are definable in $\mathbf{VNC}^1$ is by induction on the number of operations used in defining each function. The induction step is trivial if we are able to use comprehension axioms for $\mathbf{\Delta}_1^B$ formulas: using a standard translation of formulas involving new functions, each quantifier-free axiom of **VALV** is provably equivalent to a $\mathbf{\Delta}_1^B$ formula. However, it is unlikely that $\mathbf{VNC}^1$ proves such comprehension axioms, so here we have to carefully unwind the axioms of **VALV**. We will define the notion of *aggregate functions* and prove general results which are useful for similar situations. (This notion has been successfully used in [CN06] to develop theories corresponding to a number of subclasses of polytime.)

The other direction requires showing that **VALV** extends $\mathbf{VNC}^1$. The non-trivial task here is to formalize Barrington's reduction from the circuit value problem for bounded fan-in logarithmic depth circuits to the word problem for S_5. So far, this task seems to be the only reason for the absence of a valid claim about the equivalence between the above mentioned theories.

1.2 Organization

The paper is organized as follows. In Section 2 we present some preliminaries and the two-sorted theory $\mathbf{VNC}^1$, the notion of RSUV isomorphism, and show that $\mathbf{T}^0\mathbf{NC}^0$ is RSUV isomorphic to $\mathbf{VNC}^1$. Then in Section 3 we define **VALV**, show that **QALV** is RSUV isomorphic to **VALV**, and state our main theorem. The proof of the main theorem is given in Section 4. Finally, Section 5 contains some concluding remarks.

2 Preliminaries

The two-sorted language that we are using has *number* variables that ranges over $\mathbb{N}$, and *string* (or *set*) variables that ranges over finite subsets of $\mathbb{N}$ (interpreted as finite binary strings). The vocabulary $\mathcal{L}_A^2$ consists of the usual functions and predicates for numbers:

$$0, 1, +, \cdot\, ; =, \leq$$

and set membership $t \in X$ (or simply $X(t)$), string length $|X|$ and set equality $=_2$ (we will often drop the subscript 2). The length function $|X|$ is a number which is 0 if X is empty, and 1 plus the largest member of X otherwise. (So $|X|$ is roughly the length of the binary string corresponding to X.)

There are two types of terms: *number* terms which are built from $0, 1, |X|$ using $+, \cdot$; and *string* terms—over $\mathcal{L}_A^2$ the only string terms are the string variables $X, Y, \ldots$. Formulas are built from the atomic formulas

$$t = u,\ t \leq u,\ X(t),\ X = Y$$

(where u, t are number terms, X, Y are string variables) using the connectives $\wedge, \vee, \neg$ and the quantifiers. Here there are *number* quantifiers $\forall x, \exists x$ and *string* quantifiers $\forall X, \exists X$.

The bounded number quantifiers $\forall x \leq t, \exists x \leq t$ are defined in the usual way, while the bounded string quantifiers $\forall X \leq t\, \varphi$ and $\exists X \leq t\, \varphi$ stand for $\forall X(|X| \leq t \supset \varphi)$ and $\exists X(|X| \leq t \wedge \varphi)$, respectively.

$\mathbf{\Sigma}_0^B$ is the set of all bounded formulas where the only quantifiers are bounded number quantifiers. $\mathbf{\Sigma}_1^B$ formulas have a (possibly empty) block of the bounded string quantifier $\exists X \leq t$ followed by a $\mathbf{\Sigma}_0^B$ formula. ($\mathbf{\Sigma}_1^B$ corresponds to *strict* $\Sigma_1^{1,b}$ [Kra95])

The base theory $\mathbf{V}^0$ [Coo05], called Σ_0^p-comp in [Zam96] and $I\Sigma_0^{1,b}$ (without #) in [Kra95], is axiomatized by the set 2-**BASIC** given in Figure 1, together with the $\mathbf{\Sigma}_0^B$-**COMP**, i.e., the comprehension scheme:

$$\exists X \leq y \forall z < y(X(z) \leftrightarrow \varphi(z)), \tag{1}$$

B1. $x + 1 \neq 0$	**B7.** $(x \leq y \wedge y \leq x) \supset x = y$
B2. $x + 1 = y + 1 \supset x = y$	**B8.** $x \leq x + y$
B3. $x + 0 = x$	**B9.** $0 \leq x$
B4. $x + (y + 1) = (x + y) + 1$	**B10.** $x \leq y \vee y \leq x$
B5. $x \cdot 0 = 0$	**B11.** $x \leq y \leftrightarrow x < y + 1$
B6. $x \cdot (y + 1) = (x \cdot y) + x$	**B12.** $x \neq 0 \supset \exists y \leq x(y + 1 = x)$
L1. $X(y) \supset y < \lvert X\rvert$	**L2.** $y + 1 = \lvert X\rvert \supset X(y)$
SE. $[\lvert X\rvert = \lvert Y\rvert \wedge \forall i < \lvert X\rvert(X(i) \leftrightarrow Y(i))] \;\supset\; X = Y$	

Figure 1: 2-**BASIC**

where $\varphi(z)$ is any $\mathbf{\Sigma}_0^B$ formula not containing X (but may contain other free variables).

Note that bounded number induction on $\mathbf{\Sigma}_0^B$ formulas is provable in $\mathbf{V}^0$. So $\mathbf{V}^0$ extends $\mathbf{I\Delta}_0$, and thus a number of properties of numbers that are provable in $\mathbf{I\Delta}_0$ are also provable in $\mathbf{V}^0$.

In defining two-sorted complexity classes, we consider functions and relations over both sorts. The number arguments are represented as unary strings, while the set arguments are represented as binary strings. Thus the numbers are used mainly for indexing the bits of the strings.

For a complexity class **C**, let **FC** be the class of polynomially bounded string functions whose bitgraph is in **C** and polynomially bounded number functions whose graph is in **C**.

The class uniform $\mathbf{AC}^0$ (or just $\mathbf{AC}^0$) has several equivalent definitions, including **LTH** (the log time hierarchy) or **FO** (describable by first-order formulas using $<$ and *Bit* predicates) [BIS90, Imm99]. In the current setting we have:

Theorem 2.1 ([Imm99, CN06])**.** *A relation $R(\vec{x}, \vec{X})$ is in $\mathbf{AC}^0$ if and only if it is represented by a $\mathbf{\Sigma}_0^B$ formula.*

Furthermore:

Theorem 2.2. *A function is in $\mathbf{FAC}^0$ if and only if it is provably total in $\mathbf{V}^0$.*

In order to encode sequences of numbers and strings, we use the pairing function

$$\langle x, y\rangle =_{\text{def}} (x + y)(x + y + 1) + 2y$$

A sequence of strings $X_1, X_2, \ldots$ can be viewed as a 2-dimensional array:

$$X_i(x) \leftrightarrow Z(\langle i, x\rangle)$$

We often write $Z(i, x)$ for $Z(\langle i, x\rangle)$. Define the "row" function $Row(z, Z)$ (or also $Z^{[z]}$) by

$$|Row(z, Z)| \leq |Z| \;\wedge\; Row(z, Z)(x) \leftrightarrow Z(z, x) \tag{2}$$

A sequence of number can be encoded by a string using the function $seq(x, Z)$ (or also $(Z)^x$) defined as follows:

$$y = seq(x, Z) \leftrightarrow (y < |Z| \wedge Z(x, y) \wedge \forall z < y \neg Z(x, z)) \vee \\ (\forall z < |Z| \neg Z(x, z) \wedge y = |Z|) \quad (3)$$

It is not hard to see that the theory $\mathbf{V}^0(Row, seq)$ is a conservative extension of $\mathbf{V}^0$. Also, the Multiple Comprehension axioms

$$\exists Y \leq \langle b_1, ..., b_k \rangle \forall x_1 < b_1 ... \forall x_k < b_k (Y(\vec{x}) \leftrightarrow \varphi(\vec{x})) \quad (4)$$

where φ is a $\mathbf{\Sigma}_0^B$ formula, are provable in $\mathbf{V}^0$.

2.1 The Theory $\mathbf{VNC}^1$

The theory $\mathbf{VNC}^1$ [CM05, NC05] originated from the theory **AID** [Ara00]. The idea comes from the fact that the problem of evaluating a balanced Boolean formula given the values of its propositional variables is complete for $\mathbf{NC}^1$ (the problem is still complete for $\mathbf{NC}^1$ when the formula is not required to be balance, see [Bus87]). In fact, $\mathbf{VNC}^1$ is axiomatized by $\mathbf{V}^0$ together with an axiom that describes a (polytime) algorithm solving this problem.

In particular, consider a monotone Boolean formula represented as a circuit with fan-in 2, i.e., a binary tree. The circuit will be encoded using the idea of a heap structure. Consider a binary tree/circuit H with $(2a - 1)$ nodes (a leaves and $(a - 1)$ inner nodes). The a leaves of H are numbered $a, \ldots, (2a - 1)$, and the two children of an inner node x are $2x$ and $(2x + 1)$.

Each inner node x $(1 \leq x \leq a - 1)$ is labeled with either $\wedge$ or $\vee$. Therefore to encode H we need just a string G of length $\leq a$ so that $G(x)$ encodes the label of node x of H. Here we let $G(x)$ hold if and only if node x of H is an $\wedge$-gate. Let

$$\delta_{MFVP}(a, G, I, Y) \equiv \ \forall x < a, \ (Y(x + a) \leftrightarrow I(x)) \ \wedge \ [0 < x \supset \\ Y(x) \leftrightarrow [(G(x) \wedge Y(2x) \wedge Y(2x + 1)) \vee (\neg G(x) \wedge (Y(2x) \vee Y(2x + 1)))]]$$

then $Y(x)$ is the value of the gate x in H. Define

$$MFVP \equiv \forall a \forall G \forall I \exists Y \, \delta_{MFVP}(a, G, I, Y) \quad (5)$$

Computing such Y is illustrated in Figure 2.

Definition 2.3. $\mathbf{VNC}^1$ *is the theory over* $\mathcal{L}_A^2$ *which is axiomatized by* $\mathbf{V}^0$ *and* *MFVP*.

The association between $\mathbf{NC}^1$ and $\mathbf{VNC}^1$ is as follows:

Theorem 2.4. *A function is in* $\mathbf{FNC}^1$ *if and only if it is provably total in* $\mathbf{VNC}^1$.

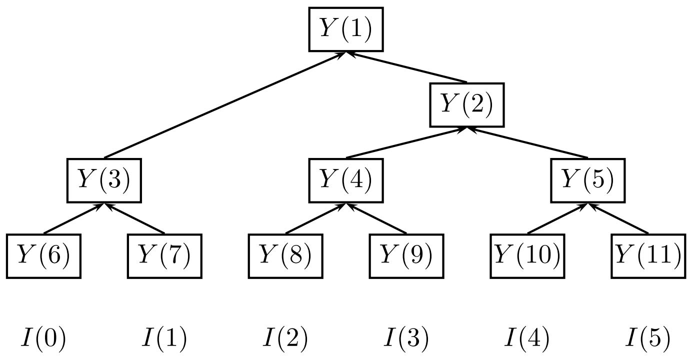

Figure 2: Computing Y which satisfies $\delta_{MFVP}(a, G, I, Y)$ for $a = 6$ (G is not shown).

The theorem can be proved using the fact that $\mathbf{FNC}^1$ consists of precisely all functions that are $\mathbf{AC}^0$-reducible to *Fval*, the function that computes a Y satisfying $\delta_{MFVP}(a, G, I, Y)$ given a, G, I:

$$Fval(a, G, I) = Y \leftrightarrow (|Y| \leq 2a \wedge \delta_{MFVP}(a, G, I, Y)) \tag{6}$$

We also need the nontrivial fact that the function $Fval^\star$ (see Definition 4.2 below) is provably total in $\mathbf{VNC}^1$. This fact follows from Theorems 4.5 below. (See [CN06] for more details.)

The original definition of $\mathbf{VNC}^1$ [CM05] uses the axiom scheme $\mathbf{\Sigma}_0^B$-*TreeRec* instead of *MFVP*. $\mathbf{\Sigma}_0^B$-*TreeRec* is the set of axioms of the form

$$\exists Y \forall x < a,\ [(Y(x{+}a) \leftrightarrow \psi(x)) \wedge (0 < x \supset (Y(x) \leftrightarrow \varphi(x)[Y(2x), Y(2x{+}1)]))] \tag{7}$$

where $\psi(x)$ is a $\mathbf{\Sigma}_0^B$ formula, $\varphi(x)[p, q]$ is a $\mathbf{\Sigma}_0^B$ formula which contains two Boolean variables p and q, and Y does not occur in ψ and φ.

We will show that the two definitions are equivalent. Since *MFVP* is an instance of the $\mathbf{\Sigma}_0^B$-*TreeRec* axiom scheme, we need only to show that $\mathbf{\Sigma}_0^B$-*TreeRec* is provable in $\mathbf{VNC}^1$. Later, we will show that $\mathbf{VNC}^1$ proves yet other generalizations of $\mathbf{\Sigma}_0^B$-*TreeRec* (Theorem 4.4 and 4.5).

Theorem 2.5. *The $\mathbf{\Sigma}_0^B$-TreeRec axiom scheme is provable in $\mathbf{VNC}^1$.*

Proof. Given a, ψ and φ, the idea is to construct a (large) treelike circuit (b, G) with input I so that from $Fval(b, G, I)$ we can extract Y (using $\mathbf{\Sigma}_0^B$-**COMP**) that satisfies (7). Our construction below will show that b, G and I can be defined from a, ψ, φ using $\mathbf{AC}^0$ functions.

Notice that in our circuit there are only $\wedge$- and $\vee$-gates and constants 0, 1, while the "gates" $\varphi(x)[p, q]$ in (7) can be any of the sixteen Boolean functions in two variables p, q. Therefore, for each "gate" $\varphi(x)[p, q]$ we will (uniformly) construct binary subtree of constant depth that simulates $\varphi(x)[p, q]$.

Let

$$\beta_1, \ldots, \beta_8, \beta_9 \equiv \neg\beta_1, \ldots, \beta_{16} \equiv \neg\beta_8$$

be the sixteen Boolean functions in two variables p, q. Each β_i can be computed by a binary and-or tree of depth 2 with inputs among 0, 1, p, q, $\neg p$, $\neg q$. For $1 \leq i \leq 16$, let X_i be defined by

$$X_i(x) \leftrightarrow (x < a \wedge \varphi(x)[p, q] \leftrightarrow \beta_i(p, q))$$

Then,

$$\varphi(x)[p, q] \leftrightarrow \bigvee_{i=1}^{16} (X_i(x) \wedge \beta_i(p, q))$$

Consequently, $\varphi(x)[p, q]$ can be computed by a binary and-or tree T_x of depth 7 whose inputs are $0, 1, p, \neg p, q, \neg q, X_i(x)$. Similarly, $\neg\varphi(x)[p, q]$ is computed by a binary and-or tree T'_x having the same depth and set of inputs. Our large tree (b, G) has one copy of T_1, and in general for each copy of T_x or T'_x, there are multiple copies of $T_{2x}, T_{2x+1}, T'_{2x}, T'_{2x+1}$ that supply the inputs $Y(2x), Y(2x+1), \neg Y(2x), \neg Y(2x+1)$, and other trivial trees that provide inputs 0, 1, $X_i(x)$ $(1 \leq i \leq 16)$.

Finally, the input I is defined by $I(x) \leftarrow \psi(x)$ for $x < a$. $\square$

2.2 RSUV Isomorphism

The equivalence between a single-sorted theory $\mathcal{T}_1$ and a two-sorted theory $\mathcal{T}_2$ is known by the notion of RSUV isomorphism [Tak93, Raz93, Kra90]. Essentially, to show that $\mathcal{T}_1$ is RSUV isomorphic to $\mathcal{T}_2$ we need to (a) construct from each model $\mathcal{M}$ of $\mathcal{T}_1$ a model $\mathcal{M}^\sharp$ of $\mathcal{T}_2$ whose second sort universe is the universe M of $\mathcal{M}$, and whose first sort universe is the subset $\log(M) = \{|u| \mid u \in M\}$; and (b) construct from each model $\mathcal{N}$ of $\mathcal{T}_2$ a model $\mathcal{N}^\flat$ of $\mathcal{T}_1$ whose universe is the second sort universe of $\mathcal{N}$. These constructions have the property that $\mathcal{M}$ and $(\mathcal{M}^\sharp)^\flat$ are isomorphic, and so are $\mathcal{N}$ and $(\mathcal{N}^\flat)^\sharp$.

These semantic mappings between models are associated with syntactic translations of formulas between the languages of $\mathcal{T}_1$ and $\mathcal{T}_2$. In particular, each two-sorted formula φ is translated into a single-sorted formula $\varphi^\flat$ such that

$$\mathcal{M}^\sharp \models \forall\varphi \text{ if and only if } \mathcal{M} \models \forall\varphi^\flat$$

for any model $\mathcal{M}$ of $\mathcal{T}_1$, and each single-sorted formula ψ is translated into a two-sorted formula $\psi^\sharp$ so that

$$\mathcal{N}^\flat \models \forall\psi \text{ if and only if } \mathcal{N} \models \forall\psi^\sharp$$

for any model $\mathcal{N}$ of $\mathcal{T}_2$.

Proving RSUV isomorphism is often tedious but straightforward in many cases. It turns out that the hard work are often required for interpreting certain functions in the appropriate structures; for example, interpreting the multiplication function in $\mathbf{VTC}^0$ [Ngu04]. In the case of **QALV** and $\mathbf{VNC}^1$, a difficulty can be seen in interpreting the function *Fval* in a model for **QALV**.

2.3 RSUV Isomorphism between $\mathbf{T^0NC^0}$ and $\mathbf{VNC^1}$

The single-sorted theory $\mathbf{T^0NC^0}$ [CT95] has vocabulary

$$0, 1, +, pad(x,y), \dot{-}\,, |x|, x\#y, \lfloor x/2 \rfloor, MSP, and, or, tree;\ \leq, = \tag{8}$$

where

$$x\#y = 2^{|x|\cdot|y|}, \qquad pad(x,y) = x \cdot 2^{|y|}, \qquad MSP(x,y) = \lfloor x/2^y \rfloor$$

$$and(x,y) = \begin{cases} 1 & \text{if } x \geq 1, y \geq 1 \\ 0 & \text{otherwise} \end{cases} \qquad or(x,y) = \begin{cases} 0 & \text{if } x = y = 0 \\ 1 & \text{otherwise} \end{cases}$$

and $tree(x)$ [Clo90b] is the function that evaluates a perfect and–or tree with leaves being the bits of x. Notice that *tree* is a complete function for $\mathbf{NC^1}$.

The axioms for $\mathbf{T^0NC^0}$ consists of the defining axioms for the symbols above, together with the *sb*-Bit Comprehension Axioms

$$\exists y < 2^{|s(\vec{a})|} \forall i < |s(\vec{a})|\, [Bit(i,y) = 1 \leftrightarrow A(\vec{a}, i)]$$

(where A is a sharply bounded formula) and the inference rule *sb*-LIND

$$\frac{A(a), \Gamma \longrightarrow \Delta, A(a+1)}{A(0), \Gamma \longrightarrow \Delta, A(|t|)}$$

(where A is a sharply bounded formula, and a does not occur in the bottom sequent). Here $Bit(i,a)$ is the i-th bit of y and is defined by

$$Bit(i,a) = MSP(a,i) \dot{-} 2 \cdot \lfloor MSP(a,i)/2 \rfloor$$

Notice that the language of $\mathbf{T^0NC^0}$ does not contain the multiplication function $x \cdot y$. (but contains some restriction of it).

Theorem 2.6. $\mathbf{T^0NC^0}$ *and* $\mathbf{VNC^1}$ *are RSUV isomorphic.*

Proof Outline. To interpret $\mathbf{T^0NC^0}$ in $\mathbf{VNC^1}$ we use the fact [CN06] that $\mathbf{VNC^1}(\mathcal{L}_{\mathbf{FNC^1}})$ proves $\mathbf{\Sigma_0^B}(\mathcal{L}_{\mathbf{FNC^1}})$-**COMP**, the comprehension axioms for all $\mathbf{\Sigma_0^B}$ formulas over $\mathcal{L}_{\mathbf{FNC^1}}$. Here $\mathcal{L}_{\mathbf{FNC^1}}$ consists of all $\mathbf{NC^1}$ functions and is defined as the $\mathbf{AC^0}$ closure of *Fval*. This fact also implies that $\mathbf{VNC^1}(\mathcal{L}_{\mathbf{FNC^1}})$ proves $\mathbf{\Sigma_0^B}(\mathcal{L}_{\mathbf{FNC^1}})$-**IND**, the induction axioms for all $\mathbf{\Sigma_0^B}$ formulas over $\mathcal{L}_{\mathbf{FNC^1}}$.

Thus the function *tree* can be interpreted by *Fval*, and other functions in (8) can be interpreted by appropriate two-sorted $\mathbf{AC^0}$ functions. The *sb*-Bit Comprehension Axioms translate into (a subset of) $\mathbf{\Sigma_0^B}(\mathcal{L}_{\mathbf{FNC^1}})$-**COMP**, and *sb*-LIND translates into (a subset of) $\mathbf{\Sigma_0^B}(\mathcal{L}_{\mathbf{FNC^1}})$-**IND**.

For the other direction, first, the translation of *MFVP* can be proved in $\mathbf{T^0NC^0}$ by making the (implicit) monotone tree-like circuit in *MFVP* a perfect and–or tree. Then, the $\mathbf{\Sigma_0^B}$-**COMP** axioms translate into (a subset of) the *sb*-Bit Comprehension Axiom scheme. Finally, it is straightforward to show that the translation of the other axioms in $\mathbf{VNC^1}$ are provable in $\mathbf{T^0NC^0}$. $\square$

3 The Theory VALV

In this section we introduce **VALV**, the two-sorted version of the theory **QALV** [Coo98, Clo93]. **VALV** is an universal theory with functions from $\mathbf{FNC}^1$ where the defining axioms of the ($\mathbf{NC}^1$-hard) functions are based on the fact that the word problem for S_5 is complete for $\mathbf{NC}^1$ [Bar89]. In Section 3.2 we show that **VALV** is RSUV isomorphic to **QALV**.

The main theorem of this paper (Theorem 3.5) that **VALV** is a conservative extension of $\mathbf{VNC}^1$ is stated in Section 3.1. Its proof will be presented in Section 4.

VALV is an extension of $\overline{\mathbf{V}}^0$, an universal conservative extension of $\mathbf{V}^0$ [Coo05].

3.1 VALV

We use the method in [Coo05, NC05] to develop **VALV**. To obtain a quantifier-free defining axioms for functions of $\mathbf{FNC}^1$, the idea is to use $\mathbf{AC}^0$ functions to eliminate bounded number quantifiers. For example, a bounded number quantifier of the form $\exists x \le t\, \varphi(x)$ can be eliminated using an $\mathbf{AC}^0$ number function that computes the least $x \le t$ that satisfies $\varphi(x)$.

First, **B12** is not a quantifier-free axiom, so we use the predecessor function pd with defining axioms:

$$\mathbf{B12}': \; pd(0) = 0 \qquad\qquad \mathbf{B12}'': \; x \neq 0 \supset pd(x) + 1 = x \tag{9}$$

Also, the extensionality axiom **SE** contains an implicit existential quantifier $\exists i < |X|$. This can be avoided by using the function $f_{\mathbf{SE}}(X, Y)$ which is the least $x < |X|$ that distinguishes X and Y, and $f_{\mathbf{SE}}(X, Y) = |X|$ if no such x exists:

$$\begin{aligned} & f_{\mathbf{SE}}(X, Y) \le |X| \wedge \\ & f_{\mathbf{SE}}(X, Y) < |X| \supset (X(f_{\mathbf{SE}}(X, Y)) \not\leftrightarrow Y(f_{\mathbf{SE}}(X, Y))) \wedge \\ & z < f_{\mathbf{SE}}(X, Y) \supset (X(z) \leftrightarrow Y(z)). \end{aligned} \tag{10}$$

(The defining axiom for $f_{\mathbf{SE}}$ is the instance of (13) below, where $\varphi(z, X, Y) \equiv X(z) \not\leftrightarrow Y(z)$, and $t(X, Y) = |X|$.)

In defining **VALV**, we will use $\mathbf{SE}'$ instead of **SE**:

$$\mathbf{SE}': \qquad (|X| = |Y| \wedge f_{\mathbf{SE}}(X, Y) = |X|) \supset X = Y. \tag{11}$$

To get $\mathbf{AC}^0$-closure of functions we use the following notation. For any formula $\varphi(z, \vec{x}, \vec{X})$ and $\mathcal{L}_A^2$-term $t(\vec{x}, \vec{X})$, let $F_{\varphi,t}(\vec{x}, \vec{X})$ be the string function with defining axiom

$$F_{\varphi,t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X}) \tag{12}$$

Also, let $f_{\varphi,t}(\vec{x}, \vec{X})$ be the least $y < t$ such that $\varphi(y, \vec{x}, \vec{X})$ holds, or t if no such y exists (we write f for $f_{\varphi,t}$, t for $t(\vec{x}, \vec{X})$, and $\ldots$ for $\vec{x}, \vec{X}$):

$$f(\ldots) \le t \;\wedge\; [f(\ldots) < t \supset \varphi(f(\ldots), \ldots)] \;\wedge\; [v < f(\ldots) \supset \neg\varphi(v, \ldots)] \tag{13}$$

The following operation can be used to obtain a two-sorted version of Lind's recursion-theoretic characterization of the logspace functions. It corresponds to the so-called *doubly bounded recursion on notation* $\mathrm{B}_2\mathrm{RN}$ defined in [CT95] for single-sorted functions.

Definition 3.1 (Number Recursion)**.** *A number function* $f(y, \vec{x}, \vec{X})$ *is obtained by* number recursion *from* $g(\vec{x}, \vec{X})$ *and* $h(y, z, \vec{x}, \vec{X})$ *if*

$$f(0, \vec{x}, \vec{X}) = g(\vec{x}, \vec{X}) \tag{14}$$

$$f(y+1, \vec{x}, \vec{X}) = h(y, f(y, \vec{x}, \vec{X}), \vec{x}, \vec{X}) \tag{15}$$

If further $f(y, \vec{x}, \vec{X}) \leq t(y, \vec{x}, \vec{X})$*, then we also say that* f *is obtained by* t*-Bounded Number Recursion from* g *and* h*.*

It will follow from our result that all $\mathbf{NC}^1$ functions can be obtained from the empty set of function by taking closure under $\mathbf{AC}^0$-reduction and the 4-Bounded Number Recursion operation.

Now to define **VALV**, note that the function $f_{g,h}(y, \vec{x}, \vec{X})$ that is defined from $g(\vec{x}, \vec{X})$ and $h(y, z, \vec{x}, \vec{X})$ by 4-Bounded Number Recursion has the following defining axiom (we drop mention of $\vec{x}, \vec{X}$, and write f for $f_{g,h}$):

$$(g \leq 4 \wedge f(0) = g) \vee (g > 4 \wedge f(0) = 0) \tag{16}$$

$$(h(y, f(y)) \leq 4 \wedge f(y+1) = h(y, f(y))) \vee (h(y, f(y)) > 4 \wedge f(y+1) = 0) \tag{17}$$

Definition 3.2. $\mathcal{L}_{\mathbf{FNC}^1}$ *is the smallest set that satisfies*[1]

1) $\mathcal{L}_{\mathbf{FNC}^1}$ *includes* $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}\}$.
2) *For each open formula* $\varphi(z, \vec{x}, \vec{X})$ *over* $\mathcal{L}_{\mathbf{FNC}^1}$ *and term* $t = t(\vec{x}, \vec{X})$ *of* $\mathcal{L}_A^2$ *there is a string function* $F_{\varphi,t}$ *and a number function* $f_{\varphi,t}$ *in* $\mathcal{L}_{\mathbf{FNC}^1}$*.*
3) *For all functions* g, h *of* $\mathcal{L}_{\mathbf{FNC}^1}$*, there is a number function* $f_{g,h}$ *in* $\mathcal{L}_{\mathbf{FNC}^1}$*.*

Definition 3.3. VALV *is the theory over* $\mathcal{L}_{\mathbf{FNC}^1}$ *with the following set of axioms:* **B1**-**B11***,* **L1***,* **L2** *(Figure 1),* **B12**$'$ *and* **B12**$''$ (9)*,* (10)*,* **SE**$'$ (11)*,* (12) *for each function* $F_{\varphi,t}$*,* (13) *for each function* $f_{\varphi,t}$ *of* $\mathcal{L}_{\mathbf{FNC}^1}$*, and* (16)*,* (17) *for each function* $f_{g,h}$ *of* $\mathcal{L}_{\mathbf{FNC}^1}$*.*

For the next theorem, the idea [Coo05] is that the functions $f_{\varphi,t}$ can be used to eliminate bounded number quantifiers. Here $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$-**COMP** is the comprehension axiom scheme (1) over all $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$ formulas, and $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$-**IND** is the *number* induction axiom scheme:

$$\varphi(0) \wedge \forall x(\varphi(x) \supset \varphi(x+1)) \supset \varphi(a)$$

where $\varphi(x)$ is a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$ formula.

[1] The notation $\mathcal{L}_{\mathbf{FNC}^1}$ used in Section 2.3 is from [CN06] where it is defined differently using the function *Fval* with defining axiom (6). In [CN06] an universal theory $\overline{\mathbf{VNC}}^1$ is defined using that definition of $\mathcal{L}_{\mathbf{FNC}^1}$; by the result of this paper, **VALV** and $\overline{\mathbf{VNC}}^1$ are equivalent. From now on, we will be using the definition of $\mathcal{L}_{\mathbf{FNC}^1}$ given here.

Theorem 3.4. **VALV** *proves* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$**-COMP** *and* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$**-IND***.*

Proof. The (bounded) number quantifiers in a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$ formula can be eliminated using the functions $f_{\varphi,t}$ (13). In other words, for each $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$ formula φ there is a quantifier-free formula ψ over $\mathcal{L}_{\mathbf{FNC}^1}$ that is provably equivalent in **VALV** to φ. Therefore we just need to prove the comprehension and induction axioms for quantifier-free formulas over $\mathcal{L}_{\mathbf{FNC}^1}$.

The comprehension axiom (1) for an open formula φ over $\mathcal{L}_{\mathbf{FNC}^1}$ is proved using the fact that the string X in (1) can be taken to be $F_{\varphi,y}$. The induction axioms are easily proved using the axioms for the length function $|X|$ and the comprehension axioms (see also [CN06]). $\square$

Theorem 3.5 (Main Theorem)**.** **VALV** *is a conservative extension of* $\mathbf{VNC}^1$*.*

The proof of this theorem is given in Section 4.

3.2 The RSUV Isomorphism between QALV and VALV

The single-sorted theory $\mathbf{ALV}'$ [Clo93] is an equational theory whose axioms include the defining axioms for some basic $\mathbf{AC}^0$ functions and the functions defined inductively by the so-called Concatenation Recursion on Notation (CRN) and k-Bounded Recursion on Notation (k-BRN). **QALV** [Coo98] is the quantified version of $\mathbf{ALV}'$ obtained essentially by treating the equations of $\mathbf{ALV}'$ as universal axioms.

First, recall that if $h_0(x), h_1(x) \leq 1$, then $f(x)$ is defined by CRN from g, h_0 and h_1 by CRN whenever

$$\begin{aligned} f(0) &= g \\ f(2x) &= s_{h_0(x)}(f(x)) \quad \text{if} \quad x \neq 0 \\ f(2x+1) &= s_{h_1(x)}(f(x)) \end{aligned}$$

where $s_0(x) = 2x, s_1(x) = 2x+1$. (Here f, g, h_0, h_1 might have other parameters.)

It is easy to see that if f is defined by CRN from g, h_0, h_1 then the bits of $f(x)$ are computed in parallel from the bits of g and x using $h_i(x)$. Thus, informally, this operation corresponds to taking $\mathbf{AC}^0$-closure (i.e., defining $f_{\varphi,t}$ and $F_{\varphi,t}$ in Definition 3.3).

Next, a function f is defined by k-BRN from g, h_0 and h_1 provided that

$$\begin{aligned} f(0) &= g \\ f(2x) &= h_0(x, f(x)) \quad \text{if} \quad x \neq 0 \\ f(2x+1) &= h_1(x, f(x)) \end{aligned}$$

and $f(x) \leq k$ for all x. (f, g, h_i might have other parameters.)

Without the bound k, BRN is the single-sorted version of the number recursion (Definition 3.1), and k-BRN is the single-sorted version of k-Bounded Number Recursion.

The above observations and Theorem 3.4 can be used to show that:

Theorem 3.6. **VALV** *and* **ALV**$'$ *are RSUV isomorphic.*

Proof Sketch. Perhaps the only mismatch between **VALV** and **QALV** is that **VALV** is defined using 4-BNR while **QALV** is defined using k-BRN for all $k \in \mathbb{N}$. The remark after the proof of Theorem 3.5 in Section 4.1.3 shows that **VALV** remains essentially the same if 4-BNR is replaced by k-BNR for all $k \in \mathbb{N}$. $\square$

4 Proof of the Main Theorem

4.1 Defining $\mathcal{L}_{\mathbf{FNC}^1}$ in $\mathbf{VNC}^1$

Our task is to show that the functions of $\mathcal{L}_{\mathbf{FNC}^1}$ are provably total in $\mathbf{VNC}^1$, and that $\mathbf{VNC}^1$ proves their defining equations. We will proceed by induction on Definition 3.2. Recall that the functions of $\mathcal{L}_{\mathbf{FNC}^1}$ are defined in stages. Let $\mathcal{L}_n$ be the set of all functions obtained at the end of stage n. For the induction step, consider a function $F_{\varphi,t}$ with defining axiom (12), where φ is a quantifier-free formula over $\mathcal{L}_n$. Then in $\mathbf{VNC}^1(\mathcal{L}_n)$ we can define $F_{\varphi,t}$ by

$$F_{\varphi,t} = Y \;\leftrightarrow\; [\, |Y| \leq t \wedge \forall z < t\, (Y(z) \leftrightarrow \varphi(z))]$$

(Clearly the defining equation (12) of $F_{\varphi,t}$ is provable from this defining axiom.) We need to show that

$$\mathbf{VNC}^1(\mathcal{L}_n) \vdash \exists! Y\, [\, |Y| \leq t \wedge \forall z < t\, (Y(z) \leftrightarrow \varphi(z))]$$

The uniqueness of Y follows from extensionality, and the existence of Y can be proved using the comprehension axiom (1) for φ. Thus we will in fact prove the following claim.

Theorem 4.1. *For $n \geq 0$,* $\mathbf{VNC}^1(\mathcal{L}_n) \vdash \mathbf{\Sigma}_0^B(\mathcal{L}_n)$-**COMP**.

Notice that the standard method of translating a $\mathbf{\Sigma}_0^B(\mathcal{L}_{n+1})$ formula into a formula over $\mathcal{L}_n$ results in a $\mathbf{\Delta}_1^B(\mathcal{L}_n)$ formula (that is, a formula which is equivalent to both a $\mathbf{\Sigma}_1^B(\mathcal{L}_n)$ formula and a $\mathbf{\Pi}_1^B(\mathcal{L}_n)$ formula). It is not known whether $\mathbf{VNC}^1$ proves the comprehension axioms for $\mathbf{\Delta}_1^B$ formulas, so we need to work harder in proving the above theorem.

In Section 4.1.1 below we will introduce the notion of *aggregate functions* [CN06] which is useful in proving the induction step of Theorem 4.1. In Section 4.1.2 we obtain some nontrivial theorems of $\mathbf{VNC}^1$ which will be used for the case where the new function in $\mathcal{L}_{n+1}$ is of the form $f_{g,h}$. The proof of Theorem 4.1 is then presented in Section 4.1.3.

4.1.1 Aggregate Functions

The idea of having the aggregate function for a function is so that we can simultaneously compute it for (polynomially) many values of the inputs. Recall the functions *Row* given in (2) and *seq* in (3).

Definition 4.2 (Aggregate Function). *Suppose that $F(x_1,\ldots,x_k,X_1,\ldots,X_n)$ is a polynomially bounded string function, i.e., for some $\mathcal{L}_A^2$ term t,*

$$|F(\vec{x},\vec{X})| \leq t(\vec{x},|\vec{X}|)$$

Then $F^\star(b,Z_1,\ldots,Z_k,X_1,\ldots,X_n)$ is the string function that satisfies

$$\begin{aligned}&[\ |F^\star(b,\vec{Z},\vec{X})| \leq \langle b, t(|\vec{Z}|,|\vec{X}|)\rangle]\ \wedge\ [F^\star(b,\vec{Z},\vec{X})(w) \leftrightarrow \\ &\quad \exists u < b \exists v < w,\ w = \langle u,v\rangle \wedge\ F((Z_1)^u,\ldots,(Z_k)^u, X_1^{[u]},\ldots,X_n^{[u]})(v)] \quad (18)\end{aligned}$$

Similarly, suppose that $f(x_1,\ldots,x_k,X_1,\ldots,X_n)$ is a polynomially bounded number function, i.e., for some $\mathcal{L}_A^2$ term t, $f(\vec{x},\vec{X}) \leq t(\vec{x},|\vec{X}|)$. Then $f^\star(b,\vec{Z},\vec{X})$ is the string *function that satisfies*

$$\begin{aligned}&[\ |f^\star(b,\vec{Z},\vec{X})| \leq \langle b, 1+t\rangle] \wedge \\ &[f^\star(b,\vec{Z},\vec{X})(w) \leftrightarrow \exists u < b,\ w = \langle u, f((Z_1)^u,\ldots,(Z_k)^u, X_1^{[u]},\ldots,X_n^{[u]})\rangle] \quad (19)\end{aligned}$$

Theorem 4.3. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0$ with vocabulary $\mathcal{L}$, where $\mathcal{L}$ contains the functions Row and seq. Suppose that $\mathcal{T}$ proves $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP**. Let F be a definable string function of $\mathcal{T}$ such that the function $F^\star$ is also definable in $\mathcal{T}$ and $\mathcal{T}(F^\star)$ proves (18). Then $\mathcal{T}(F)$ proves $\mathbf{\Sigma}_0^B(\mathcal{L}\cup\{F\})$-**COMP**. The same is true for a number function f definable in $\mathcal{T}$ for which $f^\star$ is definable in $\mathcal{T}$ and $\mathcal{T}(f^\star)$ proves (19).*

Proof. We will consider the case of extending $\mathcal{L}$ by a string function F. The case where $\mathcal{L}$ is extended by a number function is handled similarly by using number variables w_i instead of the string variables W_i in the argument below.

First, since $\mathcal{T}$ proves $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP** and since $\mathbf{V}^0$ proves the Multiple Comprehension axioms (4) for $\mathbf{\Sigma}_0^B$ formulas, it follows that $\mathcal{T}$ proves the Multiple Comprehension axioms for $\mathbf{\Sigma}_0^B(\mathcal{L})$ formulas.

Claim For any $\mathcal{L}$-terms $\vec{s},\vec{T}$ that contain variables $\vec{z}$, $\mathcal{T}(F)$ proves

$$\exists Y \forall z_1 < b_1 \ldots \forall z_m < b_m Y^{[\vec{z}]} = F(\vec{s},\vec{T}) \quad (20)$$

Proof of the Claim. Since $\mathcal{T}$ proves the Multiple Comprehension axiom scheme for $\mathbf{\Sigma}_0^B(\mathcal{L})$ formulas, it proves the existence of $\vec{X}$ such that $X_j^{[\vec{z}]} = T_j$, for $1 \leq j \leq n$. It also proves the existence of Z_i such that $(Z_i)^{\langle\vec{z}\rangle} = s_i$, for $1 \leq i \leq k$. Now the value of Y that satisfies (20) is just $F^\star(\langle\vec{b}\rangle,\vec{Z},\vec{X})$. □

Let $\mathcal{L}' = \mathcal{L}\cup\{F\}$. We show by induction on the quantifier depth of a $\mathbf{\Sigma}_0^B(\mathcal{L}')$ formula ψ that $\mathcal{T}(F)$ proves

$$\exists Z \leq \langle b_1,\ldots,b_m\rangle \forall z_1 < b_1 \ldots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi(\vec{z}) \quad (21)$$

where $\vec{z}$ are all free number variables of ψ. It follows that $\mathcal{T}(F) \vdash \mathbf{\Sigma}_0^B(\mathcal{L}')$-**COMP**.

For the base case, ψ is quantifier-free. The idea is to replace every occurrence of a term $F(\vec{s},\vec{T})$ in ψ by a new string variable W which has the intended value of $F(\vec{s},\vec{T})$. The resulting formula is $\mathbf{\Sigma}_0^B(\mathcal{L})$, and we can apply the hypothesis.

Formally, suppose that $F(\vec{s}_1,\vec{T}_1),\ldots,F(\vec{s}_k,\vec{T}_k)$ are all occurrences of F in ψ. Note that the terms $\vec{s}_i,\vec{T}_i$ may contain $\vec{z}$ as well as nested occurrences of F. Assume further that $\vec{s}_1,\vec{T}_1$ do not contain F, and for $1 < i \le k$, any occurrence of F in $\vec{s}_i,\vec{T}_i$ must be of the form $F(\vec{s}_j,\vec{T}_j)$, for some $j < i$. We proceed to eliminate F from ψ by using its defining axiom.

Let $W_1,...,W_k$ be new string variables. Let $\overrightarrow{s'_1} = \vec{s}_1$, $\overrightarrow{T'_1} = \vec{T}_1$, and for $2 \le i \le k$, $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$ be obtained from $\vec{s}_i$ and $\vec{T}_i$ respectively by replacing every maximal occurrence of any $F(\vec{s}_j,\vec{T}_j)$, for $j < i$, by $W_j^{[\vec{z}]}$. Thus F does not occur in any $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$, but for $i \ge 2$, $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$ may contain $W_1,\ldots,W_{i-1}$.

By claim above, for $1 \le i \le k$, $\mathcal{T}(F)$ proves the existence of W_i such that

$$\forall z_1 < b_1 \ldots \forall z_m < b_m,\ W_i^{[\vec{z}]} = F(\overrightarrow{s'_i},\overrightarrow{T'_i}) \tag{22}$$

Let $\psi'(\vec{z},W_1,\ldots,W_k)$ be obtained from $\psi(\vec{z})$ by replacing each maximal occurrence of $F(\vec{s}_i,\vec{T}_i)$ by $W_i^{[\vec{z}]}$, for $1 \le i \le k$. Then, by Multiple Comprehension for $\mathbf{\Sigma}_0^B(\mathcal{L})$ and the fact that $\mathcal{L}$ contains *Row*,

$$\mathcal{T} \vdash \exists Z \le \langle b_1,\ldots,b_m\rangle \forall z_1 < b_1 \ldots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi'(\vec{z},W_1,\ldots,W_k).$$

Such Z satisfies (21) when each W_i is defined by (22).

The induction step is straightforward. Consider for example the case $\psi(\vec{z}) \equiv \forall x < t \lambda(\vec{z},x)$. By the induction hypothesis,

$$\mathcal{T}(F) \vdash \exists Z' \forall z_1 < b_1 \ldots \forall z_m < b_m \forall x < t,\ Z'(\vec{z},x) \leftrightarrow \lambda(\vec{z},x).$$

Now $\mathbf{V}^0$ proves the following instance of the Multiple Comprehension axioms:

$$\exists Z \forall z_1 < b_1 \ldots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \forall x < t Z'(\vec{z},x).$$

Therefore $\mathcal{T}(F) \vdash \exists Z \forall \vec{z} < \vec{b}\, Z(\vec{z}) \leftrightarrow \psi(\vec{z})$. $\square$

4.1.2 More Theorems of $\mathbf{VNC}^1$

Here we show that $\mathbf{VNC}^1$ proves some generalization of the $\mathbf{\Sigma}_0^B$-*TreeRec* axiom scheme. The next theorem shows that in $\mathbf{VNC}^1$ we can evaluate a log-depth Boolean circuit with any constant fan-in.

Theorem 4.4. *Suppose that* $2 \le k \in \mathbb{N}$*, and* $\psi(x)$ *and* $\varphi(x)[p_0,\ldots,p_{k-1}]$ *are* $\mathbf{\Sigma}_0^B$ *formulas. Then* $\mathbf{VNC}^1$ *proves*

$$\begin{aligned}\exists Y,\ \forall x < ka,\ a \le x \supset Y(x) \leftrightarrow \psi(x) \wedge \\ \forall x < a,\ Y(x) \leftrightarrow \varphi(x)[Y(kx),\ldots,Y(kx+k-1)]\end{aligned} \tag{23}$$

Proof. We prove for the case $k = 4$; similar arguments work for other cases.

Using Theorem 2.5 we will define a', ψ', φ' so that from Y' that satisfies the $\mathbf{\Sigma}_0^B$-*TreeRec* axiom (7) for a', ψ' and φ' we can obtain Y that satisfies (23) above.

Intuitively, consider Y in (23) as a forest of three trees whose nodes are labeled with $Y(x)$, $x < |Y|$. Then Y has branching factor of 4 (since $k = 4$), and the three trees are rooted at $Y(1)$, $Y(2)$ and $Y(3)$. (See Figure 3.) Note also that each layer in Y corresponds to two layers in the binary tree Y'.

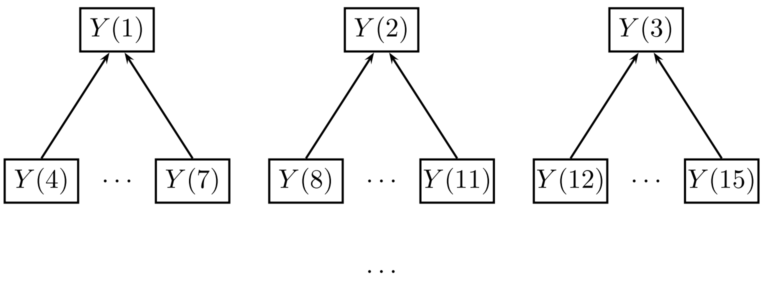

Figure 3: The "tree" Y in Theorem 4.4 when $k = 4$.

We will define an injective map f so that $Y(x) \leftrightarrow Y'(f(x))$. Since the trees rooted at $Y(1)$, $Y(2)$ and $Y(3)$ are disjoint, f is defined so that these trees are the images of disjoint subtrees in the tree Y'. For example, we can choose the subtrees rooted at $Y'(4)$, $Y'(5)$ and $Y'(6)$. Thus,

$$f(1) = 4, f(2) = 5, f(3) = 6$$

In general, consider the function f defined by:

$$f(4^m + y) = 4^{m+1} + y \qquad \text{for } \ 0 \le y < 3 \cdot 4^m$$

(Note that f is provably total in $\mathbf{I\Delta}_0$ [HP93, CN06], and hence also in $\mathbf{V}^0$.)

Now we need

$$\psi'(f(x)) \leftrightarrow \psi(x)$$

for $a \le x < 4a$. Define ψ' so that

$$\psi'(4^{m+1} + y) \leftrightarrow \psi(4^m + y)$$

for $y < 3 \cdot 4^m$ and $a \le 4^m + y < 4a$.

To obtain φ', write $\varphi(x)[p_0, p_1, p_2, p_3]$ in the form

$$\varphi_1(x)[\varphi_2(x)[p_0, p_1], \varphi_3(x)[p_2, p_3]]$$

where φ_i is $\mathbf{\Sigma}_0^B$ with at most 2 Boolean variables, for $1 \le i \le 3$. Define φ' so that

$$\begin{aligned}
\varphi'(4^{m+1}+y)[p,q] &\leftrightarrow \varphi_1(4^m+y)[p,q] && \text{for } y < 3\cdot 4^m \\
\varphi'(2\cdot 4^{m+1}+2y)[p,q] &\leftrightarrow \varphi_2(4^m+y)[p,q] && \text{for } y < 3\cdot 4^m/2 \\
\varphi'(2\cdot 4^{m+1}+2y+1)[p,q] &\leftrightarrow \varphi_3(4^m+y)[p,q] && \text{for } y < 3\cdot 4^m/2
\end{aligned}$$

Finally, let $a' = f(a)$. Let Y' satisfies (7) for a', ψ' and φ', and let Y be such that

$$Y(x) \leftrightarrow Y'(f(x))$$

It is easy to verify that Y satisfies (23). □

The next theorem shows that in $\mathbf{VNC}^1$ we can evaluate multiple inter-connected Boolean circuits with logarithmic depth and constant fan-in.

Theorem 4.5. *Suppose that $1 \le m, \ell \in \mathbb{N}$, and for $1 \le i \le m$, $\psi_i(x,y)$ and $\varphi_i(x,y)[p_1,q_1,\ldots,p_{m\ell},q_{m\ell}]$ are $\mathbf{\Sigma}_0^B$ formulas, where $\vec{p},\vec{q}$ are the Boolean variables. Then $\mathbf{VNC}^1$ proves the existences of $Z_1,\ldots,Z_m$ such that*

$$\forall z < c \forall x < a$$
$$\bigwedge_{i=1}^{m} [(Z_i^{[z]}(x+a) \leftrightarrow \psi_i(z,x)) \wedge 0 < x \supset (Z_i^{[z]}(x) \leftrightarrow \varphi_i(z,x)[\ldots]))]$$

where $[\ldots]$ is the list (of $2m\ell$ Boolean variables):

$$Z_1^{[z]}(2x), Z_1^{[z]}(2x+1), \ldots, Z_m^{[z+\ell-1]}(2x), Z_m^{[z+\ell-1]}(2x+1)$$

($Z_i^{[z]}(y)$ implicitly is $\perp$ if $z \ge c$).

Proof. Using Theorem 4.4 above, the idea is to construct a constant k, a number a' and $\mathbf{\Sigma}_0^B$ formulas $\psi'(c,x)$ and $\varphi'(c,x)[p_0,\ldots,p_{k-1}]$ so that from the set Y that satisfies (23) (for k, a', ψ' and φ') we can obtain $Z_1,\ldots,Z_m$.

Consider for example $m=2, \ell=2$. W.l.o.g., assume that $c \ge 1$. Consider the (overlapping) subtrees

$$Z_1^{[0]}, Z_2^{[0]}, \ldots, Z_1^{[c-1]}, Z_2^{[c-1]} \tag{24}$$

The branching factor of these trees are 8 (i.e., $2m\ell$). So let $k=8$ (i.e., $k=2m\ell$). We will construct Y (with branching factor 8) so that the disjoint subtrees rooted at

$$Y(c), \ldots, Y(3c-1) \tag{25}$$

are exactly the subtrees listed in (24).

Thus we can define an 1-1, into map

$$s : \{1,2\} \times \mathbb{N}^2 \to \mathbb{N}$$

so that

$$Z_i^{[z]}(x) \leftrightarrow Y(s(i,z,x))$$

The map s must be defined in such a way that the nodes of the trees listed in (24) match with those whose roots are listed in (25). For example, for the root level we need

$$s(1,0,1) = c,\ s(2,0,1) = c+1,\ s(1,1,1) = c+2,\ s(2,1,1) = c+3,\ \ldots$$

For other levels we need: If $s(i,z,x) = y$, then

$$s(1,z,2x) = 8y,\ s(1,z,2x+1) = 8y+1,\ \ldots,\ s(2,z+1,2x+1) = 8y+7$$

It turns out to be easier to define partial, onto maps $f, g : \mathbb{N} \to \mathbb{N}$ and $h : \mathbb{N} \to \{1, 2\}$ so that

$$s(h(y), g(y), f(y)) = y$$

In other words,

$$Y(y) \leftrightarrow Z_{h(y)}^{[g(y)]}(f(y))$$

For example, for $0 \leq z < 2c$:

$$f(c+z) = 1, \qquad g(c+z) = \lfloor z/2 \rfloor, \qquad h(c+z) = 1 + (z \mod 2)$$

In general, we need to define f, g, h only for values of x of the form $8^r c + z$ for $0 \leq z < 2 \cdot 8^r c$. The definitions of f, g, h at $8^r c + z$ are straightforward using the base 8 notation for z, where $0 \leq z < 2 \cdot 8^r c$.

Once f, g, h are defined, the formula ψ' and φ' are defined by

$$\psi'(c,x) \leftrightarrow \psi_{h(x)}(g(x), f(x)) \qquad \text{and} \qquad \varphi'(c,x)[\ldots] \leftrightarrow \varphi_{h(x)}(g(x), f(x))[\ldots]$$

(where $\ldots$ is the list of $2m\ell$ Boolean variables). □

4.1.3 Proof of Theorem 4.1

Proof. We tacitly assume that $\mathcal{L}_n$ contains *Row* and *seq* for $n \geq 0$. (It is easy to see that $\mathbf{VNC}^1(Row, seq)$ is a conservative extension of $\mathbf{VNC}^1$.) We prove by induction on n. The base case where $n = 0$ is trivial. For the induction step, assume that the theorem holds for n.

The case where the new function in $\mathcal{L}_{n+1}$ is of the form $F_{\varphi,t}$ or $f_{\varphi,t}$ follows from Theorem 4.3 and the easy fact that $F^{\star}_{\varphi,t}$ (or $f^{\star}_{\varphi,t}$) is also definable in $\mathbf{VNC}^1(\mathcal{L}_n)$. As we mentioned at the beginning of section 4.1, the defining equation (12) (resp. (13)) is provable in $\mathbf{VNC}^1(\mathcal{L}_n)$.

Now suppose that the new function f in $\mathcal{L}_{n+1}$ is of the form $f_{g,h}$ where $f, g \in \mathcal{L}_n$. The value of $f(y)$ can be seen as the composition of the series

$$f(0), \ldots, f(y-1)$$

using the "rules" $h(z,u)$ for $z < y, u \leq 4$. The composition of this series can be computed by constructing a binary tree whose leaves correspond to $f(0), \ldots, f(y)$: The inner node that is the root of the subtree with leaves

$$f(i), \ldots, f(j) \tag{26}$$

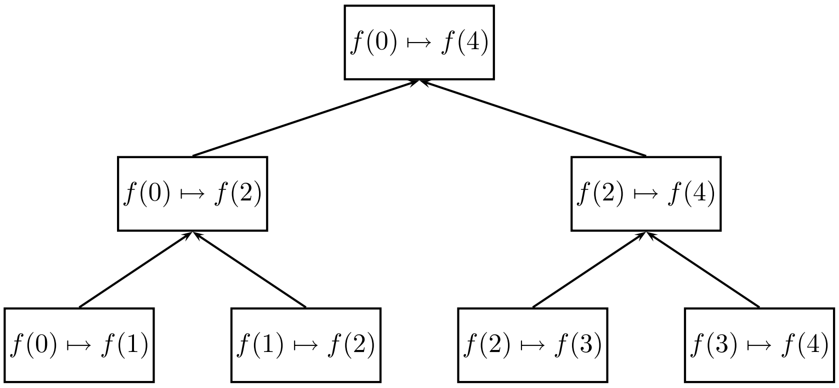

Figure 4: Computing f by a binary tree.

computes $f(j)$ from $f(i)$. (See Figure 4.)

The construction of the tree is straightforward although tedious. Note that the description of the nodes of the tree involves g and h, so to make sure that the formula describing the nodes are $\mathbf{\Sigma}_0^B$ formula (over $\mathcal{L}_A^2$), we first need to "gather" the values of g and h in some arrays using $\mathbf{\Sigma}_0^B(\mathcal{L}_n)$-**COMP**. Then the existence of such tree in $\mathbf{VNC}^1$ follows from Theorem 4.5. Details are left to the reader.

In order to apply Theorem 4.3, we need to show that $f^\star$ is also definable in $\mathbf{VNC}^1(\mathcal{L}_n)$. Thus we need polynomially many trees similar to the binary we used for computing f above. Again, the existence of such "large" tree is provable in $\mathbf{VNC}^1$ using Theorem 4.5.

Finally, the defining equations (14) and (15) of $f_{g,h}$ are provable in $\mathbf{VNC}^1(\mathcal{L}_n)$ using property 26 above. $\square$

The same proof shows that if **VALV** were defined using k-BNR for all $k \in \mathbb{N}$ (instead of 4-BNR), then it would still be a conservative extension of $\mathbf{VNC}^1$.

4.2 VALV Extends $\mathbf{VNC}^1$

By Theorem 3.4 we have in particular that **VALV** proves $\mathbf{\Sigma}_0^B$-**COMP**, and therefore **VALV** extends $\mathbf{V}^0$. To show that **VALV** extends $\mathbf{VNC}^1$ it remains to show that **VALV** proves *MFVP*. To prove the existence of Y in *MFVP*, we formalize Barrington's proof [Bar89] that the Balanced Formula Value problem can be reduced to the word problem for S_5. This reduction shows how to compute the values of the gates in a Boolean circuit whose underlying graph is a balanced binary tree. Once this has been done, the string Y can be obtained by $\mathbf{\Sigma}_0^B$-**COMP**.

First we outline the reduction.

4.2.1 Outline of the Reduction to the Word Problem for S_5

Consider a tree-like circuit T of depth $\log(a)$, with inputs $I(0), \ldots, I(a-1)$. The goal is to (uniformly) construct for each gate x in T a sequence P_x of permutations in S_5:

$$P_x = p_{x,1}, \ldots, p_{x,k}$$

where k depends on x (see below), so that

$$\circ P_x = p_{x,1} \circ \ldots \circ p_{x,k} = e \qquad \text{iff} \qquad T(x) = 0$$

(e is the identity of S_5, and $\circ$ is the composition operator). Here P_x has length $k = k(x) = 4^h$, where h is the height (i.e., longest distance to a leaf) of the gate x in T.

We use the fact that S_5 contains a nontrivial commutator (σ in Notation below):

Notation Let $\sigma_1 = (12345), \sigma_2 = (13542)$ and $\sigma = \sigma_1^{-1} \circ \sigma_2^{-1} \circ \sigma_1 \circ \sigma_2 = (12534)$. Also, let e be the identity in S_5.

Invariance We will construct P_x so that

$$p_{x,1} \circ \ldots \circ p_{x,k} = \begin{cases} \sigma & \text{if } T(x) = 1 \\ e & \text{if } T(x) = 0 \end{cases} \tag{27}$$

The sequence P_x are defined inductively based on the height of gate x. We consider the following cases:

Case I: Gate x of T is an input gate. Then $k(x) = 1$, and

$$p_{x,1} = \begin{cases} \sigma & \text{if } I(x) = 1 \\ e & \text{if } I(x) = 0 \end{cases}$$

Case II: Gate x is an $\wedge$-gate with inputs from gates y, z. The idea is to obtain

$$P'_y, P'_z, P''_y, P''_z$$

such that

$$\circ P'_y = \begin{cases} \sigma_1 & \text{if } T(y) = 1 \\ e & \text{if } T(y) = 0 \end{cases} \qquad \circ\, P''_y = \begin{cases} \sigma_1^{-1} & \text{if } T(y) = 1 \\ e & \text{if } T(y) = 0 \end{cases}$$

and

$$\circ P'_z = \begin{cases} \sigma_2 & \text{if } T(z) = 1 \\ e & \text{if } T(z) = 0 \end{cases} \qquad \circ\, P''_z = \begin{cases} \sigma_2^{-1} & \text{if } T(z) = 1 \\ e & \text{if } T(z) = 0 \end{cases}$$

Then $P_x = P'_y, P'_z, P''_y, P''_z$ satisfies the requirement (27).

Now, P'_y, P''_y and P'_z, P''_z are obtained from P_y and P_z as follows:

$$p'_{y,i} = \theta_1 \circ p_{y,i} \circ \theta_1^{-1}, \qquad p''_{y,i} = \eta_1 \circ p_{y,i} \circ \eta_1^{-1} \qquad (1 \le i \le k(y)) \tag{28}$$
$$p'_{z,i} = \theta_2 \circ p_{z,i} \circ \theta_2^{-1}, \qquad p''_{z,i} = \eta_2 \circ p_{z,i} \circ \eta_2^{-1} \qquad (1 \le i \le k(z)), \tag{29}$$

where:

Notation $\theta_1 = (14532), \theta_2 = (13425), \eta_1 = (13254), \eta_2 = (12543)$.

Note that θ_i, η_i satisfy:

$$\theta_i \circ \sigma \circ \theta_i^{-1} = \sigma_i, \qquad \eta_i \circ \sigma \circ \eta_i^{-1} = \sigma_i^{-1}$$

Case III: Gate x of T is an $\vee$-gate with inputs from gates y and z. Essentially, this case reduces to the previous case using the identity:

$$A \vee B \Leftrightarrow \neg(\neg A \wedge \neg B)$$

Thus we will construct sequences Q'_y, Q''_y and Q'_z, Q''_z so that the sequence

$$Q = Q'_y, Q''_y, Q'_z, Q''_z$$

satisfies

$$\circ Q = \begin{cases} e & \text{if } T(x) = 1 \\ \sigma^{-1} & \text{if } T(x) = 0 \end{cases}$$

Then the sequence P_x is defined to be the same as Q except for the last permutation q is replaced by $q \circ \sigma$. It is easy to verify that P satisfies (27).

More precisely, the sequences Q'_y, Q''_y and Q'_z, Q''_z are constructed so that

$$\circ Q'_y = \begin{cases} e & \text{if } T(y) = 1 \\ \sigma_2 & \text{if } T(y) = 0 \end{cases} \qquad \circ\, Q''_y = \begin{cases} e & \text{if } T(y) = 1 \\ \sigma_2^{-1} & \text{if } T(y) = 0 \end{cases}$$

and

$$\circ Q'_z = \begin{cases} e & \text{if } T(z) = 1 \\ \sigma_1 & \text{if } T(z) = 0 \end{cases} \qquad \circ\, Q''_z = \begin{cases} e & \text{if } T(z) = 1 \\ \sigma_1^{-1} & \text{if } T(z) = 0 \end{cases}$$

The elements of Q'_y, Q''_y, Q'_z, Q''_z are defined as follows:

$$q'_{y,i} = \eta_2 \circ p_{y,i} \circ \eta_2^{-1}, \qquad q''_{y,i} = \theta_2 \circ p_{y,i} \circ \theta_2^{-1} \qquad (1 \le i < k(y)) \tag{30}$$
$$q'_{z,i} = \eta_1 \circ p_{z,i} \circ \eta_1^{-1}, \qquad q''_{z,i} = \theta_1 \circ p_{z,i} \circ \theta_1^{-1} \qquad (1 \le i < k(z)) \tag{31}$$
$$q'_{y,k(y)} = \eta_2 \circ p_{y,k(y)} \circ \eta_2^{-1} \circ \sigma_2, \qquad q''_{y,k(y)} = \theta_2 \circ p_{y,k(y)} \circ \theta_2^{-1} \circ \sigma_2^{-1} \tag{32}$$
$$q'_{z,k(z)} = \eta_1 \circ p_{z,k(z)} \circ \eta_1^{-1} \circ \sigma_1, \qquad q''_{z,k(z)} = \theta_1 \circ p_{z,k(z)} \circ \theta_1^{-1} \circ \sigma_1^{-1} \tag{33}$$

4.2.2 Nonsolvability of S_5

Before formalizing the above reduction, we analyze how the fact that S_5 is nonsolvable is used. (In general, Barrington shows that the word problem for any nonsolvable group is complete for $\mathbf{NC}^1$ by a slightly modified reduction.)

What is needed in the reduction is the existence of elements σ_1, σ_2 of the group S_5 so that they are both conjugates of their commutator σ. We will show that any group G that contains two such elements σ_1, σ_2 must be nonsolvable. (It follows that S_5 is nonsolvable.)

Lemma 4.6. *Suppose that G is a group that contains two elements σ_1, σ_2 with the property that σ_i is a conjugate of $\sigma = \sigma_1^{-1} \circ \sigma_2^{-1} \circ \sigma_1 \circ \sigma_2$, for $i = 1, 2$. Then G is nonsolvable.*

Proof. Let $H = \langle \sigma_1, \sigma_2 \rangle$ (the group generated by σ_1 and σ_2). We show that H is a nonsolvable group. It follows that G is nonsolvable, since G contains a nonsolvable subgroup.

Let K be the commutator subgroup of H. Then consider the quotient map $q : H \to H/K$. Since $\sigma \in K$, $q(\sigma) = 1$. Also, since σ_i are conjugates of σ, $q(\sigma_i) = 1$, for $i = 1, 2$. Thus $H = K$, and hence H is nonsolvable. □

4.2.3 Formalization of the Reduction

We will use the fact that a number of functions and relations are definable in $\mathbf{I\Delta}_0$, and thus in $\mathbf{V}^0$. For simplicity, assume $a = 2^d$ for $d \geq 1$. Now consider a gate x of height $h \geq 0$, then $2^{d-h} \leq x < 2^{d-h+1}$, and the S_5-word P_x has length 4^h. We will show how to compute the i-th permutation $p_{x,i}$ in P_x, for $0 \leq i < 4^h$.

Write i in base 4: $i = i_{h-1} \dots i_0$, where $0 \leq i_r \leq 3$. When the gate x is an $\wedge$-gate (resp. $\vee$-gate), i_{h-1} will state which of the "quarters" P'_y, P'_z, P''_y, P''_z (resp. Q'_y, Q'_z, Q''_y, Q''_z) that $p_{x,i}$ comes from. Here $y = 2x, z = 2x + 1$: outputs of gates y, z are connected to inputs of gate x.

More precisely, consider the case where $G(x)$ is an $\wedge$-gate. If $i < 4^{h-1}$ (i.e, $i_{h-1} = 0$), then

$$p_{x,i} = \theta_1 p_{y,i'} \theta_1^{-1}$$

(see (28)), where $i' = i_{h-2} \dots i_0$ (base 4).

Similarly for the cases where $4^{h-1} \leq i < 2 \times 4^{h-1}$ (i.e. $i_{h-1} = 1$), etc. In general, $p_{x,i}$ is defined from $p_{2x+(i_{h-1} \mod 2), i'}$ using (28)–(29) and (30)–(33).

The sequence of permutations $p_{x,i}$ (for $0 \leq i < 4^h$) can be seen as being obtained after h stages: In each stage we have a sequence of length 4^h, and the i-th element of the sequence in stage $\ell + 1$ is defined from the i-th element of the previous sequence. Thus we will define the permutations $p_{x,i} : \{0, 1, 2, 3, 4\} \mapsto \{0, 1, 2, 3, 4\}$ as

$$p_{x,i}(u) = f(h, x, i, u)$$

where h is the height of gate x, and f is defined by Bounded Number Recursion on h. For readability we will write $f(h, x, i, \cdot)$ to indicate that f is treated as a permutation.

First,

$$f(0,x,i,\cdot) = \begin{cases} \sigma & \text{if } I(x'-a)=1 \\ e & \text{otherwise} \end{cases}$$

where $x' = 2^h x + i'$, and i' has the binary representation

$$(i_{h-1} \mod 2)(i_{h-2} \mod 2)\dots(i_0 \mod 2)$$

Next, for $1 \leq \ell \leq h$, $f(\ell,x,i,\cdot)$ is defined from $f(\ell-1,x,i,\cdot)$ by cases, depending on the type of gate $G(x')$ in T, where now $x' = (2^{h-\ell}x + i')$, and i' is the number with binary representation

$$(i_{h-1} \mod 2)\dots(i_\ell \mod 2)$$

(when $\ell = h$, $i' = 0$).

For example, suppose that gate x' is an $\wedge$-gate. Then

$$f(\ell,x,i,\cdot) = \begin{cases} \theta_1 \circ f(\ell-1,x,i,\cdot) \circ \theta_1^{-1} & \text{if } i_{\ell-1}=0 \\ \theta_2 \circ f(\ell-1,x,i,\cdot) \circ \theta_2^{-1} & \text{if } i_{\ell-1}=1 \\ \eta_1 \circ f(\ell-1,x,i,\cdot) \circ \eta_1^{-1} & \text{if } i_{\ell-1}=2 \\ \eta_1 \circ f(\ell-1,x,i,\cdot) \circ \eta_1^{-1} & \text{if } i_{\ell-1}=3 \end{cases}$$

(Since $\theta_1, \theta_2, \eta_1, \eta_2$ are 5-permutations, it is clear that f is defined using 4-Bounded Number Recursion.)

Finally, the value of gate x in T is determined by the composition

$$f(h,x,0,\cdot) \circ f(h,x,1,\cdot) \circ \dots \circ f(h,x,4^h-1,\cdot)$$

which can be computed using 4-Bounded Number Recursion: Define $g(h,x,i,k,\cdot)$ using 4-Bounded Number Recursion from f as follows:

$$g(h,x,i,0,\cdot) = f(h,x,i,\cdot)$$
$$g(h,x,i,k+1,\cdot) = g(h,x,i,k,\cdot) \circ f(h,x,i+k+1,\cdot)$$

Then

$$g(h,x,i,j,\cdot) = f(h,x,i,\cdot) \circ \dots \circ f(h,x,i+j,\cdot)$$

Hence,

$$f(h,x,0,\cdot) \circ f(h,x,1,\cdot) \circ \dots \circ f(h,x,4^h-1,\cdot) = g(h,x,0,4^h-1,\cdot)$$

As a result, $T(x)$ is 1 if and only if

$$g(h,x,0,4^h-1,\cdot) = \sigma$$

Our definitions of f,g above show that they are in $\mathcal{L}_{\mathbf{FNC}^1}$.

4.2.4 Proving the Correctness of the Reduction in VALV

We will show that **VALV** proves the correctness of the reduction carried in the previous subsection in the sense that if Y encodes an evaluation of the gates in the circuit (a, G) given input I (i.e., Y satisfies $\delta_{MFVP}(a, G, I, Y)$, see (5)), then $Y(x)$ holds iff the $g(h, x, 0, 4^h - 1, \cdot) = \sigma$. (Recall σ from (27).)

Lemma 4.7 (Provable in **VALV**)**.** *Suppose that Y satisfies $\delta_{MFVP}(a, G, I, Y)$. Then for $x < 2a$, $Y(x) \leftrightarrow g(h, x, 0, 4^h - 1, \cdot) = \sigma$.*

Proof. Intuitively we show that the sequences obtained in h stages (in Subsection 4.2.3) are "correct". The proof is by induction on ℓ that the sequences constructed in stage ℓ work as expected. Thus, consider a segment (of length 4^ℓ) of the sequence obtained in stage ℓ:

$$f(\ell, x, i4^\ell, \cdot),\ f(\ell, x, i4^\ell + 1, \cdot),\ \ldots,\ f(\ell, (i+1)4^\ell - 1, \cdot)$$

whose composition is $g(\ell, x, i4^\ell, 4^\ell - 1, \cdot)$. Here $0 \leq i < 4^{h-\ell}$.

Formally we will prove by induction on ℓ that for all $i < 4^{h-\ell}$,

$$g(\ell, x, i4^\ell, 4^\ell - 1, \cdot) = \begin{cases} \sigma & \text{if } Y(x') = 1 \\ e & \text{otherwise} \end{cases} \tag{34}$$

where $x' = 2^{h-\ell}x + i'$ and i' is the number with binary representation

$$(i_{h-\ell-1} \mod 2) \ldots (i_0 \mod 2)$$

Here $i = i_{h-\ell-1} \ldots i_0$ base 4.

The base case is obvious from the definition of f and Y.

For the induction step, suppose that (34) holds for $(\ell - 1)$, where $1 \leq \ell \leq h$. We prove (34) for ℓ.

Let $i < 4^{h-\ell}$, and x', i' as above. Consider for example the case where gate x' is an $\wedge$-gate. We need to verify that

$$\begin{gathered} g(\ell, x, i4^\ell, 4^{\ell-1} - 1, \cdot), \\ g(\ell, x, i4^\ell + 4^{\ell-1}, 4^{\ell-1} - 1, \cdot), \\ g(\ell, x, i4^\ell + 2 \times 4^{\ell-1}, 4^{\ell-1} - 1, \cdot), \\ g(\ell, x, i4^\ell + 3 \times 4^{\ell-1}, 4^{\ell-1} - 1, \cdot) \end{gathered}$$

respectively compute the compositions of

$$P'_y, P'_z, P''_y, P''_z$$

as in **Case II** in Subsection 4.2.1.

The verification can be done, for example, by proving by induction on $j < 4^{\ell-1} - 1$ that

$$g(\ell, x, i4^\ell, j, \cdot) = \theta_1 \circ g(\ell, x, i4^\ell, j, \cdot) \circ \theta_1^{-1}$$

Details are left to the reader. ☐

As a corollary, **VALV** proves the axiom *MFVP* (5), because a Y that satisfies $\delta_{MFVP}(a, G, I, Y)$ can be defined using $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$-**COMP**:

$$\forall x < 2aY(x) \leftrightarrow g(h, x, 0, 4^h - 1, \cdot) = \sigma$$

4.3 Proof of the Main Theorem

Proof. We noted earlier (Section 4.2) that **VALV** extends $\mathbf{V}^0$. Lemma 4.7 above can be used to show that $\mathbf{VALV} \vdash \mathit{MFVP}$: Given (a, G, I). Define Y by $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$-**COMP** as follows:

$$|Y| \leq 2a \wedge \forall x < 2a,\ Y(x) \leftrightarrow g(h, x, 0, 4^h - 1, \cdot) = \sigma$$

Lemma 4.7 shows that such Y satisfies $\delta_{MFVP}(a, G, I, Y)$.

Finally, Theorem 4.1 implies that **VALV** is conservative over $\mathbf{VNC}^1$, because $\mathbf{VNC}^1(\mathcal{L}_n)$ (see Theorem 4.1) is conservative over $\mathbf{VNC}^1$, for all n. $\square$

5 Conclusion

Although we did not discuss the theory **ALV** [Clo90a] in details, it can be shown by similar arguments that the quantified version of **ALV** is RSUV isomorphic to the theory $\overline{\mathbf{VNC}}^1$ in [CN06]. The latter is a universal conservative extension of $\mathbf{VNC}^1$. As a result, the equivalence between **ALV** and the other theories should be clear. Therefore we obtain the following result not mentioned in [CT95]:

Corollary 5.1. *The theories* **ALV** *and* **ALV**$'$ *are equivalent, and their quantified versions both are conservative extensions of* $\mathbf{T}^0\mathbf{NC}^0$.

In general, for each $k \geq 2$ a theory $\mathbf{VNC}^k$ is defined in [NC05, CN06]. When $k = 1$ this general definition gives a theory that characterizes U_D-uniform $\mathbf{NC}^1$ (i.e., uniform $\mathbf{NC}^1$ where uniformity is defined using *direct connection language* [Ruz81]). In particular, define $\mathbf{VNC}^1_D$ to be the theory over $\mathcal{L}_A^2 \cup \{\log\}$ that is axiomatized by $\mathbf{V}^0$ and the following axiom:

$$\forall a \forall E \forall G \forall I (\mathit{Fanin2}(a, d, E) \supset \exists Y\, \delta_{MCVP}(a, \log a, E, G, I, Y)) \qquad (35)$$

The axiom formalizes a polytime evaluation of an U_D-uniform Boolean circuit with fan-in 2 and logarithmic depth: Here $\mathit{Fanin2}(a, d, E)$ is the formula stating that the underlying graph encoded by (a, d, E) has indegree at most 2:

$$\mathit{Fanin2}(a, d, E) \equiv \forall z < d \forall x < a \exists u_1, u_2 < a \forall v < a,\ E(z, v, x) \supset v = u_1 \vee v = u_2$$

and $\delta_{MCVP}(g, d, E, G, I, Y)$ states that Y evaluates the circuits with a input gates and depth d encoded by (g, d, E, G) given inputs encoded by I:

$$\begin{aligned}\delta_{MCVP}(g, d, E, G, I, Y) \ \equiv\ & \forall x < g \forall z < d,\ (Y(0, x) \leftrightarrow I(x)) \wedge \\ & [Y(z+1, x) \leftrightarrow (G(z+1, x) \wedge \forall u < g,\ E(z, u, x) \supset Y(z, u)) \vee \\ & \qquad (\neg G(z+1, x) \wedge \exists u < g,\ E(z, u, x) \wedge Y(z, u))]\end{aligned}$$

Then it is easy to see that $\mathbf{VNC}^1 \subseteq \mathbf{VNC}^1_D$. However it is an open problem whether $\mathbf{VNC}^1_D$ is a conservative extension of $\mathbf{VNC}^1$.

A related question is whether the two "minimal" theories that characterize **L** and **SL** [Kol04, Ngu05] are equivalent, given the fact recently proved that $\mathbf{L} = \mathbf{SL}$ [Rei05].

Using the Bounded Number Recursion we obtain recursion-theoretic characterizations for several other small classes, such as $\mathbf{FTC}^0$, $\mathbf{FAC}^0(2)$ and $\mathbf{FAC}^0(6)$ and **FL**. (The characterizations of $\mathbf{FAC}^0(2)$ and $\mathbf{FAC}^0(6)$ are two-sorted version of Clote-Takeuti results [CT95], and the characterization of **FL** is the two-sorted version of Lind's characterization of **FL**.) These are discussed in the author's upcoming PhD thesis.

6 Acknowledgment

I would like to thank Steve Cook for carefully reading a draft of this paper and giving numerous invaluable comments and suggestions. I would like to thank Eric Allender for prompt answer on U_D-uniform $\mathbf{NC}^1$. Thanks also to Ho Minh Toan for our discussions on abstract algebra, and the referee for helpful comments.

References

[Ara00] Toshiyasu Arai. Bounded arithmetic AID for Frege system. *Annals of Pure and Applied Logic*, 103:155–199, 2000.

[Bar89] David A. Barrington. Bounded-Width Polynomial-Size Branching Programs Recognizes Exactly Those Languages in $\mathbf{NC}^1$. *Journal of Computer and System Sciences*, 38:150–164, 1989.

[BIS90] David A. Mix Barrington, Neil Immerman, and Howard Straubing. On Uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306, 1990.

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

[Bus87] Samuel Buss. The Boolean formula value problem is in ALOGTIME. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 123–131, 1987.

[Clo90a] Peter Clote. Alogtime and a Conjecture of S. A. Cook. In *Proceedings of IEEE Symposium on Logic in Computer Science*, 1990.

[Clo90b] Peter Clote. Sequential, Machine-Independent Characterizations of the Parallel Complexity Classes AlogTIME, AC^k, NC^k and NC. In S. R. Buss and P.J. Scott, editors, *Feasible Mathematics*. Birkhauser, 1990.

[Clo93] Peter Clote. On Polynomial Size Frege Proofs of Certain Combinatorial Principles. In Peter Clote and Jan Krajíček, editors, *Arithmetic, Proof Theory, and Computational Complexity.* Oxford, 1993.

[CM05] Stephen Cook and Tsuyoshi Morioka. Quantified Propositional Calculus and a Second-Order Theory for NC^1. *Archive for Mathematical Logic*, 44:711–749, 2005.

[CN06] Stephen Cook and Phuong Nguyen. Foundations of Proof Complexity: Bounded Arithmetic and Propositional Translations. Book in progress, 2006.

[Coo98] Stephen Cook. Relating the Provable Collapse of P to NC1 and the Power of Logical Theories. *DIMACS Series in Discrete Math. and Theoretical Computer Science*, 39, 1998.

[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.

[CT92] Peter Clote and Gaisi Takeuti. Bounded Arithmetic for NC, ALOGTIME, L and NL. *Annals of Pure and Applied Logic*, 56:73–117, 1992.

[CT95] Peter Clote and Gaisi Takeuti. First Order Bounded Arithmetic and Small Boolean Circuit Complexity Classes. In P. Clote and J. B. Remmel, editors, *Feasible Mathematics II.* Birkhäuser, 1995.

[HP93] Petr Hájek and Pavel Pudlák. *Metamathematics of First-Order Arithmetic.* Springer–Verlag, 1993.

[Imm99] Neil Immerman. *Descriptive Complexity.* Springer, 1999.

[Kol04] Antonina Kolokolova. *Systems of Bounded Arithmetic from Descriptive Complexity.* PhD thesis, University of Toronto, 2004.

[Kra90] Jan Krajíček. Exponentiation and second-order bounded arithmetic. *Annals of Pure and Applied Logic*, 48:261–276, 1990.

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

[NC05] Phuong Nguyen and Stephen Cook. Theory for $\mathbf{TC}^0$ and Other Small Complexity Classes. *Logical Methods in Computer Science*, 2, 2005.

[Ngu04] Phuong Nguyen. VTC^0: A Second-Order Theory for TC^0. Master's thesis, University of Toronto, 2004. `http://www.cs.toronto.edu/~ntp/`.

[Ngu05] Phuong Nguyen. Two–Sorted Theories for **L, SL, NL** and **P** Based on Graph Accessibility Problems. ECCC Report TR05-017, January, 2005.

[Pit00] Francois Pitt. *A Quantifier-Free String Theory ALOGTIME Reasoning.* PhD thesis, University of Toronto, 2000.

[Raz93] Alexander A. Razborov. An Equivalence between Second Order Bounded Domain Bounded Arithmetic and First Order Bounded Arithmetic. In Peter Clote and Jan Krajíček, editors, *Arithmetic, Proof Theory and Computational Complexity*, pages 247–277. Oxford, 1993.

[Rei05] Omer Reingold. Undirected ST-Connectivity in Log-Space. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 376–385, 2005.

[Ruz81] Walter Ruzzo. On Uniform Circuit Complexity. *Journal of Computer and System Sciences*, 22:365–383, 1981.

[Tak93] Gaisi Takeuti. RSUV Isomorphism. In Peter Clote and Jan Krajíček, editors, *Arithmetic, Proof Theory and Computational Complexity*, pages 364–386. Oxford, 1993.

[Zam96] Domenico Zambella. Notes on Polynomially Bounded Arithmetic. *Journal of Symbolic Logic*, 61(3):942–966, 1996.