Functional Interpretations of Feasibly Constructive Arithmetic[1]

Stephen Cook[2]

Alasdair Urquhart[3]

Revised March, 1993

Abstract

A notion of feasible function of finite type based on the typed lambda calculus is introduced which generalizes the familiar type 1 polynomial time functions. An intuitionistic theory IPV^{ω} is presented for reasoning about these functions. Interpretations for IPV^{ω} are developed both in the style of Kreisel's modified realizability and Gödel's *Dialectica* interpretation. Applications include alternative proofs for Buss's results concerning the classical first order system S^1_2 and its intuitionistic counterpart IS^1_2, as well as proofs of some of Buss's conjectures concerning IS^1_2, and a proof that IS^1_2 cannot prove that extended Frege systems are not polynomially bounded.

[1]Research supported by the Natural Sciences and Engineering Research Council of Canada.

[2]Department of Computer Science, University of Toronto, Toronto, Canada. e-mail address: sacook@theory.toronto.edu.

[3]Department of Philosophy, University of Toronto, Toronto, Canada. e-mail address: urquhart@theory.toronto.edu.

Contents

0 Introduction

0.1 Setting

In 1975 Cook [11] introduced an equational system PV ("polynomially verifiable") of number theory, patterned after Skolem's equational theory of primitive recursive arithmetic. Whereas Skolem's system has a function symbol for each primitive recursive function, PV has one for each polynomial time computable function. The system PV was supposed to capture an intuitive notion of "feasibly constructive proof", a form of highly constructive proof satisfying the equation: Constructive proof is to algorithm as feasibly constructive proof is to polynomial time algorithm.

A major purpose of the PV paper was to relate propositional calculus proof lengths to feasibly constructive proofs. For example, every true equation in PV gives rise to a family of propositional tautologies which assert the equation, and if the equation is a theorem of PV then each tautology in the family has an extended Frege proof whose length is bounded by a polynomial in the length of the tautology. It was shown that if the soundness of a propositional proof system was provable in PV, then an extended Frege system polynomially simulates the proof system. (See [16] for terminology.) This allowed Martin Dowd [19] to later prove the surprising result that extended Frege systems polynomially simulate Frege systems with substitution.

In 1985 Buss [2] introduced a system S_2^1 of arithmetic based on classical first order predicate calculus and proved that the definable functions in his system are precisely the polynomial time computable functions. Buss developed S_2^1 without knowing about PV, but later incorporated a proof in [2] that $S_2^1(PV)$, that is S_2^1 augmented by the notation and theorems of PV, is a conservative extension of PV. In a later paper [4] Buss developed an intuitionistic version IS_2^1 of S_2^1 and used a complicated polynomial time variant of Kleene realizability to prove that for every theorem of the form $\forall x \exists y B(x,y)$ there is a polynomial time computable function f such that $B(n, f(n))$ holds for all n.

It seemed to the present authors that IS_2^1 was a good candidate for formalizing the notion of feasibly constructive proof for propositions expressed in first order arithmetic. This impression has been reinforced by our experience with the system and serves as motivation for the present paper. In particular we give a more natural set of axioms for IS_2^1 and give a more direct proof of Buss's theorem about the system. Buss takes as axioms for IS_2^1 all theorems of the classical system S_2^1 of a certain form $(H\Sigma_1^b \to H\Sigma_1^b)$, and his realizability proof relies heavily on the main result of [2] (which is proved by a cut elimination argument). In contrast, we formulate IS_2^1 directly (in the style of Buss's S_2^1), axiomatizing it with a list of open formulas and an induction axiom scheme. We then prove Buss's result for IS_2^1 directly, using a variant of Kreisel's modified realizability, and avoid use of cut elimination. We proceed to establish a conjecture in [4] by showing that the formula $B(n, f(n))$ mentioned above is not only true, but provable in IS_2^1 extended by a definition for f. The last argument does use cut elimination.

Finally we adapt Gödel's *Dialectica* interpretation [20] for Heyting arithmetic to IS^1_2. As applications, we reprove Buss's result that $S^1_2(PV)$ is a conservative extension of PV, and we prove that the theorems of S^1_2 which Buss chose as axioms for IS^1_2 are also theorems in our formulation of IS^1_2, and hence the two formulations are equivalent. A further application shows that, in a natural sense, IS^1_2 cannot prove that extended Frege systems are not polynomially bounded.

For our versions of both modified realizability and the *Dialectica* interpretation we introduce a system of feasible functions of finite type based on the typed lambda calculus, which is of independent interest in complexity theory.

0.2 Feasibly Constructive Proofs

One way to explain the notion of feasibly constructive proof (which we abbreviate as "feasible proof") is to adapt a standard explanation of constructive proof by injecting the word "feasible" wherever appropriate. Thus the logical connectives $\wedge$, $\vee$, $\neg$, and $\exists$ are explained as follows: A feasible proof of $A \wedge B$ is a feasible proof of A together with a feasible proof of B. A feasible proof of $A \vee B$ is a feasible proof of A or a feasible proof of B (it must be clear which). $\neg A$ is $A \to 0 = 1$. A feasible proof of $\exists x A(x)$ is an object n together with a feasible proof of $A(n)$

A feasible proof of $\forall x A(x)$ must give rise to a feasible algorithm for transforming n to a feasible proof of $A(n)$, and the proof must make it plain that the algorithm is correct. This is discussed further below.

Finally, a feasible proof of $A \to B$ gives rise to a feasible algorithm which transforms a feasible proof of A into a feasible proof of B, together with a feasible proof of correctness for the algorithm.

This kind of explanation of $A \to B$ is not universally accepted by constructivists. For example, Bishop [1] argues that Gödel's *Dialectica* interpretation provides a more satisfactory explanation of implication. Fortunately the standard system for constructive first-order arithmetic HA (Heyting Arithmetic) admits both the *Dialectica* interpretation and the standard explanation, as evidenced by Kleene realizability. Fortunately for us, the system IS^1_2 admits both a feasible *Dialectica* interpretation (Section 9) and a feasible form of modified realizability (Section 8). It is not clear whether IS^1_2 admits a feasible form of Kleene realizability. Buss [4] manages to avoid interpreting $\to$ in his polynomial time version of Kleene realizability.

Returning to the universal quantifier, consider the case $\forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n)$ ($\forall \vec{x} A(\vec{x})$ for short) in which $A(\vec{x})$ manifestly defines a polynomial time predicate. Here the major requirement of a proof to make it feasible is that no infeasible concept occurs. For example, consider the instance $n = 4$ of Fermat's Last Theorem: If $xyz \neq 0$ then $x^4 + y^4 \neq z^4$. The standard proof involves a lemma such as $x^2|y^2 \to x|y$. The obvious proof of the lemma uses the unique factorization theorem for $\mathbb{Z}$. Since there is no known polynomial time algorithm for factoring integers, this proof is not feasible. On the other hand a little thought provides a feasible proof of the lemma based on the extended Euclidean algorithm (this is a good exercise for the reader). Thus the case $n = 4$ does indeed have a feasible proof. Pitassi [29] has worked out a feasible proof for the more difficult case $n = 3$.

Assertions of the form $\forall \vec{x} A(\vec{x})$, with $A(\vec{x})$ a polynomial time predicate, comprise the subject matter of the equational system PV [11]. Here a proof boils down to a proof of an identity of the form $f(\vec{x}) = 0$, where f is a polynomial time computable function. According to [11], a feasible proof in this case is *p-verifiable*, in the sense that it provides a polynomial $Q(\vec{n})$ and a uniform method of verifying within $Q(|x_1|, \ldots, |x_n|)$ steps that $f(x_1, \ldots, x_n)$ evaluates to 0. (Of course one must be able to see that the verification always succeeds.) In general, the verification time Q exceeds the evaluation time for f, since the proof of the identity involves introduction of new polynomial time concepts. For example, in the proof of Fermat's Theorem for $n = 4$, the extended Euclidean algorithm is used. All theorems of PV are p-verifiable in this sense, and the converse was conjectured. This leads in [11] to the

0.2.1 Verifiability Thesis A formula of PV is provable in PV iff it is p-verifiable.

Many theorems of graph theory and combinatorics require quantifiers to formulate and are not naturally expressed in PV. Some of these, for example Kuratowski's theorem, Menger's theorem, and Hall's theorem, have feasibly constructive proofs. Others, such as the pigeon hole principle (suitably stated), appear not to. Hence we need a suitable first order theory which extends PV. The system IPV of Section 4 is a conservative extension of both PV and IS_2^1 (Theorem 10.1 and 4.12). This and other considerations lead us to formulate the

0.2.2 Feasible Proof Thesis A statement of first order number theory of the form $\forall \vec{x} \exists \vec{y} A(\vec{x}, \vec{y})$, where $A(\vec{x}, \vec{y})$ is a polynomial time decidable predicate, is a theorem of IPV iff it has a feasibly constructive proof.

As evidence in the forward direction, that IPV theorems have feasible proofs, we point out that theorems of IPV can be realized by feasible functions (of finite type) both in the sense of modified realizability (Section 8) and the *Dialectica* interpretation (Section 9).

The evidence for the converse, that IPV is adequate to formalize feasibly constructive proofs for $\forall\exists$ theorems, comes partly from the robustness of the system. For example, IPV is a conservative extension of IS_2^1 although the two systems are formulated quite differently. Also Buss's formulation of IS_2^1 is formally much more powerful than our formulation, but turns out to be equivalent (10.6). Furthermore IPV is conservative over PV, and the system IPV^ω, the analog of IPV for feasible functions of finite type, is conservative over IPV.

In the preliminary version [9] of this paper we formulated 0.2.2 more generally for all statements of first order number theory, rather than just for $\forall\exists$ statements. While we still have no difficulty with the forward direction of this more general statement (that IPV theorems have feasible proofs), we now have reservations about the converse for propositions with a complicated quantifier structure. An example is the induction axiom scheme $PIND$ (see §1). As presently formulated for the systems IS_2^1 and IPV, the induction formula $A(x)$ must explicitly represent an NP predicate. However Buss [5] has formulated an extension IPV^+ of IPV in which the formula has the more general form $A(x) \vee B$, where $A(x)$ is as above and B is an arbitrary formula not containing x. It is difficult philosophically to argue that the more general induction scheme is not feasible, and in fact Buss finds a pleasing model theoretic characterization of IPV^+. It is an open question whether IPV^+ is a proper extension of IPV.

Another apparently innocuous extension of the NP $PIND$ scheme is to allow $\neg\neg A(x)$ as an induction formula, where $A(x)$ is an NP formula. We do not know whether this extension is proper. In fact, if we allowed this more general scheme in the formulation of IPV, then results of Leivant [27] could be used in place of the *Dialectica* interpretation to prove Theorem 10.4 and Corollary 10.6.

Although these different formulation of $PIND$ are not obviously equivalent, it turns out that only quantifier-free $PIND$ is needed to prove the $\forall\exists$ theorems of IPV. Furthermore, even using classical logic in place of intuitionistic logic does not increase the $\forall\exists$ theorems of IPV (see Corollary 10.5 and the ensuing discussion). Hence we have confidence in the present formulation of 0.2.2. This also suggests we could strengthen 0.2.2 to state that a $\forall\exists$ formula B can be proved using polynomial time reasoning (whether constructive or not) iff B is a theorem of IPV.

Ultimately our confidence in the adequacy of IPV comes from using it to formalize proofs. In addition to the work in Sections 3 and 4, examples of formalizing proofs occur in Gödel's second incompleteness theorem for PV [11] and S_2^1 [2].

An example which we expect is not provable in IPV is $sat(x) \vee \neg sat(x)$, where $sat(x)$ is "x codes a satisfiable propositional formula". If this is provable in IPV, then $P = NP$ (see 8.15 and succeeding discussion). This motivates trying to show that $sat(x) \vee \neg sat(x)$ is unprovable, since such a result would rule out a feasible proof that $P = NP$, for example by reducing an NP complete problem to linear programming.

A second example which we expect is unprovable in IPV is Fermat's "Little Theorem" (see after 8.6). If this were provable, it would seem to provide a surprising practical method for factoring large integers.

0.3 Other Complexity Classes

The system IPV defines precisely the polynomial time computable functions. Other systems of arithmetic define other complexity classes. (See [7] for a discussion.) The realizing methods of this paper can be adapted to other complexity classes as well; some indication of how to do this occurs in the course notes [14]. In the present paper, we confine our attention to polynomial time computability.

0.4 Outline

In Section 1 we introduce Buss's S_2^1 (slightly reformulated), our version of the system IS_2^1, and compare it with Buss's original IS_2^1 (here called IS_2^1B).

In Section 2 we prove that all polynomial time computable functions are definable in (our) IS_2^1.

In Section 3 and the Appendix we give a detailed treatment of the equational system PV. The formulation here is based on binary notation, in contrast to the original formulation in [11], which uses dyadic notation.

In Section 4 we introduce the system IPV, which has the notation of IS_2^1 augmented by all function symbols in PV. The axioms for IPV are more "rational" than for IS_2^1, consisting essentially of recursive defining equations for each function symbol, and an induction scheme which is simpler than the one for IS_2^1. We show that IPV is a conservative extension of IS_2^1. The system IPV is not only arguably a more pleasing formulation of feasibly constructive arithmetic, it is also technically useful in proving the realizability results in later sections.

In Section 5 we present a standard formulation of the typed lambda calculus.

In Section 6 we introduce the system PV^ω, which is an equational system extending PV to include terms of the typed lambda calculus. The system includes a recursor for introducing higher type functions by a kind of limited recursion on notation. We prove that PV^ω is a conservative extension of PV.

In Section 7 we introduce the system IPV^ω, which is a common extension of IPV and PV^ω. We prove (using cut elimination) that the extension is conservative over IPV. (The proof that IPV^ω is conservative over PV^ω must await Section 10.)

In Section 8 we present our version of modified realizability, and prove that every theorem of IPV^ω is provably realizable in IPV^ω. As a corollary we get Buss's result mentioned earlier that proofs of existence in IS_2^1 and IPV give rise to polynomial time algorithms for finding the numbers claimed to exist. We then apply the conservative extension result of Section 7 to establish that IPV proves that these polynomial time algorithms work. Some of Buss's conjectures for IS_2^1 follow.

In Sections 9 and 10 we present our version of Gödel's *Dialectica* interpretation, for IPV^ω. We prove that IPV^ω is a conservative extension of PV^ω, that IPV is a conservative extension of PV, that IS_2^1 is equivalent to IS_2^1B, and other applications.

In Section 11 we present results inspired by the preliminary version [9] of this paper.

Fig. 1 shows how the various systems stand with respect to the relation "is a conservative extension of".

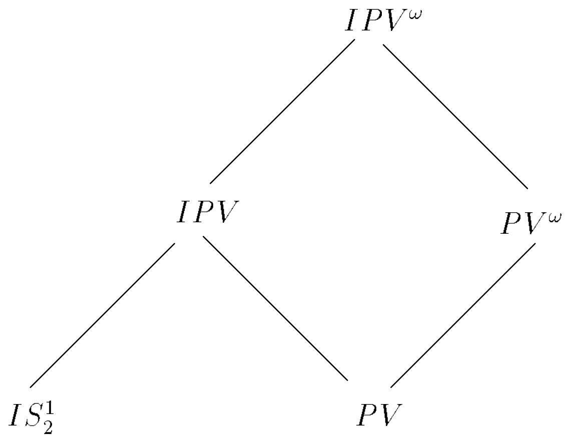

Figure 1: All extensions are conservative

0.5 Notation

$\equiv$ is syntactic identity

$\stackrel{d}{\equiv}$ is definitional equality

$S[t/x]$ is the result of substituting t for free occurrences of x in S, after changing bound variables in S to avoid clashes. (S can be a term or formula.)

1 Systems of Feasible Arithmetic

We begin by defining the logical basis and some of the notation for the systems considered in this paper. In particular, we describe the intuitionistic systems IS^1_2 and IS^1_2B and the classical system S^1_2.

The logical vocabulary for all systems of predicate logic considered consists of the propositional connectives $\wedge$, $\vee$, $\rightarrow$, the quantifiers $\forall$, $\exists$, and the equality predicate $=$. The connectives $\leftrightarrow$ and $\neg$ are introduced by definition below.

The non-logical constants of the systems IS^1_2, IS^1_2B and S^1_2 are the following: an individual constant 0, one-place functions Sx, $|x|$ and $\lfloor \frac{1}{2}x \rfloor$, two-place functions $x+y$, $x \cdot y$ and $x\#y$, and the two-place predicate $x \leq y$.

The intended interpretation of $|x|$ is $\lceil \log_2(x+1) \rceil$ (the length of the binary notation for x), and that of $x\#y$ is $2^{|x|\cdot|y|}$. Note that for $x > 1$, $\lfloor \frac{1}{2}x \rfloor$ is x with its rightmost bit deleted.

Definitions:

$1 \stackrel{d}{\equiv} S0$ $\qquad\qquad$ $A \leftrightarrow B \stackrel{d}{\equiv} (A \rightarrow B) \wedge (B \rightarrow A)$

$2 \stackrel{d}{\equiv} SS0$ $\qquad\qquad$ $\neg A \stackrel{d}{\equiv} (A \rightarrow (0 = 1))$.

Logical Axioms and Rules

For all intuitionistic systems, including IS^1_2 and IS^1_2B, we employ the following set of axioms and rules for intuitionistic predicate logic, which are due to Gödel [20]:

Axiom Schemes

1.	$A \rightarrow (A \wedge A)$	2.	$(A \vee A) \rightarrow A$
3.	$(A \wedge B) \rightarrow B$	4.	$B \rightarrow (A \vee B)$
5.	$(A \wedge B) \rightarrow (B \wedge A)$	6.	$(A \vee B) \rightarrow (B \vee A)$
7.	$\forall x A \rightarrow A[t/x]$	8.	$A[t/x] \rightarrow \exists x A$

9. $(0 = 1) \rightarrow A$

Rules of Inference

$$10.\quad \frac{A, A \to B}{B} \qquad 11.\quad \frac{A \to B, B \to C}{A \to C}$$

$$12.\quad \frac{(A \wedge B) \to C}{A \to (B \to C)} \qquad 13.\quad \frac{A \to (B \to C)}{(A \wedge B) \to C}$$

$$14.\quad \frac{A \to B}{C \vee A \to C \vee B}$$

$$15.\quad \frac{A \to B}{A \to \forall x B} \qquad 16.\quad \frac{B \to A}{\exists x B \to A}$$

In axiom schemes 7 and 8, $A[t/x]$ stands for the result of substituting the term t for all free occurrences of x in A; t must not contain an occurrence of a free variable which becomes bound in $A[t/x]$. In rules 15 and 16, x must not occur free in A.

For identity, we add the axiom schemes:

$$17.\quad x = x \qquad 18.\quad x = y \to (A \leftrightarrow A[y/x])$$

We shall write $\Gamma \vdash A$ if A is derivable from the set of formulas Γ, using the axiom schemes and rules of inference just given. In such a derivation, applications of rules 15 and 16 must satisfy the restriction that the premiss of the application does not depend on any formula in Γ which contains x free.

We shall use dots instead of brackets occasionally, in accordance with the conventions of Church [6].

For the classical systems S^1_2, CPV, and CPV^ω we add to the logical basis the axiom scheme of excluded middle:

$$19.\quad A \vee \neg A.$$

We define *bounded quantifiers* as follows:

$$(\exists x \leq t)A \overset{d}{\equiv} \exists x(x \leq t \ \wedge A)$$

$$(\forall x \leq t)A \overset{d}{\equiv} \forall x(St \leq x \ \vee A),$$

where t is a term not containing x. *Sharply bounded quantifiers* are bounded quantifiers of the form: $(\exists x \leq |t|)$, $(\forall x \leq |t|)$.

Following Buss [2], pp. 29–30, we define a hierarchy of bounded formulas:

(1) $\Pi^b_0 = \Sigma^b_0$ is the set of formulas all of whose quantifiers are sharply bounded.

(2) Σ^b_{k+1} is defined inductively by:

(a) $\Pi^b_k \subseteq \Sigma^b_{k+1}$;

(b) If A is in Σ_{k+1}^b so are $(\exists x \leq t)A$ and $(\forall x \leq |t|)A$;

(c) If A, B are in Σ_{k+1}^b, so are $A \wedge B$ and $A \vee B$;

(d) If A is in Σ_{k+1}^b and B in Π_{k+1}^b, then $\neg B$ and $B \rightarrow A$ are in Σ_{k+1}^b.

(3) Π_{k+1}^b is defined inductively by:

(a) $\Sigma_k^b \subseteq \Pi_{k+1}^b$;

(b) If A is in Π_{k+1}^b then so are $(\forall x \leq t)A$ and $(\exists x \leq |t|)A$;

(c) If A, B are in Π_{k+1}^b,so are $A \wedge B$ and $A \vee B$;

(d) If A is in Π_{k+1}^b and B in Σ_{k+1}^b, then $\neg B$ and $B \rightarrow A$ are in Π_{k+1}^b;

(4) Σ_{k+1}^b and Π_{k+1}^b are the smallest sets which satisfy (1)–(3).

A formula is *positive* if it contains no occurrence of $\rightarrow$ (recall that $\neg A$ is by definition $(A \rightarrow (0 = 1))$. A formula A is $H\Sigma_1^b$ (*hereditarily* Σ_1^b) if all subformulas of A are Σ_1^b. A formula is Σ_1^{b+} if it is both Σ_1^b and positive. All Σ_1^{b+} formulas are $H\Sigma_1^b$, but not conversely.

It follows from results of Meyer, Stockmeyer and Wrathall that, for $k > 0$, an n-place numerical predicate $Q(\vec{x})$ is in the k-th level Σ_k^p (Π_k^p) of the Meyer/Stockmeyer polynomial hierarchy if and only if it is defined by a Σ_k^b (Π_k^b) formula of S_2^1; for a proof see Buss [2], pp. 16-21. (This also follows easily from 2.7 below.) In particular, the NP sets are exactly those defined by Σ_1^b formulas.

Non-Logical Axioms for IS_2^1 and S_2^1

First, we have the following twenty-one BASIC axioms:

(1) $x = Sx \rightarrow 0 = 1$

(2) $0 \leq x$

(3) $x \leq y \rightarrow (x = y \vee Sx \leq y)$

(4) $(x \leq y \wedge y \leq z) \rightarrow x \leq z$

(5) $(x \leq y \wedge y \leq x) \rightarrow x = y$

(6) $x \leq y \vee y \leq x$

(7) $|0| = 0$

(8) $S0 \leq x \rightarrow |2 \cdot x| = S(|x|)$

(9) $|S(2 \cdot x)| = S(|x|)$

(10) $x \leq y \rightarrow |x| \leq |y|$

(11) $|x\#y| = S(|x| \cdot |y|)$

(12) $1\#1 = 2$

(13) $x\#y = y\#x$

(14) $|x| = |u| + |v| \rightarrow x\#y = (u\#y) \cdot (v\#y)$

(15) $x + 0 = x$

(16) $x + Sy = S(x + y)$

(17) $(x + y) + z = x + (y + z)$

(18) $x + y \leq x + z \leftrightarrow y \leq z$

(19) $x \cdot 1 = x$

(20) $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

(21) $x = (\lfloor \frac{1}{2}x \rfloor + \lfloor \frac{1}{2}x \rfloor) \vee x = S(\lfloor \frac{1}{2}x \rfloor + \lfloor \frac{1}{2}x \rfloor)$

Second, we have the axiom scheme of $\Sigma_1^{b+} - PIND$:

$$[A(0) \wedge \forall x(A(\lfloor \tfrac{1}{2}x \rfloor) \rightarrow A(x))] \quad \rightarrow \quad \forall z A(z)$$

where A is a Σ_1^{b+}-formula. The scheme of induction on binary notation formalized here is natural in the context of feasible mathematics, where we think of numbers as presented to us as binary strings. It may be considered as a more efficient form of induction than the usual form involving the normal successor function. Given a fixed number n, to prove that n has a certain property by the usual form of induction requires a number of steps proportional to n, whereas the number of steps using induction on notation is proportional to $\log_2 n$.

This completes the description of IS_2^1 and S_2^1, which differ only in their logical basis. The present formulation of S_2^1 differs from Buss's S_2^1 in having 21 BASIC axioms, where Buss's formulation has 32, but it is not difficult, though somewhat tedious, to show the two formulations equivalent.

The reader should be warned, however, that the derivation of Buss's BASIC axiom set from the present 21 axioms requires the use of the $PIND$ scheme for open formulas. In weak systems not containing this scheme, a stronger set of BASIC axioms is required. In a forthcoming note [3], Buss gives a detailed discussion of the relationship between the two definitions of the BASIC axioms and IS_2^1.

Buss's version of intuitionistic bounded arithmetic [4], IS^1_2B, differs from IS^1_2 in two respects. First, Buss postulates a more general form of the induction scheme; in the statement of $\Sigma^b_1 - PIND$ he allows $A(x)$ to be any $H\Sigma^b_1$ formula. Although this scheme is more convenient in proving theorems, it is not well behaved with respect to definitional extensions, which is why we adopt the more restricted scheme. Theorem 1.10 below shows that the more general scheme is derivable from the restricted one. Second, Buss replaces the BASIC axioms by the following rule: if $(A \to B)$ is a theorem of S^1_2 and A, B are $H\Sigma^b_1$ formulas then $(A \to B)$ is an axiom of IS^1_2B.

We shall not use the system IS^1_2B in what follows, but after we have developed functional interpretations for IS^1_2 we shall prove the two systems equivalent (10.6).

We associate with each formula A of IS^1_2 two positive formulas $POS(A)$ and $NEG(A)$, the positive and negative transforms of A.

Definition:

$$
\begin{array}{lllll}
(1) & \text{For A atomic:} & & \\
 & POS(A) = A & & NEG(x = y) = (Sx \le y \vee Sy \le x) \\
 & & & NEG(x \le y) = (Sy \le x); \\
(2) & POS(A \to B) & = & NEG(A) \vee POS(B) \\
 & NEG(A \to B) & = & POS(A) \wedge NEG(B); \\
(3) & POS(A \wedge B) & = & POS(A) \wedge POS(B), \\
 & NEG(A \wedge B) & = & NEG(A) \vee NEG(B); \\
(4) & POS(\forall x\ A) & = & \forall x\ POS(A), \\
 & NEG(\forall x\ A) & = & \exists x\ NEG(A); \\
(5) & POS(A \vee B) & = & POS(A) \vee POS(B), \\
 & NEG(A \vee B) & = & NEG(A) \wedge NEG(B); \\
(6) & POS(\exists x\ A) & = & \exists x\ POS(A), \\
 & NEG(\exists x\ A) & = & \forall x\ NEG(A).
\end{array}
$$

1.1 Lemma $BASIC \vdash (POS(A) \to A) \wedge (NEG(A) \to \neg A)$.

Proof: For atomic A, we need to prove $Sx \le y \to (x = y \to (0 = 1))$ and $Sy \le x \to (x \le y \to (0 = 1))$, which follow easily from the BASIC axioms 1, 2, 4, 5, 15, 16, 18, via the theorem $x \le x + y$ and its corollary $x \le Sx$.

The remainder of the proof is a straightforward induction on the complexity of A in which only intuitionistic predicate logic is required. □

1.2 Lemma $BASIC \vdash x = y \vee \neg x = y$.

Proof: By BASIC axioms 3 and 6, $IS^1_2 \vdash x = y \vee NEG(x = y)$. The result follows by Lemma 1.1. □

1.3 Lemma $IS_2^1 \vdash 2 \cdot x = x + x$.

Proof: By $\Sigma_1^{b+} - PIND$. The basis requires BASIC axioms 2, 5, 15, 18, 19, and 20. The induction step is based on axiom 21 and requires the theorem $1 + x = Sx$, which again uses $\Sigma_1^{b+} - PIND$ based on axiom 21. $\square$

1.4 Lemma $IS_2^1 \vdash x = 0 \vee |x| = S(|\lfloor \frac{1}{2}x \rfloor|)$.

Proof: By Lemma 1.2, $\lfloor \frac{1}{2}x \rfloor = 0$ or $\neg \lfloor \frac{1}{2}x \rfloor = 0$. In the first case, the result follows from axiom 21, using axioms 7 and 9 and the theorem $2 \cdot 0 = 0$ (from Lemma 1.3). In the second case, the result follows from axiom 21 and Lemma 1.3, using axioms 2, 3, 8, and 9. $\square$

Definition: The $\Sigma_1^{b+} - LIND$ schema has the form:

$$[A(0) \wedge \forall x(A(x) \rightarrow A(Sx))] \rightarrow \forall x\, A(|x|),$$

where $A(x)$ is a Σ_1^{b+} formula.

1.5 Lemma *The scheme of* $\Sigma_1^{b+} - LIND$ *is provable in* IS_2^1.

Proof: Let $B(x)$ be the formula $A(|x|)$. Then $B(0)$ follows from $A(0)$ by BASIC (7). Furthermore, from $\forall x(A(x) \rightarrow A(Sx))$ we can derive $\forall x(B(\lfloor \frac{1}{2}x \rfloor) \rightarrow B(x))$ by using Lemma 1.4. Since $B(x)$ is a Σ_1^{b+} formula, the conclusion $\forall x\, A(|x|)$ follows by $\Sigma_1^{b+} - PIND$. $\square$

1.6 Lemma *If* A *is* Σ_0^b *then* $IS_2^1 \vdash POS(A) \vee NEG(A)$.

Proof: By induction on the complexity of A. The induction step for sharply bounded quantifiers employs $\Sigma_1^{b+} - LIND$. $\square$

1.7 Corollary *If* A *is* Σ_0^b *then* $IS_2^1 \vdash A \vee \neg A$.

1.8 Corollary *If* A *is* Σ_0^b *then* $IS_2^1 \vdash (A \leftrightarrow POS(A)) \wedge (\neg A \leftrightarrow NEG(A))$.

Proof: By 1.1, $\vdash NEG(A) \rightarrow \neg A$, hence by 1.6, $\vdash A \rightarrow POS(A)$. Again by 1.1, $\vdash POS(A) \rightarrow A$ so $\vdash \neg A \rightarrow NEG(A)$ by 1.6. $\square$

1.9 Lemma *If* A *is* $H\Sigma_1^b$ *then* $IS_2^1 \vdash A \leftrightarrow POS(A)$.

Proof: This is a straightforward induction on the complexity of A. The only case worthy of remark is where A is of the form $B \rightarrow C$. In that case, B must be Σ_0^b, so:

$$\begin{aligned} \vdash POS(B \rightarrow C) &\leftrightarrow NEG(B) \vee POS(C) \\ &\leftrightarrow \neg B \vee C \\ &\leftrightarrow B \rightarrow C. \end{aligned}$$

The last step follows because $\vdash B \vee \neg B$ by Corollary 1.7. $\square$

1.10 Theorem *The* $H\Sigma_1^b - PIND$ *scheme is derivable in* IS_2^1. $\square$

Theorem 1.10 does not necessarily hold for definitional extensions of IS_2^1, such as we consider in the following section. This may be seen from the fact that Lemma 1.9 depends on the decidability of Σ_0^b formulas. This condition fails (for example) in an extension of IS_2^1 containing a non-recursive predicate introduced by definition.

2 Bootstrapping IS_2^1

Definition: Let T be an extension of IS_2^1. Let A be a Σ_1^{b+} formula of T and let t be a term of T such that

$$T \vdash \forall \vec{x}(\exists y \leq t)A(\vec{x}, y)$$

$$T \vdash \forall \vec{x}\, \forall y\, \forall z[A(\vec{x}, y) \wedge A(\vec{x}, z) \rightarrow y = z]$$

Then we say that T can Σ_1^{b+}-define the function f such that $\forall \vec{x}\ A(\vec{x}, f(\vec{x}))$. The *defining axiom* for f is:

$$f(\vec{x}) = y \leftrightarrow A(\vec{x}, y)$$

where f is a new n-ary function symbol. The *defining formula* for f is A and the *bounding term* for f is t.

2.1 Theorem *Let T and f be as in the above definition, and assume that all $\Sigma_1^{b+} - PIND$ axioms in the language of T are theorems of T. Let $T(f)$ be the extension of T obtained by adding f as a new function symbol, together with its defining axiom, and also all $\Sigma_1^{b+} - PIND$ axioms in the language of $T(f)$. Then $T(f)$ is a conservative extension of T.*

Proof: The theory obtained by adjoining only the defining axiom is well known to be a conservative extension of T; see Kleene [[24], §74] or Nelson [[28], Ch. 27] for a proof. Hence it suffices to prove that the $\Sigma_1^{b+} - PIND$ scheme is derivable in this theory. Let A be a Σ_1^{b+} formula in the expanded language containing a term $f(t_1 \ldots t_n)$, where $t_1 \ldots t_n$ are terms of T. Let the term $f(t_1 \ldots t_n)$ occur in an atomic subformula B of A. We can eliminate the occurrences of $f(t_1 \ldots t_n)$ in B by replacing B by the formula:

$$(\exists x \leq t)[B[x/f(t_1 \ldots t_n)] \wedge C(\vec{t}, x)]$$

where x is a variable not occurring in B, C is the defining formula for f and t is the bounding term. The formula which results from A by replacing B in this way is still Σ_1^{b+} (since A has no negative subformulas) and is equivalent to A in the theory. Proceeding in this way, we can find a Σ_1^{b+} formula A^* which is equivalent to A in the theory, but contains no occurrence of f. Hence, the $\Sigma_1^{b+} - PIND$ scheme for $T(f)$ can be derived from the $\Sigma_1^{b+} - PIND$ scheme for T. $\Box$

The above Theorem also extends to predicate symbols introduced by Σ_1^{b+} definitions. However, it should be noted that Lemma 1.6 only extends to the new theory provided that the new predicate P is *decidable* in the resulting theory, i.e. $\vdash P(\vec{x}) \vee \neg P(\vec{x})$. If, for example, P is a formalization of an NP-complete predicate, then it is extremely implausible that it is decidable in IS_2^1 (in fact, Proposition 8.15 below shows that its decidability would imply $P = NP$).

2.2 Definition *Let T be a theory of arithmetic containing a function $|x|$ which is a binary length function (i.e. $|x| = \lceil \log_2(x+1) \rceil$). Then we say that T contains a set of* efficient coding functions *provided that T contains a one-place predicate Seq, a one-place function Len, two-place functions $*$, β and $Bound$, for which the following are theorems of T:*

(A) $Seq(0) \wedge Len(0) = 0$;

(B) $Seq(s) \to Seq(s * u)$;

(C) $Seq(s) \to Len(s * u) = Len(s) + 1$;

(D) $Seq(s) \to Len(s) \leq |s|$;

(E) $Seq(s) \to .\ i < Len(s) \to \beta(i, s * u) = \beta(i, s)$;

(F) $Seq(s) \to \beta(Len(s), s * u) = u$;

(G) $(Seq(s) \wedge Seq(t)) \to . \left[\begin{array}{l} Len(s) = Len(t) \wedge \\ (\forall i < Len(s))(\beta(i,s) = \beta(i,t)) \end{array} \right] \to s = t$;

(H) $Seq(s) \wedge Len(s) \leq |b| + 1 \wedge (\forall i < Len(s))(\beta(i,s) \leq a) \to s \leq Bound(a,b)$.

The predicate letters and function symbols in Definition 2.2 can be explained informally as follows. $Seq(s)$ is true just in case s encodes a sequence of numbers $s_0, \ldots, s_k$; in that case, $Len(s) = k+1$, the length of the sequence. The null sequence is encoded by zero. If s encodes a sequence, then for any number u, $s * u$ encodes the sequence obtained by adding u as the last member. If s encodes a sequence $s_0, \ldots, s_k$, then $\beta(i,s) = s_i$. The function $Bound(a,b)$ bounds the size of a number encoding a sequence in terms of the largest entry in the sequence and the length of the sequence.

2.3 Definition *Let T be an extension of IS^1_2. We say that T is* sufficiently strong *if it satisfies the conditions:*

(1) *T contains a 2-place function $Lmin$ for which the theorem*

$$Lmin(x,y) = z \leftrightarrow (|x| \leq |y| \wedge z = x) \vee (\neg(|x| \leq |y|) \wedge z = y)$$

is provable in T;

(2) *T contains a set of efficient coding functions;*

(3) *T contains the $\Sigma^{b+}_1 - PIND$ scheme for all formulas in the language of T;*

(4) T contains a 2-place function $Trunc$ for which the theorems:

$$i \leq |\lfloor \tfrac{1}{2}a \rfloor| \ \rightarrow \ Trunc(\lfloor \tfrac{1}{2}a \rfloor, i) = Trunc(a, i)$$

$$Trunc(a, |a|) = a$$

are provable in T.

The intention is that $Trunc(a, i)$ is the first i bits of a.

2.4 Lemma *There is an extension IS_2^{1*} of IS_2^1 by Σ_1^{b+} definitions which is sufficiently strong.*

Proof: For the details of the definitional sequence required see Buss [2] pp. 37–50. Buss's proof of the corresponding lemma for S_2^1 uses classical rather than intuitionistic logic; however, it is not hard (though very tedious) to check that the proof is constructive, so that it also goes through in the context of intuitionistic logic. □

2.5 Lemma *Let T be a sufficiently strong extension of IS_2^1, and $k(a, \vec{b})$ an $n+1$-place function symbol of T. Then there is an extension T' of T by a Σ_1^{b+}-definition of a function m so that:*

$$\begin{array}{lll} T' & \vdash & k(a, \vec{b}) \leq m(a, \vec{b}) \\ T' & \vdash & m(\lfloor \tfrac{1}{2}a \rfloor, \vec{b}) \leq m(a, \vec{b}). \end{array}$$

Proof: It is straightforward to prove in T by $\Sigma_1^{b+} - PIND$ on a that:

$$\forall a (\exists i \leq |a|)(\forall j \leq |a|)[k(Trunc(a, j), \vec{b}) \leq k(Trunc(a, i), \vec{b})].$$

We can thus define:

$$\begin{array}{rll} m(a, \vec{b}) = c & \leftrightarrow & (\exists i \leq |a|)[k(Trunc(a, i), \vec{b}) = c \wedge \\ & & (\forall j \leq |a|)k(Trunc(a, j), \vec{b}) \leq c]. \end{array}$$

The uniqueness condition for m is immediate. The first inequality follows from the definition of m, and the second from the equation $Trunc(a, |\lfloor \tfrac{1}{2}a \rfloor|) = \lfloor \tfrac{1}{2}a \rfloor$. □

2.6 Theorem *Let T be a sufficiently strong extension of IS_2^1 and g, h, k n-place, $n+2$-place and $n+1$-place function symbols of T. Then there is an extension T^* of T by Σ_1^{b+}-definitions so that*

$$\begin{array}{lll} T^* & \vdash & f(0, \vec{b}) = g(\vec{b}) \\ T^* & \vdash & a = 0 \vee f(a, \vec{b}) = Lmin[h(a, \vec{b}, f(\lfloor \tfrac{1}{2}a \rfloor, \vec{b})), k(a, \vec{b})]. \end{array}$$

Proof: First, we extend T by adding a Σ_1^{b+} definition of a function $m(a,\vec{b})$ for which the theorems

$$\begin{array}{l} k(a,\vec{b}) + g(\vec{a}) + 1 \leq m(a,\vec{b}) \\ m(\lfloor \frac{1}{2}a \rfloor, \vec{b}) \leq m(a,\vec{b}) \end{array}$$

are provable. Lemma 2.5 guarantees the existence of such a definition.

Now let $B(w,a)$ be the following formula:

$$\begin{array}{l} \{Seq(w) \wedge Len(w) = |a| + 1 \wedge \beta(0,w) = g(\vec{b}) \\ \quad \wedge \\ (\forall i < |a|)[\beta(i+1,w) = Lmin[h(Trunc(a,i+1),\vec{b},\beta(i,w)), k(Trunc(a,i+1),\vec{b})]\}. \end{array}$$

The defining axiom for f is:

$$f(a,\vec{b}) = c \leftrightarrow (\exists w \leq Bound(m(a,\vec{b}),a))[B(w,a) \wedge \beta(|a|,w) = c].$$

To justify this definition, we need to prove the existence and uniqueness conditions.

<u>EXISTENCE</u>

We prove $(\exists w \leq Bound(m(a,\vec{b}),a))B(w,a)$ by $\Sigma_1^{b+} - PIND$ on a.

Case 1: $a = 0$. We wish to prove $(\exists w \leq Bound(m(0,\vec{b}),0))B(w,0)$. By 2.2(A), $Seq(0) \wedge Len(0) = 0$. Let w be $0 * g(\vec{b})$. By 2.2(B), $seq(w)$ and by 2.2(C) $Len(w) = 1$. By 2.2(F), $\beta(0,w) = \beta(Len(0),w) = g(\vec{b})$. Finally, $w \leq Bound(m(0,\vec{b}),0)$ by 2.2(H), completing the basis of the induction.

Case 2: Induction step. Assume $w_1 \leq Bound(m(\lfloor \frac{1}{2}a \rfloor,\vec{b}), \lfloor \frac{1}{2}a \rfloor)$, $B(w_1, \lfloor \frac{1}{2}a \rfloor)$ and $a > 0$. Define u as:

$$Lmin\{h(a,\vec{b},\beta(|\lfloor \tfrac{1}{2}a \rfloor|, w_1)), k(a,\vec{b})\}$$

and let w_2 be $w_1 * u$. We need to prove $B(w_2,a)$ and $w_2 \leq Bound(m(a,\vec{b}),a)$.

First, $seq(w_2)$ follows from 2.2(B). Second,

$$\begin{array}{rcll} Len(w_2) & = & Len(w_1 * u) & \\ & = & Len(w_1) + 1 & \text{(by 2.2(C))} \\ & = & (|\lfloor \frac{1}{2}a \rfloor| + 1) + 1 & \text{since } B(w_1, \lfloor \frac{1}{2}a \rfloor) \\ & = & |a| + 1. & \end{array}$$

Third,

$$\begin{array}{rcll} \beta(0,w_2) & = & \beta(0,w_1 * u) & \text{Df. } w_2 \\ & = & \beta(0,w_1) & \text{2.2(E)} \\ & = & g(\vec{b}) & \text{since } B(w_1, \lfloor \frac{1}{2}a \rfloor). \end{array}$$

Finally, we need to show that for all $i < |a|$,

$$\beta(i+1, w_2) = Lmin[h(Trunc(a, i+1), \vec{b}, \beta(i, w)), k(Trunc(a, i+1), \vec{b})].$$

Firstly, if $i < |\lfloor \frac{1}{2}a \rfloor|$ then

$$\begin{aligned}
& \beta(i+1, w_2) \\
= \;& \beta(i+1, w_1 * u) \\
= \;& \beta(i+1, w_1) && \text{by 2.2(E)} \\
= \;& Lmin\{h(Trunc(\lfloor \tfrac{1}{2}a \rfloor, i+1), \vec{b}, \beta(i, w_1)), k(Trunc(\lfloor \tfrac{1}{2}a \rfloor, i+1), \vec{b})\} \\
= \;& Lmin\{h(Trunc(a, i+1), \vec{b}, \beta(i, w_2)), k(Trunc(a, i+1), \vec{b})\} && \text{by 2.3(4) and 2.2(E).}
\end{aligned}$$

Secondly, if $i = |\lfloor \frac{1}{2}a \rfloor|$ then

$$\begin{aligned}
& \beta(i+1, w_2) \\
= \;& \beta(|\lfloor \tfrac{1}{2}a \rfloor| + 1, w_2) \\
= \;& \beta(Len(w_1), w_1 * u) \\
= \;& u && \text{by 2.2(F)} \\
= \;& Lmin\{h(a, \vec{b}, \beta(|\lfloor \tfrac{1}{2}a \rfloor|, w_1)), k(a, \vec{b})\} && \text{df. u} \\
= \;& Lmin\{h(Trunc(a, |a|), \vec{b}, \beta(i, w_1)), k(Trunc(a, |a|), \vec{b})\} && \text{by 2.3(4)} \\
= \;& Lmin\{h(Trunc(a, i+1), \vec{b}, \beta(i, w_2)), k(Trunc(a, i+1), \vec{b})\} && \text{by 2.2(E).}
\end{aligned}$$

This completes the proof that $B(w_2, a)$. We need to show, finally, that $w_2 \le Bound(m(a, \vec{b}), a)$. By definition, for $i < Len(w_2)$,

$$\begin{aligned}
\beta(i, w_2) &\le k(Trunc(a, i), \vec{b}) + 1 \\
&\le m(Trunc(a, i), \vec{b}) \\
&\le m(a, \vec{b}),
\end{aligned}$$

so by 2.2(H), $w_2 \le Bound(m(a, \vec{b}), a)$, completing the induction step for the existence condition.

UNIQUENESS

We need to show:

$$[B(v, a) \wedge B(w, a)] \rightarrow (\forall i < Len(v))(\beta(i, v) = \beta(i, w)).$$

Let $C(d)$ be the formula:

$$(\forall i \le |v|)[(i < Len(v) \wedge B(v, a) \wedge B(w, a) \wedge i \le d) \rightarrow \beta(i, v) = \beta(i, w)].$$

We prove $C(|d|)$ by $\Sigma_1^{b+} - LIND$ on the variable d.

For $d = 0, i = 0$ so $\beta(0, w) = \beta(0, v)$ by definition. Assume $C(d)$. If $i \leq d$, the result holds by induction hypothesis. Let $i = d + 1$. Then $i < Len(v) = |a| + 1$ so $d < |a|$, hence

$$\begin{array}{cll} & \beta(i, v) & \\ = & \beta(d+1, v) & \\ = & Lmin[h(Trunc(a, d+1), \vec{b}, \beta(d, v)), k(Trunc(a, d+1), \vec{b})] & \\ = & Lmin[h(Trunc(a, d+1), \vec{b}, \beta(d, w)), k(Trunc(a, d+1), \vec{b})] & \text{Induction Hypothesis} \\ = & \beta(d+1, w) = \beta(i, w). & \end{array}$$

By $\Sigma_1^{b+} - LIND$, the implication follows, so $v = w$ by 2.2(G).

This completes the proof that f is well defined. We need to establish the recursion equations for f.

The equation

$$f(0, \vec{b}) = g(\vec{b})$$

follows immediately from the definition of $B(w, 0)$. Next, we have:

$$(\exists w_1)[B(w_1, \lfloor \tfrac{1}{2} a \rfloor) \wedge \beta(|\lfloor \tfrac{1}{2} a \rfloor|, w_1) \;\; = \;\; f(\lfloor \tfrac{1}{2} a \rfloor, \vec{b})]$$

$$(\exists w_2)[B(w_2, a) \wedge \beta(|a|, w_2) \;\; = \;\; f(a, \vec{b})].$$

Let $u = Lmin[h(a, \vec{b}, \beta(|\lfloor \frac{1}{2} a \rfloor|, w_1)), k(a, \vec{b})]$ and $w_3 = w_1 * u$. Then it is easy to prove that $a = 0 \vee B(w_3, a)$ so that $w_3 = w_2$. Thus:

$$\begin{array}{rcl} \text{Either } a & = & 0 \quad \text{or} \\ f(a, \vec{b}) & = & \beta(|a|, w_2) = \beta(|a|, w_3) \\ & = & Lmin[h(a, \vec{b}, \beta(|\lfloor \frac{1}{2} a \rfloor|, w_1)), k(a, \vec{b})] \\ & = & Lmin[h(a, \vec{b}, f(\lfloor \frac{1}{2} a \rfloor, \vec{b})), k(a, \vec{b})]. \square \end{array}$$

2.7 Corollary *If $f(\vec{x})$ is a polynomial-time computable function then f is Σ_1^{b+}-definable in IS_2^1.*

Proof: A theorem of Cobham ([8] and [2], Sec. 1.1) characterizes the polynomial-time computable functions as those which are definable by composition and bounded recursion on notation from a small set of initial functions. The initial functions are Σ_1^{b+}-definable in IS_2^1, and the Σ_1^{b+}-definable functions are easily seen to be closed under composition. Closure under bounded recursion on notation follows from Theorem 2.6. $\square$.

3 The System PV

The system PV is a logic-free equational calculus, which bears the same relation to the polynomial-time computable functions as Skolem's primitive recursive arithmetic bears to the primitive recursive functions. The first formulation of PV is given in Cook [11], where PV stands for "polynomially verifiable". The idea is that a proof in PV of an equation $t = u$ provides a template for verifying each instance of the equation in time polynomial in the length of the instance. The main theorem states that for each theorem $t = u$, there is a polynomial bound on the length of the shortest extended Frege [16] proofs of the propositional formulas asserting the instances of $t = u$.

The formulation in [11] uses dyadic notation in the style of Smullyan [33] as its basis; the present formulation uses binary notation. A detailed development of PV appears in the M.Sc. thesis of Martin Dowd [17]. We are unable to follow Dowd's formulation for two reasons. First, his version uses dyadic notation; second, there is an error in the proof of his Lemma 3 which invalidates a substantial part of the earlier derivations (the derived rule numbered 35 in Appendix 1 is unsound). However, we wish to acknowledge our considerable debt to the work of Dowd in the detailed development below.

3.1 Definition *The* function symbols *and* terms *of PV are defined as follows:*

(1) *There are infinitely many numerical variables, and each such variable is a term.*

(2) *The constant* 0 *is a term.*

(3) *If f is an n-place function symbol and $t_1 \cdots t_n$ terms, then $(\cdots(ft_1)\cdots t_n)$ is a term.*

(4) $s_0, s_1, Parity, \lfloor \frac{1}{2} \rfloor$ *are* 1*-place function symbols.*

(5) $\dot{-}, \boxplus, \#$ *are* 2*-place function symbols.*

(6) $Cond$ *is a* 3*-place function symbol.*

(7) *If t is a term and $x_1 \cdots x_n$ $(n \geq 0)$ a list of variables including all variables in t, then $[\lambda x_1 \cdots x_n.t]$ is an n-place function symbol.*

(8) *If g, h and k are n-place, $n+2$-place and $n+1$-place function symbols respectively, then $R[g, h, k]$ is an $n+1$-place function symbol.*

The term notation $(\cdots(ft_1)\cdots t_n)$ in (3) is designed to fit the system PV^ω of §6. We shall depart from the formal definition above by employing the informal conventions of writing $ft_1 \cdots t_n$ or $f(t_1 \cdots t_n)$ instead of $(\cdots(ft_1)\cdots t_n)$, by employing infix notation $[x \dot{-} y$ for $((\dot{-}\ x)y)]$ and writing $\lfloor \frac{1}{2} x \rfloor$ for $(\lfloor \frac{1}{2} \rfloor x)$. The intended interpretation of the function symbols $s_0, s_1, \boxplus, \dot{-}, \#$ is as follows: $s_0 x = 2x$, $s_1 x = 2x+1$, $x \boxplus y = x \cdot 2^{|y|}$, $x \dot{-} y = \lfloor x \div 2^{|y|} \rfloor$, $x \# y = 2^{|x| \cdot |y|}$. We use "1" as an abbreviation of "$s_1 0$".

The terms "$x \dot{-} y$" "$x \boxplus y$" and "$x\#y$" are pronounced "x chop y", "x pad y" and "x smash y". The primitive functions of PV are most easily understood as operating on the binary strings which represent numbers. In these terms, s_0 and s_1 add a single digit to the right of a string, $x \boxplus y$ is the result of adding $|y|$ zeros to the end of the binary numeral for x, $x \dot{-} y$ is the result of chopping off $|y|$ digits from the numeral for x and the numeral for $x\#y$ is '1' followed by $|x| \cdot |y|$ zeros.

The axioms of PV give either explicit or recursive defining equations for each function symbol of PV except s_0 and s_1, which are considered primitive.

3.2 Definition *The* axioms of PV *are as follows:*

(0) $s_0(0) = 0$

(1a) $Parity(s_0 x) = 0$

(1b) $Parity(s_1 x) = 1$

(2a) $\lfloor \frac{1}{2} s_0 x \rfloor = x$

(2b) $\lfloor \frac{1}{2} s_1 x \rfloor = x$

(3a) $Cond(0, y, z) = y$

(3b) $Cond(s_0 x, y, z) = Cond(x, y, z)$

(3c) $Cond(s_1 x, y, z) = z$

(4a) $x \boxplus s_0 y = Cond(y, x, s_0(x \boxplus y))$

(4b) $x \boxplus s_1 y = s_0(x \boxplus y)$

(5a) $x \dot{-} s_0 y = Cond(y, x, \lfloor \frac{1}{2}(x \dot{-} y) \rfloor)$

(5b) $x \dot{-} s_1 y = \lfloor \frac{1}{2}(x \dot{-} y) \rfloor$

(6a) $x\#s_0 y = Cond(y, 1, (x\#y) \boxplus x)$

(6b) $x\#s_1 y = (x\#y) \boxplus x$

(7) $[\lambda x_1 \cdots x_n . t](x_1 \cdots x_n) = t$

(8) $R[g,h,k](x, \vec{y}) = Cond(x, g(\vec{y}), Cond(t \dot{-} k(x, \vec{y}), t, k(x, \vec{y})))$, *where* $t \stackrel{d}{\equiv} h(x, \vec{y}, R[g,h,k](\lfloor \frac{1}{2} x \rfloor, \vec{y}))$.

Note that according to (8), the function $R[g, h, k]$ is defined from g, h, and k by a form of bounded recursion on notation, with bounding function k. It follows from Cobham's theorem (Buss [2], Sec. 1.1) that every polynomial time computable function is represented by a function symbol in PV.

3.3 Definition *The* rules of inference *of PV are as follows:*

R1. $t = u \vdash u = t$

R2. $t = u, u = v \vdash t = v$

R3. $t_1 = u_1 \cdots t_n = u_n \vdash ft_1 \cdots t_n = fu_1 \cdots u_n$

R4. $t = u \vdash t[v/x] = u[v/x]$, *$x$ a variable, v any term*

R5.

$$\frac{\begin{array}{c} t_1[0/x] = t_2[0/x] \\ t_1[s_0x/x] = v_0[t_1/a] \quad t_2[s_0x/x] = v_0[t_2/a] \\ t_1[s_1x/x] = v_1[t_1/a] \quad t_2[s_1x/x] = v_1[t_2/a] \end{array}}{t_1 = t_2}$$

for any terms t_1, t_2, v_0, v_1 and any variable a.

The rule R5 is a form of induction on binary notation. The rule may be understood as follows: if the terms t_1 and t_2 satisfy the premisses of R5 for all x, then the functions defined by the two terms satisfy the same recursion equations, and hence are the same function.

We shall adopt "If $x = 0$, then t else u" as a way of writing "$Cond(x, t, u)$" in what follows.

In the formal development of PV, the abbreviations "T", "DR" and "D" are used for theorems, derived rules and definitions. Most of the theorems proved below are derived using induction on notation (R5). In such a case, the proof usually breaks down into three steps, corresponding to the base of the induction, and the two induction steps for the successor functions s_0 and s_1. In proving Theorem m we shall denote these three steps by (m.$\emptyset$), (m.0), (m.1) respectively; the two equations constituting step (m.0), for example, will be denoted by (m.0a) and (m.0b). In a case where the two steps (m.0) and (m.1) can be treated simultaneously, this will be written as (m.i). In a statement of a derived rule, the notation "$t_1 = u_1, \ldots, t_n = u_n \vdash v = w$" should be read as an abbreviation for "If $\vdash t_1 = u_1, \ldots, \vdash t_n = u_n$ then $\vdash v = w$".

The formal development of PV is very similar to that of primitive recursive arithmetic, but is complicated by the fact that function symbols introduced by definition contain a built-in bounding term. To make use of such a symbol in the later development, it is usually necessary to show that the bound can be eliminated, so that the recursion equations hold unconditionally. If $f(x, \vec{y})$ is a function symbol introduced by definition as $R[g, h, k](x, \vec{y})$, the temporary abbreviation "$t(f)$" will be used for the term "$h(x, \vec{y}, R[g, h, k](\lfloor \frac{1}{2}x \rfloor, \vec{y}))$" defined in Axiom 8.

T1: $Cond(x, y, y) = y$.
Proof: By induction on x:
(1.∅): $Cond(0, y, y) = y$ (Ax. 3a, R4);
(1.0): $Cond(s_0 x, y, y) = Cond(x, y, y)$ (Ax. 3b, R4);
(1.1): $Cond(s_1 x, y, y) = y$ (Ax. 3c, R4).
Hence, T1 follows by R5, setting $t_1 \equiv Cond(x, y, y)$, $t_2 \equiv y$, $v_0 \equiv a$, $v_1 \equiv y$ (note that $y = y$ follows from Ax. 2a by R1, R2, and R4). $\square$

In the sequel we shall usually not mention uses of R1, ..., R4.

T2: $Cond(x, fy_1 \cdots y_n, fz_1 \cdots z_n) = f(Cond(x, y_1, z_1) \cdots Cond(x, y_n, z_n))$, f any n-place function symbol of PV.
Proof: Induction on x:
(2.∅): $Cond(0, fy_1 \cdots y_n, fz_1 \cdots z_n)$
$= fy_1 \cdots y_n$
$= f(Cond(0, y_1, z_1) \cdots Cond(0, y_n, z_n))$ (Ax. 3a, R3);
(2.0a): $Cond(s_0 x, fy_1 \cdots y_n, fz_1 \cdots z_n)$
$= Cond(x, fy_1 \cdots y_n, fz_1 \cdots z_n)$ (Ax. 3b);
(2.0b): $f(Cond(s_0 x, y_1, z_1) \cdots Cond(s_0 x, y_n, z_n))$
$= f(Cond(x, y_1, z_1) \cdots Cond(x, y_n, z_n))$ (Ax. 3b, R3);
(2.1): $Cond(s_1 x, fy_1 \cdots y_n, fz_1 \cdots z_n)$
$= fz_1 \cdots z_n$ (Ax. 3c)
$= f(Cond(s_1 x, y_1, z_1) \cdots Cond(s_1 x, y_n, z_n))$ (Ax. 3c, R3).
Hence, T2 follows by R5 with $v_0 \equiv a$, $v_1 \equiv fz_1 \cdots z_n$. $\square$

T3: $Cond(x, fay_1 \cdots y_n, faz_1 \cdots z_n) = faCond(x, y_1, z_1) \cdots Cond(x, y_n, z_n)$, f any $(n+1)$-place function symbol, a any variable distinct from x.
Proof: Same as for T2. $\square$

T4: $Cond(x, Cond(x, c, d), Cond(x, e, f)) = Cond(x, c, f)$.
Proof: Induction on x:
(4.∅): $Cond(0, Cond(0, c, d), Cond(0, e, f))$
$= c = Cond(0, c, f)$ (Ax. 3a);
(4.0a): $Cond(s_0 x, Cond(s_0 x, c, d), Cond(s_0 x, e, f))$
$= Cond(x, Cond(x, c, d), Cond(x, e, f))$ (Ax. 3b);
(4.0b): $Cond(s_0 x, c, f) = Cond(x, c, f)$ (Ax. 3b);
(4.1): $Cond(s_1 x, Cond(s_1 x, c, d), Cond(s_1 x, e, f))$

$= f = Cond(s_1 x, c, f)$ (Ax. 3c).
Hence, T4 follows by R5, with $v_0 \equiv a$, $v_1 \equiv f$. $\Box$

T5: $Cond(x, 0, x) = x$.
Proof: By induction on x:
(5.∅): $Cond(0, 0, 0) = 0$ (Ax. 3a);
(5.0): $Cond(s_0 x, 0, s_0 x)$
$= Cond(x, 0, s_0 x)$ (Ax. 3b)
$= Cond(x, s_0 0, s_0 x)$ (Ax. 0)
$= s_0\{Cond(x, 0, x)\}$ (T2);
(5.1): $Cond(s_1 x, 0, s_1 x) = s_1 x$ (Ax. 3c).
Hence, T5 follows by R5 with $v_0 \equiv s_0 a$, $v_1 \equiv s_1 x$. $\Box$

T6: $Cond(x, u[0/x], u) = u$, where u is any term of PV.
Proof: By induction on the complexity of u. If $u \equiv x$, then

$$\begin{array}{rcll} Cond(x, u[0/x], u) & \equiv & Cond(x, 0, x) & \\ & = & u & \text{(T5);} \end{array}$$

if u is a variable, $u \not\equiv x$, or if $u \equiv 0$, then:

$$\begin{array}{rcll} Cond(x, u[0/x], u) & \equiv & Cond(x, u, u) & \\ & = & u & \text{(T1).} \end{array}$$

For the induction step, let u be $ft_1 \cdots t_n$. Then:

$$\begin{array}{rll} & Cond(x, u[0/x], u) & \\ = & Cond(x, ft_1[0/x] \cdots t_n[0/x], ft_1 \cdots t_n) & \\ = & f(Cond(x, t_1[0/x], t_1) \cdots Cond(x, t_n[0/x], t_n)) & \text{(T2)} \\ = & ft_1 \cdots t_n \quad \text{(I.H.)} & \\ = & u. \quad \Box & \end{array}$$

T7: $Cond(x, u, y) = Cond(x, u[0/x], y)$.
Proof:

$$\begin{array}{rcll} Cond(x, u, y) & = & Cond(x, Cond(x, u[0/x], u), y) & \text{(T6)} \\ & = & Cond(x, u[0/x], y) & \text{(T1, T4)}. \quad \Box \end{array}$$

T8: $Cond(x, y, u) = Cond(x, y, u[Cond(x, z, a)/a])$, for u any term of PV.
Proof: By induction on the complexity of the term u. For $u \equiv a$, this follows from T1

and T4. Assuming T8 for $t_1 \cdots t_n$;

$$\begin{array}{rll} & Cond(x,y,ft_1\cdots t_n) & \\ = & Cond(x,y,Cond(x,fy\cdots y,ft_1\cdots t_n)) & \text{(T1, T4)} \\ = & Cond(x,y,f(Cond(x,y,t_1)\cdots Cond(x,y,t_n))) & \text{(T2)} \\ = & Cond(x,y,f(Cond(x,y,t_1[Cond(x,z,a)/a])\cdots & \\ & \qquad Cond(x,y,t_n[Cond(x,z,a)/a]))) & \text{(I.H.)} \\ = & Cond(x,y,Cond(x,fy\cdots y,ft_1\cdots t_n[Cond(x,z,a)/a])) & \text{(T2)} \\ = & Cond(x,y,ft_1\cdots t_n[Cond(x,z,a)/a]) & \text{(T1, T4)}\ . \quad \Box \end{array}$$

D9: "$v \neq 0 \,.\!\supset\!.\ t = u$" abbreviates "$Cond(v,u,t) = u$", for any terms v, t, u.

DR10: $t = u \vdash v \neq 0 \,.\!\supset\!.\, t = u$.
Proof: Assuming $t = u$, by T1;

$$Cond(v,u,t) = Cond(v,u,u) = u. \quad \Box$$

T11: $x \neq 0 \,.\!\supset\!.\, Cond(x,t,u) = u$.
Proof:

$$\begin{array}{rcll} Cond(x,u,Cond(x,t,u)) & = & Cond(x,u,u) & \text{(T4)} \\ & = & u & \text{(T1)} \quad \Box \end{array}$$

DR12: $v \neq 0 \,.\!\supset\!.\, t = u \vdash v \neq 0 \,.\!\supset\!.\, u = t$.
Proof: By assumption, $Cond(v,u,t) = u$. Hence,

$$\begin{array}{rcll} Cond(v,t,u) & = & Cond(v,t,Cond(v,u,t)) & \\ & = & Cond(v,t,t) & \text{(T4)} \\ & = & t & \text{(T1)} \quad \Box \end{array}$$

DR13: $w \neq 0 \,.\!\supset\!.\, t = u,\ w \neq 0 \,.\!\supset\!.\, u = v \vdash w \neq 0 \,.\!\supset\!.\, t = v$.
Proof: By DR12, $w \neq 0 \,.\!\supset\!.\, u = t$, so $Cond(w,t,u) = t$. Hence:

$$\begin{array}{rcll} Cond(w,v,t) & = & Cond(w,v,Cond(w,t,u)) & \\ & = & Cond(w,v,u) & \text{(T4)} \\ & = & v, & \text{(Hypothesis)} \end{array}$$

that is, $w \neq 0 \,.\!\supset\!.\, t = v$. $\Box$

DR14: $v \neq 0 \,.\supset .\, t_i = u_i, i = 1 \cdots n \vdash v \neq 0 \,.\supset .\, ft_1 \cdots t_n = fu_1 \cdots u_n$.
Proof:

$$\begin{array}{rl} & Cond(v, fu_1 \cdots u_n, ft_1 \cdots t_n) \\ = & f(Cond(v, u_1, t_1) \cdots Cond(v, u_n, t_n)) \qquad \text{(T2)} \\ = & fu_1 \cdots u_n \qquad \text{(Hypothesis)} \quad \square \end{array}$$

The preceding theorem and derived rules show that all the basic properties of identity continue to hold under the hypothesis $v \neq 0$.

In proving theorems of PV, it is often natural to divide the proof into two cases: $v = 0$ and $v \neq 0$. Thus when applying a recursive definition of a function in PV, it is frequently necessary to assume that $x \neq 0$ in order to apply an inductive hypothesis to an expression containing s_0x, since $s_00 = 0$. This difficulty was avoided in the original formulation of PV [11], [17] by the adoption of dyadic notation; however, the definition of interleaved n-tuples is more complicated in dyadic notation than in binary. We now prove a derived rule which formalizes reasoning by cases.

DR15 (Conditional proof principle): $t[0/x] = u[0/x]$, $v \neq 0 \,.\supset .\, t' = u' \vdash t' = u'$ where $t' \equiv t[v/x]$ and $u' \equiv u[v/x]$.
Proof:

$$\begin{array}{rcl} t' & = & Cond(v, t[0/x], t') \qquad \text{(T6 } [v/x] \text{)} \\ & = & Cond(v, u[0/x], Cond(v, t', u')) \qquad \text{(DR12, Hypotheses)} \\ & = & Cond(v, u[0/x], u') \qquad \text{(T4)} \\ & = & u' \qquad \text{(T6 } [v/x] \text{)} \quad \square \end{array}$$

DR16: $t[0/x] = u[0/x]$, $x \neq 0 \,.\supset .\, t = u \vdash t = u$.
Proof: DR15 with $v \equiv x$. $\square$

DR17 (Conditional Induction):

$$\frac{\begin{array}{cc} \multicolumn{2}{c}{t_1[0/x] = t_2[0/x]} \\ x \neq 0 \,.\supset .\, t_1[s_0x/x] = w_0[t_1/a] & x \neq 0 \,.\supset .\, t_2[s_0x/x] = w_0[t_2/a] \\ t_1[s_1x/x] = w_1[t_1/a] & t_2[s_1x/x] = w_1[t_2/a] \end{array}}{t_1 = t_2.}$$

Proof: Rename the variable a if necessary, so that it does not occur in t_1, t_2. By DR12 and assumption, we have:

$$\begin{array}{rcl} t_1[s_0x/x] & = & Cond(x, t_1[s_0x/x], w_0[t_1/a]) \\ & = & Cond(x, t_1[0/x], w_0[t_1/a]) \qquad \text{(T7, Ax. 0)} \\ & = & Cond(x, t_2[0/x], w_0)[t_1/a] \qquad \text{(Hypothesis)} . \end{array}$$

Similarly:

$$t_2[s_0x/x] = Cond(x, t_2[0/x], w_0)[t_2/a].$$

Hence, we can infer $t_1 = t_2$ by R5, where $v_0 \equiv Cond(x, t_2[0/x], w_0)$, $v_1 \equiv w_1$. $\square$

This is the most useful form of the induction rule, when used in conjunction with DR10–DR14.

T18: $0 \dot{-} x = 0$.
Proof: By induction on x:
(18.$\emptyset$): $0 \dot{-} 0$
$= 0 \dot{-} s_0 0$ (Ax. 0)
$= Cond(0, 0, \lfloor \frac{1}{2}(0 \dot{-} 0) \rfloor)$ (Ax. 5a)
$= 0$ (Ax. 3a);
(18.0a): $0 \dot{-} s_0 x = Cond(x, 0, \lfloor \frac{1}{2}(0 \dot{-} x) \rfloor)$ (Ax. 5a);
(18.0b): $0 = Cond(x, 0, 0)$ (T1)
$= Cond(x, 0, \lfloor \frac{1}{2} 0 \rfloor)$ (Ax. 2a, 0);
(18.1a): $0 \dot{-} s_1 x = \lfloor \frac{1}{2}(0 \dot{-} x) \rfloor$ (Ax. 5b);
(18.1b): $0 = \lfloor \frac{1}{2} 0 \rfloor$ (Ax. 2a, 0).
Hence, $0 \dot{-} x = 0$ by R5, where $t_1 \equiv 0 \dot{-} x$, $t_2 \equiv 0$, $v_0 \equiv Cond(x, 0, \lfloor \frac{1}{2} a \rfloor)$, $v_1 \equiv \lfloor \frac{1}{2} a \rfloor$. $\square$

T19: $\lfloor \frac{1}{2}(s_i x \dot{-} y) \rfloor = x \dot{-} y$, $i = 0, 1$.
Proof: By conditional induction on y (DR17):
(19.$\emptyset$): $\lfloor \frac{1}{2}(s_i x \dot{-} 0) \rfloor$
$= \lfloor \frac{1}{2}(s_i x) \rfloor$ (Ax. 5a, 0, 3a)
$= x$ (Ax. 2)
$= x \dot{-} 0$ (Ax. 5a, 0, 3a);
(19.0a): $y \neq 0 \,.\!\supset\!.\, \lfloor \frac{1}{2}(s_i x \dot{-} s_0 y) \rfloor = \lfloor \frac{1}{2} \lfloor \frac{1}{2}(s_i x \dot{-} y) \rfloor \rfloor$ (Ax. 5a, T11, DR10–DR14);
(19.0b): $y \neq 0 \,.\!\supset\!.\, (x \dot{-} s_0 y) = \lfloor \frac{1}{2}(x \dot{-} y) \rfloor$ (Ax. 5a, T11, DR10–DR14),
(19.1a): $\lfloor \frac{1}{2}(s_i x \dot{-} s_1 y) \rfloor = \lfloor \frac{1}{2} \lfloor \frac{1}{2}(s_i x \dot{-} y) \rfloor \rfloor$ (Ax. 5b);
(19.1b): $x \dot{-} s_1 y = \lfloor \frac{1}{2}(x \dot{-} y) \rfloor$ (Ax. 5b).
Hence T19 follows by DR17, with $w_0 \equiv w_1 \equiv \lfloor \frac{1}{2} a \rfloor$. $\square$

T20: For $i = 0, 1$,

$$\begin{aligned} y \neq 0 \quad .\!\supset\!. \quad & s_i x \dot{-} s_0 y = x \dot{-} y; \\ & s_i x \dot{-} s_1 y = x \dot{-} y. \end{aligned}$$

Proof: For the second equation,

$$\begin{aligned} s_i x \dot{-} s_1 y &= \lfloor \tfrac{1}{2}(s_i x \dot{-} y) \rfloor && \text{(Ax. 5b)} \\ &= x \dot{-} y && \text{(T19).} \end{aligned}$$

The first assertion follows in the same way, except T11 and DR10–DR14 are used. $\square$

T21: $\lfloor \frac{1}{2}(x \dot{-} y) \rfloor = \lfloor \frac{1}{2}x \rfloor \dot{-} y$.
Proof: By induction on x:
(21.∅): $\lfloor \frac{1}{2}(0 \dot{-} y) \rfloor$
$= \lfloor \frac{1}{2}0 \rfloor$ (T18)
$= 0$
$= 0 \dot{-} y$ (T18)
$= \lfloor \frac{1}{2}0 \rfloor \dot{-} y$;
(21.i): $\lfloor \frac{1}{2}(s_i x \dot{-} y) \rfloor$
$= x \dot{-} y$ (T19)
$= \lfloor \frac{1}{2}\, s_i x \rfloor \dot{-} y$ (Ax. 2).
Hence, T21 follows by R5. $\square$

T22: $x \dot{-} x = 0$.
Proof: By conditional induction on x:
(22.∅): $0 \dot{-} 0 = 0$;
(22.i): $x \neq 0 \;.\!\supset\!.\; s_i x \dot{-} s_i x = x \dot{-} x$ (T20).
Hence, T22 follows by DR17. $\square$

T23: $R[g,h,k](x,\vec{y}) \dot{-} k(x,\vec{y}) = Cond(x, g(\vec{y}) \dot{-} k(0,\vec{y}), 0)$.
Proof: $R[g,h,k](x,\vec{y}) \dot{-} k(x,\vec{y})$
$= Cond(x, g(\vec{y}), Cond(t \dot{-} k(x,\vec{y}), t, k(x,\vec{y}\,))) \dot{-} k(x,\vec{y})$ (Ax. 8)
$= Cond(x, g(\vec{y}) \dot{-} k(x,\vec{y}), Cond(t \dot{-} k(x,\vec{y}), t, k(x,\vec{y})) \dot{-} k(x,\vec{y}))$
(T3 with $fay = y \dot{-} a$ and $a = k(x,\vec{y})$)
$= Cond(x, g(\vec{y}) \dot{-} k(x,\vec{y}), Cond(t \dot{-} k(x,\vec{y}), t \dot{-} k(x,\vec{y}), 0))$ (T3, T22)
$= Cond(x, g(\vec{y}) \dot{-} k(x,\vec{y}), 0)$ ($Cond(x,x,0) = 0$ by T7, T1)
$= Cond(x, g(\vec{y}) \dot{-} k(0,\vec{y}), 0)$ (T7). $\square$

The successor function S has the form $R[g,h,k]$, where $g \equiv [\lambda.1]$, $h \equiv [\lambda xz.Cond(Parity(x), s_1\lfloor \frac{1}{2}x \rfloor, s_0 z)]$, and $k \equiv s_1$. From Ax. 8 we have

D24 (The successor function):

$$Sx = Cond(x, 1, Cond(t(S) \dot{-} s_1 x, t(S), s_1 x)),$$

where $t(S) \equiv Cond(Parity(x), s_1\lfloor \frac{1}{2}x \rfloor, s_0 S\lfloor \frac{1}{2}x \rfloor)$.

T25: $Sx \dot{-} s_1 x = 0$.
Proof: $Sx \dot{-} s_1 x$
$= Cond(x, 1 \dot{-} s_1 0, 0)$ (T23)
$= 0$ (T22, T1). $\square$

T26: $Sx = Cond(Parity(x), s_1\lfloor\frac{1}{2}x\rfloor, s_0S\lfloor\frac{1}{2}x\rfloor)$.
Proof: We establish $t(S) \dot{-} s_1x = 0$ by induction on x:
(26.$\emptyset$): $Cond(Parity(0), s_1\lfloor\frac{1}{2}0\rfloor, s_0S\lfloor\frac{1}{2}0\rfloor) \dot{-} s_10$
$= s_10 \dot{-} s_10 = 0$;
(26.0): $Cond(Parity(s_0x), s_1\lfloor\frac{1}{2}s_0x\rfloor, s_0S\lfloor\frac{1}{2}s_0x\rfloor) \dot{-} s_1s_0x$
$= s_1x \dot{-} s_1s_0x$ (Axs. 1a, 2a, 3a)
$= x \dot{-} s_0x$ (T20)
$= Cond(x, x, \lfloor\frac{1}{2}(x \dot{-} x)\rfloor)$ (Ax. 5a)
$= Cond(x, x, 0)$ (T22)
$= 0$ (T7, T1);
(26.1): $Cond(Parity(s_1x), s_1\lfloor\frac{1}{2}s_1x\rfloor, s_0S\lfloor\frac{1}{2}s_1x\rfloor) \dot{-} s_1s_1x$
$= s_0S\lfloor\frac{1}{2}s_1x\rfloor \dot{-} s_1s_1x$ (Axs. 1b, 3c)
$= Sx \dot{-} s_1x$ (T20, Ax. 2b)
$= 0$ (T25).
Hence, $t(S) \dot{-} s_1x = 0$ by R5, where $v_0 \equiv v_1 \equiv 0$; thus T26 follows by T7 and T1. □

The preceding theorem illustrates the typical method of proof in PV, in which a definition by bounded recursion is immediately followed by a proof of an inequality which establishes the unconditional recursion scheme. If f is introduced by the definition $R[g, h, k](x, \vec{y})$, then the equation "$t(f) \dot{-} k(x, \vec{y}) = 0$" will be called "the bounding inequality for f".

T27: $x \dot{-} 1 = \lfloor\frac{1}{2}x\rfloor$.
Proof: By Axioms 5b, 0, 5a, 3a. □

The abbreviations "$x0$" for "$s_0(x)$" and "$x1$" for "$s_1(x)$" are employed below.

DR28: $t[x0/x] = u[x0/x]$, $t[x1/x] = u[x1/x] \vdash t = u$.
Proof: $t = u$ is proved by induction on x (R5) with $t_1 \equiv t$, $t_2 \equiv u$, $v_0 \equiv u[x0/x]$, $v_1 \equiv u[x1/x]$. □

T29: $x = Cond(Parity(x), \lfloor\frac{1}{2}x\rfloor 0, \lfloor\frac{1}{2}x\rfloor 1)$.
Proof: Use DR28. □

T30: $Parity(Parity(x)) = Parity(x)$.
Proof: By Ax. 1, $Parity(Parity(xi)) = Parity(i) = i = Parity(xi)$, for $i = 0, 1$. Hence T30 follows by DR28. □

D31: $sg(x) \stackrel{d}{\equiv} Cond(x, 0, 1)$.

D32: $\overline{sg}(x) \stackrel{d}{\equiv} Cond(x, 1, 0)$.

We shall write "$\sim x$" for "$\overline{sg}(x)$".

D33: $(x\&y) \stackrel{d}{\equiv} Cond(x, sg(y), 1)$.

D34: $(x \vee y) \stackrel{d}{\equiv} Cond(x, 0, sg(y))$.

D35: $(x \Rightarrow y) \stackrel{d}{\equiv} Cond(x, sg(y), 0)$.

D36: $(x \Leftrightarrow y) \stackrel{d}{\equiv} Cond(x, sg(y), \overline{sg}(y))$.

Note that we are employing the convention: $0 =$ true, $1 =$ false.

We define the class of *propositional terms* as follows:

(a) $0, 1, x, y, z \ldots$ are propositional terms;

(b) if P, Q are propositional terms, so are $sg(P)$, $\sim P$, $(P\&Q)$, $(P \vee Q)$, $(P \Rightarrow Q)$ and $(P \Leftrightarrow Q)$.

T37: $Cond(sg(x), y, z) = Cond(x, y, z)$.
Proof: The argument is by cases. For $x = 0$, $Cond(sg(0), y, z) = Cond(0, y, z)$. If $x \neq 0$, then $sg(x) = 1$, so $Cond(sg(x), y, z) = z = Cond(x, y, z)$, by T11. Hence, T37 follows by DR16. □

In the preceding proof, DR10–DR14 were used without special mention for reasoning under the hypothesis $x \neq 0$; they will be used tacitly in the subsequent proofs.

T38: $Cond(\sim x, y, z) = Cond(x, z, y)$.
Proof: By cases, as in T37. □

T39: $sg(sg(x)) = sg(x)$.
Proof: $sg(sg(x))$

$= Cond(sg(x), 0, 1)$

$= Cond(x, 0, 1)$ (T37)

$= sg(x)$. □

T40: $sg(Parity(x)) = Parity(x)$.
Proof: For $i = 0, 1$, $sg(Parity(xi)) = sg(i) = Parity(xi)$. The result follows by DR28. □

T41: If P is a propositional term which is not a variable then $P[sg(x)/x] = P$.
Proof: This follows by induction on the complexity of P, from equations such as $sg(x)\&y = x\&y$, which are easily established from the definitions D33–D36. □

T42: $Cond(x, y, t[sg(x)/x]) = Cond(x, y, t[1/x])$.
Proof: By T8, or by the cases $x = 0$, $x \neq 0$ using DR16. □

T43: If P and Q are equivalent propositional terms and neither is a variable, then $P = Q$.
Proof: By induction on the number of variables in P and Q. If P and Q contain no variables, the claim follows from the fact that the usual truth table identities are provable in PV.

Assume for n variables. If P and Q contain $n + 1$ variables including x, then $P[0/x]$, $P[1/x]$, $Q[0/x]$ and $Q[1/x]$ are each equal in PV to a propositional term. Hence, by the induction hypothesis, $P[0/x] = Q[0/x]$ and $P[1/x] = Q[1/x]$. Hence

$$\begin{aligned}
P &= Cond(x, P, P) && \text{(T1)} \\
&= Cond(x, P[0/x], P[sg(x)/x]) && \text{(T7, T41)} \\
&= Cond(x, P[0/x], P[1/x]) && \text{(T42)} \\
&= Cond(x, Q[0/x], Q[1/x]) && \text{(Ind. Hyp.)} \\
&= Q && \text{(apply first 3 lines to } Q\text{).} \quad \Box
\end{aligned}$$

T44: If P is a propositional term which is not a variable then $sg(P) = P$.
Proof: By D33 and DR16, $x\&y = sg(x\&y)$. Similarly for each of the propositional connectives. $\Box$

T45: If P is a propositional term which is not a variable, then $P \dot{-} 1 = 0$.
Proof: This follows immediately from such equations as $(x\&y) \dot{-} 1 = 0$. $\Box$

DR46: Let P and Q be arbitrary terms. $(P \Rightarrow Q) = 0$, $P = 0 \vdash Q = 0$.
Proof: If $P = 0$, and $(P \Rightarrow Q) = 0$, then $sg(Q) = Cond(P, sg(Q), 0) = 0$. Hence, by T5,

$$\begin{aligned}
Q &= Cond(Q, 0, Q) \\
&= Cond(sg(Q), 0, Q) && \text{(T37)} \\
&= Cond(0, 0, Q) \\
&= 0. \quad \Box
\end{aligned}$$

Notice that since Q is not necessarily a propositional term, P may not be truth functionally valid even when the premisses hold. DR46 and T43 allow us to use general propositional reasoning in PV. That is, if $P_1, \ldots, P_k \vdash Q$ is truth functionally sound, then $[P_1 \Rightarrow (P_2 \Rightarrow \cdots \Rightarrow (P_k \Rightarrow Q) \cdots)] = 0$ by T43, so we may conclude Q from $P_1, \ldots, P_k$ by repeated use of DR46.

D47: "$t \neq 0$" abbreviates "$\overline{sg}(t) = 0$".

D48: $P \;.\!\supset\!.\; t = u$ abbreviates "$Cond(P, t, u) = u$"

Note that these two conventions are consistent with D9, in the sense that by T38, the D9 meaning of "$v \neq 0 \ .\supset.\ t = u$" is equivalent to the D48 meaning of "$\overline{sg}(v) \ .\supset.\ t = u$". Further, it is easy to check that the analogs of DR10–DR14 hold for D48. In addition, we have

DR49: $P \ .\supset.\ t = u, P = 0 \vdash t = u$.
Proof: Immediate. □

DR50: $P \ .\supset.\ t = u, Q \ .\supset.\ t = u \vdash P \vee Q \ .\supset.\ t = u$.
Proof: By assumption, $Cond(P, t, u) = u$ and $Cond(Q, t, u) = u$, hence

$$\begin{aligned} Cond(P \vee Q, t, u) &= Cond(Cond(P, 0, sg(Q)), t, u) \\ &= Cond(P, Cond(0, t, u), Cond(sg(Q), t, u)) \qquad \text{(DR15 with } v \equiv P) \\ &= Cond(P, t, Cond(Q, t, u)) \qquad \text{(T37)} \\ &= Cond(P, t, u) \\ &= u. \quad \square \end{aligned}$$

DR51: $sg(t) = 0 \vdash t = 0$.
Proof: $[sg(x) \Rightarrow x] = 0$ by T43, so $[sg(t) \Rightarrow t] = 0$. Hence $t = 0$ by the premiss and DR46. □

DR52: For any function symbol f, $f(0, \vec{y}) = 0$, $[f(x, \vec{y}) \Rightarrow f(xi, \vec{y})] = 0$, $i = 0, 1 \vdash f(x, \vec{y}) = 0$.
Proof: We prove $sg(f(x, \vec{y})) = 0$ by induction on x. The result follows by DR51.
(52.∅): $sg(f(0, \vec{y})) = sg(0) = 0$.
(52.i): $sg(f(xi, \vec{y}))$
$= Cond(f(x, \vec{y}), [f(x, \vec{y}) \Rightarrow f(xi, \vec{y})], sg(f(xi, \vec{y})))$ (DR15 with $v \equiv f(x, \vec{y})$, D35)
$= Cond(f(x, \vec{y}), 0, sg(f(xi, \vec{y})))$ (premiss)
$= Cond(0, 0, sg(f(xi, \vec{y}))$ (T37, I.H.)
$= 0$.
The induction R5 is applied with $v_i \stackrel{d}{\equiv} Cond(a, 0, sg(f(xi, \vec{y}))$. □

To continue the development of PV, and in particular to define such basic functions as the equality function, addition and multiplication, it is necessary to show as a derived rule that functions may be defined by bounded recursion on several variables simultaneously, rather than on just one variable, as in Axiom 8. Accomplishing this requires a rather lengthy development leading to the introduction of interleaved n-tuple functions and associated projection functions. This material is relegated to the appendix. The basic recursion equations satisfied by these functions are contained in the following theorems:

T100: $< 0, \ldots, 0 > = 0$.

T101: $< x_1 i_1, \ldots, x_n i_n > = < x_1, \ldots, x_n > i_1 \cdots i_n$, where $i_1, \ldots, i_n \in \{0, 1\}$,

T116: $\Pi_k^n(< x_1, \ldots, x_k, \ldots, x_n >) = x_k$.

The derived rules of multi-variable induction and multi-variable bounded recursion are proved in the appendix as DR118 and DR120.

In the following section, we require the fact that the characteristic functions of the relations $x = y$, $x < y$, and $x \leq y$ are definable in PV. These functions $Equ(x, y)$, $Less(x, y)$, $Lessequ(x, y)$ are defined in the appendix by D121, D136 and D137.

4 The System IPV

The system IPV arises by adding intuitionistic predicate logic to PV, together with a form of induction on NP predicates. Because of the availability of function symbols in PV, the form of predicates used in the induction scheme is much more restricted than in IS_2^1. This restriction will simplify the realizability interpretations in §8 and §9. We shall show in this section that IPV is a conservative extension of IS_2^1.

The predicate symbols of IPV are the same as those of IS_2^1, that is, $x = y$ and $x \leq y$. The terms and function symbols of IPV are those of PV (Def. 3.1). Bounded quantifiers and the classes of formulas $\Sigma_k^b, \Pi_k^b, \Sigma_k^{b+}, \Pi_k^{b+}$ are defined in §1.

4.1 Definition *The nonlogical axioms of IPV are:*

(1) *All axioms of PV*

(2) $x \leq y \leftrightarrow Lessequ(x, y) = 0$

(3) $x = s_0\lfloor \frac{1}{2}x \rfloor \vee x = s_1\lfloor \frac{1}{2}x \rfloor$

(4) $Cond(x, a, b) = c \leftrightarrow (x = 0 \wedge a = c) \vee (\neg(x = 0) \wedge b = c)$

(5) *Any formula of the form:*

$$[A(0) \wedge \forall x(A(\lfloor \tfrac{1}{2}x \rfloor) \rightarrow A(x))] \rightarrow \forall z A(z),$$

*where A is of the form $(\exists y \leq t)u = v$, t, u, v terms of PV. Any such formula is an instance of the NP-*induction *scheme.*

4.2 Theorem *Any theorem of PV is a theorem of IPV.*

Proof: It is sufficient to establish the rule R5 of PV as a derived rule of IPV. To apply the NP-induction scheme, we need to derive

$$\forall x(t_1[\lfloor \tfrac{1}{2}x \rfloor/x] = t_2[\lfloor \tfrac{1}{2}x \rfloor/x] \rightarrow t_1 = t_2)$$

from the premisses of R5. Assume these premisses, and $t_1[\lfloor \frac{1}{2}x \rfloor/x] = t_2[\lfloor \frac{1}{2}x \rfloor/x]$. If $x = s_0\lfloor \frac{1}{2}x \rfloor$ then defining $u_0 \stackrel{d}{\equiv} v_0[\lfloor \frac{1}{2} \rfloor/x]$,

$$\begin{aligned} t_1 &= t_1[s_0\lfloor \tfrac{1}{2}x \rfloor/x] \\ &= u_0[t_1[\lfloor \tfrac{1}{2}x \rfloor/x]/a] \\ &= u_0[t_2[\lfloor \tfrac{1}{2}x \rfloor/x]/a] \\ &= t_2[s_0\lfloor \tfrac{1}{2}x \rfloor/x] \\ &= t_2. \end{aligned}$$

Similarly if $x = s_1\lfloor \frac{1}{2}x \rfloor$; so by Axiom 4.1.3, $t_1 = t_2$. We can now use the first premiss of R5 and the implication just proved to derive $t_1 = t_2$ by NP-induction. $\Box$

We now show that the defined logical operations of PV are in effect equivalent to the corresponding primitive operations of IPV. To deal with sharply bounded quantifiers, we employ the functions $f^{\exists}$ and $f^{\forall}$ of PV (Appendix, DR204).

4.3 Theorem *The following are theorems of IPV:*

(1) $Equ(x, y) = 0 \leftrightarrow (x = y)$

(2) $\sim x = 0 \leftrightarrow \neg(x = 0)$

(3) $x \& y = 0 \leftrightarrow (x = 0 \wedge y = 0)$

(4) $x \vee y = 0 \leftrightarrow (x = 0 \vee y = 0)$

(5) $x \supset y = 0 \leftrightarrow (x = 0 \rightarrow y = 0)$

(6) $x \Leftrightarrow y = 0 \leftrightarrow (x = 0 \leftrightarrow y = 0)$

(7) $f^{\exists}(a, \vec{y}) = 0 \leftrightarrow (\exists x \leq |a|) f(x, \vec{y}) = 0$

(8) $f^{\forall}(a, \vec{y}) = 0 \leftrightarrow (\forall x \leq |a|) f(x, \vec{y}) = 0$

Proof:

(1) By T125, $Cond(Equ(x, y), x, y) = y$. Hence, if $Equ(x, y) = 0$ then $x = Cond(Equ(x, y), x, y) = y$. The converse follows from $Equ(x, x) = 0$ (T180).

(2) If $\sim x = 0$ and $x = 0$ then $0 = Cond(x, 1, 0) = 1$. Conversely, if $\neg(x = 0)$ then $\sim x = 0$ by 4.1.4.

(3) If $x \& y = 0$ then by 4.1.4, $x = 0$ and $sg(y) = 0$, hence $y = 0$ by 4.1.4.

(4, 5, 6) The theorem $x = 0 \vee \neg(x = 0)$ follows from 4.1.4 and T1; thus all classical 2-valued tautologies constructed from atomic formulas are theorems of IPV. The cases 4–6 then follow directly from the classical definitions of $\vee$, $\supset$ and $\Leftrightarrow$ in terms of $\&$, $\sim$, and cases 2 and 3 above.

Axiom 4.1.2 and (1)–(6) show that any quantifier-free formula of IPV is provably equivalent to an equation. This allows us to extend the NP-induction scheme to the case where A has the form $(\exists y \leq t)B$, where B is quantifier-free.

(7) We first establish

$$(x \leq |a| \wedge f(x, \vec{y}) = 0) \rightarrow f^{\exists}(a, \vec{y}) = 0,$$

by NP-induction on the variable a. If $a = 0$ then if $x \leq |0|$, $x = 0$ so if $f(x, \vec{y}) = 0$ then $f^{\exists}(a, \vec{y}) = 0$ by DR204. For the induction step, assume $(x \leq |\lfloor \frac{1}{2}a \rfloor| \wedge f(x, \vec{y}) = 0) \rightarrow f^{\exists}(\lfloor \frac{1}{2}a \rfloor, \vec{y}) = 0$, $x \leq |a|$ and $f(x, \vec{y}) = 0$. Then $x \leq |\lfloor \frac{1}{2}a \rfloor|$ or $x = |a|$ (by the appropriate theorem of PV); in the first case $f^{\exists}(a, \vec{y}) = 0$ by inductive assumption, in the second case $f^{\exists}(a, \vec{y}) = 0$ by DR204.

To complete the proof of (7), we prove by NP-induction on a that $(\exists x \leq |a|)(f^{\exists}(a, \vec{y}) = 1 \vee f(x, \vec{y}) = 0)$. For $a = 0$, $f^{\exists}(0, \vec{y}) = 1 \vee f(0, \vec{y}) = 0$ follows by DR204. Assume for $\lfloor \frac{1}{2}a \rfloor$. If $(\exists x \leq |\lfloor \frac{1}{2}a \rfloor|)f(x, \vec{y}) = 0$ then $(\exists x \leq |a|)(f^{\exists}(a, \vec{y}) = 1 \vee f(x, \vec{y}) = 0)$ follows immediately. If $f^{\exists}(\lfloor \frac{1}{2}a \rfloor, \vec{y}) = 1$ then if $f(|a|, \vec{y}) = 1$ then $f^{\exists}(a, \vec{y}) = 1$ by DR204; in the other case, $(\exists x \leq |a|)f(x, \vec{y}) = 0$, completing the proof.

(8) From (7) it follows that $(\exists x \leq |a|)f(x, \vec{y}) = 0 \vee \neg(\exists x \leq |a|)f(x, \vec{y}) = 0$ is a theorem of IPV, hence $(\forall x \leq |a|)f(x, \vec{y}) = 0 \leftrightarrow \neg(\exists x \leq |a|)\neg(f(x, \vec{y}) = 0)$. Case (8) thus follows from (7) and the PV theorem $f^{\forall}(a, \vec{y}) = \sim (\sim f)^{\exists}(a, \vec{y})$. $\Box$.

4.4 ***Theorem*** *If A is a Σ_0^b formula of IPV (i.e. all quantifiers are sharply bounded) then there is a term t^A of PV so that*

$$IPV \vdash A \leftrightarrow t^A = 0.$$

Proof: By induction on the complexity of A, using Axiom 4.1.2 and Theorem 4.3. $\Box$

4.5 ***Theorem*** *If A is a Σ_0^b formula of IPV then $IPV \vdash A \vee \neg A$.*

Proof: By 4.4, 4.3, and T43 in §3. $\Box$.

4.6 ***Lemma*** *The Σ_1^b-$PIND$ and Σ_1^b-$LIND$ schemes are derivable in IPV, provided that the formula $A(x)$ is of the form $(\exists y \leq t)B$, where B is Σ_0^b.*

Proof: By 4.4, A is equivalent to a formula of the form $(\exists y \leq t)t^B = 0$, so that the Σ_1^b-$PIND$ scheme for $A(x)$ is a theorem of IPV. The Σ_1^b-$LIND$ scheme for $A(x)$ is proved exactly as in 1.5. $\Box$

4.7 ***Theorem*** *There are functions $Sx, |x|, \lfloor \frac{1}{2}x \rfloor$, $x + y, x \cdot y$ and $x \# y$ definable in PV so that all the $BASIC$ axioms of IS_2^1 are provable in IPV.*

Proof: The theorems of PV which correspond to the $BASIC$ axioms of IS_2^1 are: 148, 144, 147, 139, 141, 142, 135, 192, 193, 151, 194, 195, 198, 202, 155, 158, 159, 162, 184, 187, 203. The $BASIC$ axioms follow immediately from these theorems by 4.3. $\Box$

4.8 ***Theorem*** *IPV contains a set of efficient coding functions.*

Proof: A sequence $s_0, \ldots, s_k$ can be encoded by replacing each digit in an entry in the sequence by a two-digit sequence, then concatenating all the encoded entries with a two-digit code as a separator. The resulting coding functions are fairly easy to define in PV, and the theorems required by Definition 2.2 may be established by using multi-variable induction (DR118). □

4.9 Definition *The* Σ_0^b-replacement scheme *is the family of formulas of the form:*

$$(\forall x \leq |t|)(\exists y \leq s)A(x,y) \leftrightarrow (\exists w \leq Bound(s^*, t))(\forall x \leq |t|)[A(x, \beta(x,w)) \wedge \beta(x,w) \leq s],$$

where $A(x,y)$ is a Σ_0^b formula of IPV and $s^* = s^M[|t|/x]$, where s^M is the monotone upper bound on s obtained from T205 (Appendix).

This replacement scheme is similar to the one in §2.7 of Buss [2]. We note that Buss's scheme should be corrected to have $SqBd(s^*, t)$ instead of $SqBd(s,t)$, where now $s^* = s[|t|/x]$.

4.10 Theorem *All instances of the Σ_0^b-replacement scheme are provable in IPV.*

Proof: The right-to-left implication follows by intuitionistic logic. To prove the converse, let

$$\begin{aligned} Y &\stackrel{d}{\equiv} (\forall x \leq |t|)(\exists y \leq s)A(x,y) \\ Z(u) &\stackrel{d}{\equiv} (\exists w \leq Bound(s^*, t))[Seq(w) \wedge (\forall x \leq |t|)(x \leq u \rightarrow . A(x, \beta(x,w)) \wedge \beta(x,w) \leq s)]. \end{aligned}$$

We first show that $Y \rightarrow Z(0)$, that is,

$$Y \rightarrow (\exists w \leq Bound(s,t))[Seq(w) \wedge A(0, \beta(0,w)) \wedge \beta(0,w) \leq s].$$

Assuming Y, we have $A(0,b)$ for some $b \leq s[0/x]$. Let w be $0 * b$. By 2.2 (A, B, C), $Seq(w)$, $Len(w) = 1$; by 2.2 (F), $\beta(0,w) = b$, and by 2.2 (H) and T205, $w \leq Bound(s^*, t)$ so the proof of $Y \rightarrow Z(0)$ is complete.

Secondly, we prove $Y \wedge Z(u) \wedge (u < |t|) \rightarrow . \ Z(Su)$. Assuming the antecedent, we have:

$$(\exists w_1 \leq Bound(s^*, t))(\forall x \leq |t|)(x \leq u \rightarrow . A(x, \beta(x, w_1)) \wedge \beta(x, w_1) \leq s).$$

If $u < |t|$ then by Y we have $A(Su, b)$ for some $b \leq s[Su/x]$. Define w_2 to be $w_1 * b$. By 2.2, $Seq(w_2)$ and $(\forall x \leq |t|)(x \leq Su \rightarrow . \ A(x, \beta(x, w_2)) \wedge \beta(x, w_2) \leq s)$. By 4.6, we can apply the appropriate form of Σ_1^b-$LIND$ to conclude $Y \rightarrow Z(|t|)$. □

4.11 Lemma *Every Σ_1^{b+} formula of IPV is equivalent in IPV to a formula of the form $(\exists y \leq t)(u = v)$.*

Proof: First we claim that for each formula A of the form $(\exists y_1 \leq t_1)(\exists y_2 \leq t_2)u = v$ there is a formula of the form $(\exists z \leq s)u' = v'$ so that

$$IPV \vdash (\exists y_1 \leq t_1)(\exists y_2 \leq t_2)u = v \leftrightarrow (\exists z \leq s)u' = v'.$$

By the definition of bounded quantifiers (§1), y_i does not occur in t_i, $i = 1, 2$. If y_2 occurs in t_1 then we may avoid this occurrence by renaming the variable in the second quantifier. However, y_1 may occur in t_2. To eliminate this occurrence, we appeal to T205 in the Appendix and 4.3 to find a term s_2 whose variables are among those of t_2 such that $\vdash t_2 \leq s_2$ and $\vdash y_1 \leq t_1 \rightarrow s_2 \leq s_2[t_1/y_1]$. Then $IPV \vdash A \leftrightarrow (\exists y_1 \leq t_1)(\exists y_2 \leq s_2[t_1/y_1])(y_2 \leq t_2 \wedge u = v)$. Thus we may assume that in the formula A neither y_1 or y_2 occur in t_1 or t_2.

Now using the pairing function $< x, y >$ (D99) and projection functions Π^2_1, Π^2_2 (D115), we let $B \stackrel{d}{\equiv} (\exists z \leq < t_1, t_2 >)(\Pi^2_1(z) \leq t_1 \wedge \Pi^2_2(z) \leq t_2 \wedge u' = v')$ where u', v' are obtained from u, v by the substitution $[\Pi^2_1 z, \Pi^2_2 z / y_1, y_2]$. That $IPV \vdash A \leftrightarrow B$ follows from T116 and the theorem $(x_1 \leq y_1 \wedge x_2 \leq y_2) \rightarrow < x_1, x_2 > \leq < y_1, y_2 >$, which follows by NP-induction from T101. The above claim follows by 4.4.

Now 4.11 is proved by induction on the complexity of the Σ^{b+}_1 formula A. If A is atomic, the result follows by axiom 4.1.2. If A is $(\exists y \leq t)B$, the result follows from the induction hypothesis applied to B and the claim above. If A is $B \wedge C$ then by the induction hypothesis it can be written as

$$(\exists y \leq t)(\exists z \leq s)(u = v \wedge r = w),$$

which is equivalent to a formula of required form by 4.4 and the claim above. Similarly if A is $B \vee C$. Finally, if A is $(\forall x \leq |t|)B$, then it is equivalent to a formula of the required form by the induction hypothesis applied to B, and 4.10 and 4.4. □

4.12 Theorem *IPV is a conservative extension of* IS^1_2.

Proof: All $BASIC$ axioms of IS^1_2 are provable in IPV by 4.7. By 4.11, any Σ^{b+}_1 formula of IPV is equivalent to a formula of the form $(\exists y \leq t)u = v$, so the Σ^{b+}_1-$PIND$ scheme follows from the NP-induction scheme. Thus all theorems of IS^1_2 are theorems of IPV.

For the other direction, we need to show that the axioms of IPV are derivable in an extension of IS^1_2 by Σ^{b+}_1-definitions. This extension is conservative by 2.1.

Each basic function symbol of PV is either a function symbol of IS^1_2, or has a simple defining formula in IS^1_2. Thus

$$\begin{array}{l}
s_0(x) = 2 \cdot x \\
s_1(x) = 2 \cdot x + 1 \\
y = Parity(x) \leftrightarrow x = y + 2 \cdot \lfloor \frac{1}{2} x \rfloor \\
w = Cond(x, y, z) \leftrightarrow (x = 0 \wedge y = w) \vee (\neg(x = 0) \wedge z = w) \\
Pad(x, y) = x \cdot (y \# 1) \\
z = x \dot{-} y \leftrightarrow (\exists w \leq x)(|w| \leq |y| \wedge x = Pad(z, y) + w)
\end{array}$$

The PV axioms (0)–(6b) are readily derived in IS_2^1 extended by these defining axioms. The defining axiom $[\lambda x_1 \cdots x_n.t](x_1, \ldots, x_n) = t$ takes care of axiom scheme (7).

To deal with (8), let us assume that g, h, k are n-place, $n+2$-place and $n+1$-place function symbols in a theory T which is an extension of IS_2^1 by Σ_1^{b+} definitions of function symbols. By 2.4, we may assume that T is sufficiently strong. Then by 2.6, there is an $n+1$-place function symbol f in an extension of T by Σ_1^{b+} definitions for which the theorems:

$$f(0, \vec{b}) = g(\vec{b})$$
$$a = 0 \lor f(a, \vec{b}) = Lmin[h(a, \vec{b}, f(\lfloor \tfrac{1}{2}a \rfloor, \vec{b})), k(a, \vec{b})]$$

are provable. Then the theorem:

$$f(a, \vec{b}) = \begin{cases} \text{If } a = 0 \text{ then } g(\vec{b}) \\ \quad \text{else} \\ Cond(t \dot{-} k(a, \vec{b}), t, k(a, \vec{b})) \end{cases}$$

where $t \overset{d}{\equiv} h(a, \vec{b}, f(\lfloor \frac{1}{2}a \rfloor, \vec{b}))$ is derivable by the basic properties of $Cond$, $\dot{-}$ and $Lmin$. Thus f satisfies the axiom scheme (8) of PV.

Hence all axioms of PV are derivable in a suitable extension T of IS_2^1 by Σ_1^{b+}-definitions. The remaining axioms and axiom schemes (2)–(5) of IPV are derivable in this same extension T. $\Box$

5 The Typed λ Calculus

In this section we introduce the basic concepts of the typed λ-calculus which will be used in later sections. See [23], and [26] for more details.

5.1 Definition *The class of* types *is defined as follows:*

1) *0 is a type.*

2) *$(\sigma \to \tau)$ is a type if σ and τ are types.*

The intention is that each natural number has type 0, and each function from the set of objects of type σ to the set of objects of type τ has type $(\sigma \to \tau)$.

We use the notation $\tau_1 \to \tau_2 \to \cdots \to \tau_k$ to mean $(\tau_1 \to (\tau_2 \to \cdots \to (\tau_{k-1} \to \tau_k) \cdots))$ (association to the right).

5.2 Proposition *For each type $\tau \neq 0$ there is a unique sequence $\tau_1, \ldots, \tau_k$ $(k \geq 1)$ of types such that*

5.3
$$\tau = \tau_1 \to \tau_2 \to \cdots \to \tau_k \to 0$$

Proof: Induction on the definition of type. Uniqueness uses the standard argument showing the unique readability of parenthesized formulas. □

A function F with the type τ given in 5.3 is a function from objects of type τ_1 to objects of type $\tau_2 \to \cdots \to \tau_k \to 0$. However it is often useful to think of F as a function of k variables $X_1, \ldots, X_k$, with X_i ranging over objects of type τ_i. In this case $F(X_1, \ldots, X_k) = F(X_1) \cdots (X_k)$ is a natural number. The case in which $\tau_i = 0$ $(1 \leq i \leq k)$ is especially interesting, since from our point of view such are the types of the number-theoretic functions of PV and IPV.

5.4 Definition *If $\tau = 0 \to 0 \to \cdots \to 0$, with at least one "$\to$", then τ is a* type *1 type.*

5.5 Definition *Let $\mathcal{F}$ be a collection of function symbols (of any types). The collection of λ-terms generated from $\mathcal{F}$, denoted $\Lambda(\mathcal{F})$, is defined as follows:*

(1) *There are infinitely many variables $X^\sigma, Y^\sigma, Z^\sigma, \ldots$ for each type σ, and each such variable is a term of type σ;*

(2) *Every function symbol in $\mathcal{F}$ of type σ is a term of type σ;*

(3) *If T is a term of type τ and X is a variable of type σ, then $(\lambda X.T)$ is a term of type $(\sigma \to \tau)$ (an* abstraction*);*

(4) If S is a term of type $(\sigma \to \tau)$ and T is a term of type σ, then (ST) is a term of type τ (an application*).*

We use lower case letters x, y, z, and t, u, v for type 0 variables and terms, respectively, and upper case letters X, Y, Z and S, T, U for variables and terms of arbitrary types. Lower case letters f, g, h stand for type 1 constants and upper case letters $F, G, H, \ldots$ for constants of arbitrary type. If F has type $\sigma_1 \to \sigma_2 \to \cdots \to \sigma_n \to \tau$ and T_i has type τ_i, $1 \leq i \leq n$, we will sometimes write $F(T_1, \ldots, T_n)$ or $F(\vec{T})$ instead of $(\cdots(FT_1)\cdots T_n)$. (This is consistent with our comments after 5.3.) We abbreviate $(\lambda X_1.(\lambda X_2.(\cdots(\lambda X_n.T)\cdots)))$ by $\lambda X_1 \cdots X_n.T$ or $\lambda \vec{X}.T$.

5.6 Definition *If S and T are terms of types σ and τ, and X is a variable of type τ, then $S[T/X]$ is defined to be the term of type σ which results from S by substituting T for all free occurrences of X in S. We assume that bound variables are changed if necessary to avoid clashes (see [23] p. 7).*

5.7 Definition *A term of the form $(\lambda X.S)T$ is said to be a β* redex*, and the term $S[T/X]$ is its* contractum*. A term of the form $(\lambda X.TX)$ such that X has no free occurrence in T is said to be an η* redex *and the term T is its* contractum*. A term S* contracts *to a term T if T is obtained from S by replacing a β or η redex in S by its contractum. A term S* reduces to *a term T if T is obtained from S by a finite sequence of contractions and changes of bound variable.*

5.8 Definition *A term T is in* normal form *if T has no redex.*

5.9 Church-Rosser Theorem *If S reduces to T and S reduces to T' then there is a term T'' such that both T and T' reduce to T''.*

5.10 Strong Normalization Theorem *Every sequence of contractions of a term T terminates with a term in normal form.*

The proofs of the two preceding theorems can be found, for example, in [37], 2.2, or [23], appendix.

These theorems justify the following:

5.11 Definition *$NF(T)$ denotes a particular term in normal form such that T reduces to $NF(T)$. (The choice of $NF(T)$ is unique up to changes in bound variables).*

5.12 Theorem *Let $\mathcal{F}$ be a collection of function symbols of types 0 and 1. Let t be a term of type 0 in $\Lambda(\mathcal{F})$ in which all free variables have type 0. Then $NF(t)$ has no occurrence of λ, and hence all subterms have type 0 or type 1.*

Proof: If not, then $NF(t)$ has a left-most subterm of the form $\lambda X.S$. This subterm cannot coincide with $NF(t)$ (since t has type 0), it cannot occur in the context $(\lambda X.S)U$ (since $NF(t)$ is in normal form), and it cannot occur in the context $\lambda Y.(\lambda X.S)$ (since $\lambda X.S$ is left-most). Hence it occurs in the context $U(\lambda X.S)$. But the term U cannot contain λ (since $\lambda X.S$ is left-most) and it cannot be of types 0 or 1 (since $\lambda X.S$ is not of type 0). This contradicts our assumption that all function symbols in $\mathcal{F}$ have type 0 or type 1. $\square$

6 The System PV^ω

We now extend the system PV to a system PV^ω by adding variables and functions of all (finite) types. We shall prove that the extension is conservative (Corollary 6.18). We need only one constant in addition to those in PV, namely the *recursor* $\mathcal{R}$. This is used to introduce higher type functions by limited recursion on notation, and is the polynomial time analog of the constant R_0 in [37], p. 40.

6.1 Definition *The* function symbols *of PV^ω are as follows:*

a) *For each $n \geq 0$ each n-ary function symbol f of PV is a function symbol of PV^ω of type $0 \to 0 \cdots \to 0$ ($n+1$ zeroes).*

b) *The constant $\mathcal{R}$ is a function symbol of PV^ω of type*

$$0 \to (0 \to 0 \to 0) \to (0 \to 0) \to 0 \to 0.$$

6.2 Definition *The* terms *of PV^ω comprise the set $\Lambda(\mathcal{F})$ (see Definition 5.5), where $\mathcal{F}$ is the set of function symbols given in 6.1.*

Note that the function symbols of PV are *not* λ-terms in PV^ω, but are taken as primitive function symbols in PV^ω. By Definition 3.1.3 all terms of PV are type 0 terms of PV^ω.

In the sequel we use the following:

6.3 Notation *$S\{T\}$ refers to a term with a distinguished occurrence of a subterm T. Then $S\{U\}$ means $S\{T\}$ with the indicated occurrence of T replaced by U. In general T and U may have free variables which become bound in $S\{T\}$ and $S\{U\}$.*

6.4 Definition *The* formulas *of PV^ω are all equations $s = t$, where s, t are type 0 terms of PV^ω.*

6.5 Definition *The* axioms *of PV^ω consist of all axioms for PV (i.e. the defining equations for all function symbols for PV), together with the axiom $(HTLRN)$ (higher type limited recursion on notation) and the schemes (α), (β), and (η).*

$(HTLRN)$

$$\mathcal{R}(y, Z, W, x) = \begin{cases} \textit{if } x = 0 \textit{ then } y \\ \quad \textit{else} \\ Cond(t \dot{-} W(x), t, W(x)) \end{cases}$$

where

$$t \stackrel{d}{\equiv} Z(x, \mathcal{R}(y, Z, W, \lfloor \tfrac{1}{2}x \rfloor)),$$

and the variables

$$x, y \text{ have type } 0,$$
$$Z \text{ has type } 0 \to 0 \to 0,$$
$$W \text{ has type } 0 \to 0.$$

(α) $s\{\lambda X.T\} = s\{\lambda Y.T[Y/X]\}$,
provided Y does not occur free in T.

(β) $s\{(\lambda X.T)U\} = s\{T[U/X]\}$,

(η) $s\{(\lambda X.TX)\} = s\{T\}$,
provided X does not occur free in T.

6.6 Definition *The rules of PV^ω are the following:*

$R1^\omega$ $s = t \vdash t = s$

$R2^\omega$ $s = t, t = u \vdash s = u$

$R3^\omega$ $s = t \vdash u\{s\} = u\{t\}$

$R4^\omega$ $s = t \vdash s[T/X] = t[T/X]$

$R5^\omega$ *(Induction on notation): Same as R5 for PV, except terms are now terms of PV^ω.*

Here s, t, u, T are any terms of PV^ω, and s, t, u have type 0.

This completes the formal description of PV^ω. We note that PV^ω is an extension of PV. Further, if t is a λ-free type 0 term of PV^ω whose variables have type 0 and whose function symbols are in PV, then t is a term of PV.

We note that the constant $\mathcal{R}$ would allow us to dispense with the PV function symbols $R[g, h, k]$ if we did not insist that PV^ω be an extension of PV. In fact we could replace $R[g, h, k](x, \vec{y})$ by $\mathcal{R}(s, T, U, x)$, where $s \overset{d}{\equiv} g(\vec{y})$, $T \overset{d}{\equiv} \lambda x'z.h(x', \vec{y}, z))$, and $U \overset{d}{\equiv} \lambda x'.k(x', \vec{y})$.

The standard interpretation of terms of PV^ω is higher type functions on the natural numbers. We do not use this interpretation here, but the complexity theory of such functions is developed in [15].

6.7 Proposition

$$PV^\omega \vdash s\{T\} = s\{NF(T)\}$$
$$PV^\omega \vdash s = NF(s)$$

Proof: Immediate from axiom schemes $(\alpha), (\beta), (\gamma)$, and rule $R2^\omega$. ($NF(T)$ is defined in 5.11.) $\Box$

The propositional connectives $\wedge, \vee, \neg, \supset$ are introduced in PV^ω just as in PV, and classical truth functional reasoning can be carried out.

We now introduce a term $Cond_\tau(x, Y, Z)$ for each type τ, where x, Y, and Z are variables of types $0, \tau$, and τ.

6.8 Definition *Let $\tau = \tau_1 \to \cdots \to \tau_n \to 0$ ($n \geq 0$) and $\vec{W} = (W_1, \ldots, W_n)$, where W_i is a variable of type τ_i, $1 \leq i \leq n$. Then*

$$Cond_\tau(x, Y, Z) \stackrel{d}{\equiv} \lambda\vec{W}.Cond(x, Y(\vec{W}), Z(\vec{W})).$$

In particular,

$$Cond_0(x, y, z) \stackrel{d}{\equiv} Cond(x, y, z).$$

Intuitively, $Cond_\tau(0, X, Y) = X$ and $x \neq 0 \to Cond_\tau(x, Y, Z) = Z$. However these equations cannot be expressed in PV^ω for $\tau \neq 0$, so instead we prove the following two propositions.

6.9 Lemma

$$PV^\omega \vdash Cond_0(x, t\{S\}, t\{T\}) = t\{Cond_\tau(x, S, T)\}.$$

Proof: We use rule $R5^\omega$ (induction) applied to x. We may assume that x has no free occurrence in $t\{S\}$ and $t\{T\}$ by renaming the x indicated in 6.9 with a new variable x', and then applying $R4^\omega$ to change x' back to x.

When $x = 0$, both sides of 6.9 equal $t\{S\}$ in PV^ω, by the defining equations for $Cond_0$ ($= Cond$), rules $R3^\omega$ and $R4^\omega$, and n applications of the axiom scheme (η). The induction steps are handled similarly. $\Box$

The next result uses the notation introduced in D9 of §3.

6.10 Theorem *a) $PV^\omega \vdash t\{Cond_\tau(0, S, T)\} = t\{S\}$.*
b) $PV^\omega \vdash x \neq 0. \supset .t\{Cond_\tau(x, S, T)\} = t\{T\}$.

Proof: a) comes from 6.9 with $x = 0$.

For b), we have from T11

$$PV \vdash Cond_0(x, z, Cond_0(x, y, z)) = z,$$

hence

$$PV^\omega \vdash Cond_0(x, t\{T\}, Cond_0(x, t\{S\}, t\{T\})) = t\{T\}$$

and by 6.9

$$PV^\omega \vdash Cond_0(x, t\{T\}, t\{Cond_\tau(x, S, T)\}) = t\{T\}$$

which is b). $\square$

Theorem 6.10 is especially useful in conjunction with the conditional proof principle, which generalizes DR16.

6.11 Theorem *(Conditional Proof Principle). If $PV^\omega \vdash t[0/x] = u[0/x]$ and $PV^\omega \vdash x \neq 0. \supset .t = u$ then $PV^\omega \vdash t = u$.*

Proof: Same as DR16. $\square$

Next we show that $HTLRN$ can be generalized to simultaneous recursion in PV^ω. For fixed $n \leq 2$ we use vector notation such as $\vec{T}$ for $T_1, \ldots, T_n$ and $\vec{U}(\vec{T})$ for $U_1(\vec{T}), \ldots, U_n(\vec{T})$. Recall from T116 (Appendix) that the tupling functions $<>$ and projection functions Π_i^n satisfy

$$\Pi_i^n(< \vec{x} >) = x_i \quad (1 \leq i \leq n).$$

6.12 Theorem *(Simultaneous Recursion). For each $n \geq 2$ there are closed terms $\rho_1, \ldots, \rho_n$ such that for $1 \leq i \leq n$*

$$PV^\omega \vdash \rho_i(\vec{y}, \vec{Z}, \vec{W}, x) = \begin{cases} \textit{if } x = 0 \textit{ then } y_i \\ \quad \textit{else} \\ Cond(< \vec{t} > \dot{-} < \vec{W}(x) >, t_i, W_i(x)) \end{cases}$$

where

$$t_i \stackrel{d}{\equiv} Z_i(x, \vec{\rho}(\vec{y}, \vec{Z}, \vec{W}, \lfloor \tfrac{1}{2} x \rfloor))$$

and the variables

x, y_i *have type* 0
Z_i *has type* $0 \to 0 \to \cdots \to 0$ *($n+2$ zeroes)*
W_i *has type* $0 \to 0$.

Proof: For $1 \leq i \leq n$ let

$$\rho_i \stackrel{d}{\equiv} \lambda \vec{y} \vec{Z} \vec{W} x. \pi_i^n(\mathcal{R}(s, T, U, x))$$

where

$$s \stackrel{d}{\equiv} < \vec{y} >,$$

$$T \stackrel{d}{\equiv} \lambda x' z. < \vec{Z}(x', \pi_1^n(z), \cdots, \pi_n^n(z)) >,$$

and

$$U \stackrel{d}{\equiv} \lambda x'. < \vec{W}(x') >.$$

Then from the axioms (β) and $(HTLRN)$ and equality reasoning we have in PV^ω

$$\mathcal{R}(s,T,U,x) = \begin{cases} \text{if } x = 0 \text{ then } < \vec{y} > \\ \quad \text{else} \\ Cond(< \vec{t} > \dot{-} < \vec{W}(x) >, < \vec{t} >, < \vec{W}(x) >) \end{cases}$$

where

$$< \vec{t} > \stackrel{d}{\equiv} < \vec{Z}(x, \vec{\rho}(\vec{y}, \vec{Z}, \vec{W}, \lfloor \tfrac{1}{2} x \rfloor)) > .$$

The theorem now follows easily from Lemma 6.9 applied with $\tau = 0$ and $t \stackrel{d}{\equiv} \Pi_i^n$. $\square$

The remainder of this section is devoted to proving that PV^ω is conservative over PV (Corollary 6.18). To do this, we define a transformation $t \rightsquigarrow \{t\}^{PV}$ (Definition 6.14) where t is a type 0 term of PV^ω which is zero-order open (i.e. all free variables have type 0), and $\{t\}^{PV}$ is an equivalent term of PV. If t has no occurrence of $\mathcal{R}$, then $\{t\}^{PV}$ is $NF(t)$, which is a term of PV by Theorem 5.12. In general, subterms t such that $\mathcal{R}$ occurs in $NF(t)$ must be replaced by subterms with appropriate occurrences of $R[g,h,k]$. Although this is easily done, there seems to be no canonical way to do it. We wish to choose a method which is transparent to substitution (Lemma 6.15).

For example, the term

$$\mathcal{R}(y, \lambda x'z.x' + y, \lambda x'.x' + y, x)$$

contains three free occurrences of y. We might translate this as

$$R[[\lambda y'.y'], [\lambda x'y'z.x' + y'], [\lambda x'y'.x' + y']](x,y),$$

but instead we treat each occurrence separately, and (abbreviating $y_1y_2y_3$ by $\vec{y}$) we translate it as

$$R[[\lambda \vec{y}.y_1], [\lambda x'\vec{y}z.x' + y_2], [\lambda x'\vec{y}.x' + y_3]](x,y,y,y).$$

6.13 Definition *A term T is* zero-order open *if all free variables of T have type 0. A subterm U of a term T is* free in T *if no variable has an occurrence which is free in U and bound in T.*

Note that U is free in T iff there is a term T' with a single free occurrence of a variable Y such that $T \equiv T'[U/Y]$.

6.14 Definition *The transformation $t \rightsquigarrow \{t\}^{PV}$ takes a zero-order open type 0 term t of PV^ω to an equivalent term $\{t\}^{PV}$ of PV. The following three cases partition the set of such terms t (we sometimes write t^{PV} for $\{t\}^{PV}$, and $\vec{u}^{PV}$ for $u_1^{PV}, \ldots, u_n^{PV}$).*

Case 1. *$NF(t)$ is a term of PV. Then*

$$\{t\}^{PV} \stackrel{d}{\equiv} NF(t).$$

Case 2. *$NF(t) \equiv f(t_1, \ldots, t_n)$, where f is a PV function symbol but not all of $t_1, \ldots, t_n$ are PV terms. Then*

$$\{t\}^{PV} \stackrel{d}{\equiv} f(\{t_1\}^{PV}, \ldots, \{t_n\}^{PV}).$$

Case 3. *$NF(t) \equiv \mathcal{R}(s, T, U, v)$. Then*

$$\{t\}^{PV} \stackrel{d}{\equiv} R[g, h, k](v^{PV}, \vec{u}^{PV})$$

where $u_1 \stackrel{d}{\equiv} s$, and $u_2, \ldots, u_n$ are the maximal type 0 subterms, listed in order and not necessarily distinct, occurring free (see Definition 6.13) in T, U, and g, h, k are PV function symbols defined as follows. Let $y_1, \ldots, y_n$ be distinct new variables, and let T', U' be terms whose only type 0 subterms occurring free are the y_i's such that

$$T \equiv T'[u_2/y_2, \ldots, u_n/y_n]$$

and

$$U \equiv U'[u_2/y_2, \ldots, u_n/y_n].$$

Then

$$\begin{aligned} g &\equiv [\lambda \vec{y}.y_1] \\ h &\equiv [\lambda x\vec{y}z.\{T'(x,z)\}^{PV}] \\ k &\equiv [\lambda x\vec{y}.\{U'(x)\}^{PV}], \end{aligned}$$

where x and z are new variables. $\square$

Note that $t^{PV} \equiv t$, if t is a term of PV.

It is not hard to check that the normal forms of the arguments of all inductive applications of $\{\cdot\}^{PV}$ in Definition 6.14 are shorter than $NF(t)$. Hence the definition is well-founded.

An important property of the transformation $\{\cdot\}^{PV}$ is its transparency under substitution of type 0 terms.

6.15 Lemma *$\{t[u/x]\}^{PV} \equiv t^{PV}[u^{PV}/x]$, for all terms t, u in the domain of $\{\cdot\}^{PV}$, and all type 0 variables x.*

Proof: Induction on the length of $NF(t[u/x])$. First note that by the Church-Rosser theorem 5.8), $NF(t[u/x]) \equiv NF(t)[NF(u)/x]$. Applying the above definition to $t[u/x]$, the lemma is thus clear for Case 1, and clear for Case 2 by the induction hypothesis. Now suppose Case 3 applies to $t[u/x]$. Then $NF(t[u/x]) \equiv \mathcal{R}(\hat{s}, \hat{T}, \hat{U}, \hat{v})$, where $NF(t) \equiv \mathcal{R}(s, T, U, v)$ and $\hat{s}, \hat{T}, \hat{U}, \hat{v}$ result from s, T, U, v by the substitution $[NF(u)/x]$. The crucial observation is that the number n of variables $y_1, \ldots, y_n$ and the terms T', U' are the same whether Case 3 is applied to t or $t[u/x]$, and thus g, h, and k are the same. The lemma now follows for this case by the induction hypothesis. □

6.16 Theorem *$PV^\omega \vdash t = t^{PV}$, for each t in the domain of $\{\cdot\}^{PV}$.*

Proof: Induction on the definition of $\{\cdot\}^{PV}$. Case 1 follows from 6.7. For case 3, $(HTLRN)$ and the defining equations for $R[g, h, k]$ are required, as well as induction on notation. □

6.17 Theorem *If $PV^\omega \vdash t = u$, then $PV \vdash t_1^{PV} = u_1^{PV}$, where $t_1 = u_1$ is any zero-order open substitution instance of $t = u$.*

6.18 Corollary *PV^ω is a conservative extension of PV.* □

Proof of 6.17: Induction on the PV^ω proof of $t = u$. If $t = u$ is an axiom of PV, then by Lemma 6.15 $t_1^{PV} = u_1^{PV}$ is an instance of that axiom and a theorem of PV by rule $R4$. If $t = u$ is an instance of $(\alpha), (\beta)$, or (η), then $NF(t_1) \equiv NF(u_1)$ so $t_1^{PV} = u_1^{PV}$ is an instance of $x = x$.

The only remaining axiom is $(HTLRN)$. In this case, by Lemma 6.15 it suffices to prove in PV

$$\{\mathcal{R}(s, T, U, v)\}^{PV} = \begin{cases} \text{if } v^{PV} = 0 \text{ then } s^{PV} \\ \quad \text{else} \\ Cond(t \dot{-} \{U(v^{PV})\}^{PV}, t, \{U(v^{PV})\}^{PV}) \end{cases}$$

where

$$t \overset{d}{\equiv} \{T(v^{PV}, \{\mathcal{R}(s, T, U, \lfloor \tfrac{1}{2} v \rfloor)\}^{PV})\}^{PV}.$$

By definition 6.14, the left hand side is $R[g, h, k](v^{PV}, \vec{u}^{PV})$, which by the defining equations for $R[g, h, k], g, h$, and k and Lemma 6.15 provably equals in PV

$$\begin{cases} \text{if } v^{PV} = 0 \text{ then } s^{PV} \\ \quad \text{else} \\ Cond(t' \dot{-} \{U(v^{PV})\}^{PV}, t', \{U(v^{PV})\}^{PV}) \end{cases}$$

where

$$t' \stackrel{d}{\equiv} \{T(v^{PV}, R[g,h,k](\lfloor \tfrac{1}{2} v^{PV} \rfloor, \vec{u}^{PV}))\}^{PV}.$$

Again by Definition 6.14,

$$\{\mathcal{R}(s, T, U, \lfloor \tfrac{1}{2} v \rfloor)\}^{PV} \equiv R[g,h,k](\lfloor \tfrac{1}{2} v \rfloor^{PV}, \vec{u}^{PV}).$$

Thus $t \equiv t'$. This completes the case of $(HTLRN)$.

Now suppose the last line in the PV^ω proof follows from previous lines by one of the five rules $R1^\omega, \ldots, R5^\omega$. In the case of $R1^\omega$ and $R2^\omega$, Theorem 6.17 follows immediately by the induction hypothesis and the PV rules $R1$ and $R2$, respectively. In the case of $R4^\omega$, the induction hypothesis alone suffices, and no PV rule is required. In the case of $R5^\omega$ (Induction on Notation), the theorem follows from the induction hypothesis, Lemma 6.15 and the PV rules $R4$ and $R5$.

The case of $R3^\omega$ requires more work.

6.19 Definition *Let S, T be terms of PV^ω. We say that (S', T') is an* instance *of (S,T) iff there is a common substitution of terms of free variables which yields S' from S and T' from T, and (S', T') is a* zero-order open instance *of (S,T) if in addition S' and T' are zero-order open (see 6.13).*

6.20 Definition *Suppose S, T are terms of type $\sigma_1 \to \sigma_2 \to \cdots \to \sigma_n \to 0$. We write $S \overset{PV}{\sim} T$ iff $PV \vdash \{S'(\vec{\phi})\}^{PV} = \{T'(\vec{\phi})\}^{PV}$ for all zero-order open instances (S', T') of (S,T), and all $\vec{\phi} \equiv \phi_1, \ldots, \phi_n$ such that ϕ_i is zero-order open of tpe σ_i, $1 \le i \le n$.*

The following lemma suffices to handle rule $R3^\omega$.

6.21 Lemma *If $S \overset{PV}{\sim} T$ then $U\{S\} \overset{PV}{\sim} U\{T\}$ (see 6.3 for notation).*

Proof: Induction on the relative depth of T in $U\{T\}$. The following three lemmas suffice.

6.22 Lemma *If $S \overset{PV}{\sim} T$, then $(\lambda X.S) \overset{PV}{\sim} (\lambda X.T)$.*

6.23 Lemma *If $S \overset{PV}{\sim} T$, then $(SU) \overset{PV}{\sim} (TU)$.*

6.24 Lemma *If $S \overset{PV}{\sim} T$, then $(US) \overset{PV}{\sim} (UT)$.*

Proof of 6.22: It suffices to show

$$PV \vdash \{(\lambda X.S')(\psi, \vec{\phi})\}^{PV} = \{(\lambda X.T')(\psi, \vec{\phi})\}^{PV}$$

for each zero-order open instance (S', T') of (S, T) in which X is not substituted for, and for all appropriate zero-order open terms $\psi, \vec{\phi}$. Since $\{\cdot\}^{PV}$ operates on normal forms, this is equivalent to showing

$$PV \vdash \{S'[\psi/X](\vec{\phi})\}^{PV} = \{T'[\psi/X](\vec{\phi})\}^{PV},$$

which holds by definition of $S \overset{PV}{\sim} T$.

Proof of 6.23: It suffices to show

$$PV \vdash \{(S'U')(\vec{\phi})\}^{PV} = \{(T'U')(\vec{\phi})\}^{PV}$$

for all zero-order open instances (S', T') of (S, T) and U' of U, and all appropriate $\vec{\phi}$. This follows immediately by definition of $S \overset{PV}{\sim} T$.

The proof of 6.24 is inspired by ([37], p. 112), and requires the introduction of an equivalence relation $\approx$ on zero-order open terms, and properties G, G^* (G for "good") of terms.

6.25 Definition *Let S, T be zero-order open terms of PV^ω of type $\sigma_1 \to \sigma_2 \to \cdots \to \sigma_n \to 0$. Then $S \approx T$ iff $PV \vdash \{S(\vec{\phi})\}^{PV} = \{T(\vec{\phi})\}^{PV}$ for all $\vec{\phi} \equiv \phi_1, \ldots, \phi_n$ such that ϕ_i is zero-order open of type σ_i, $1 \leq i \leq n$.*

Thus for S, T any terms of PV^ω, $S \overset{PV}{\sim} T$ iff $S' \approx T'$ for all zero-order open instances (S', T') of (S, T).

6.26 Definition *The properties $G_\sigma(T)$ are defined by induction on the type σ.*

1) $G_0(t)$ iff t is zero-order open of type 0.

2) If $\sigma \equiv \sigma_1 \to \sigma_2 \cdots \to \sigma_n \to 0$, then $G_\sigma(T)$ iff T is zero-order open of type σ and $T(\vec{\phi}) \approx T(\vec{\psi})$ for all $\vec{\phi} \equiv \phi_1, \ldots, \phi_n$, and $\vec{\psi} \equiv \psi_1, \ldots, \psi_n$ such that $G_{\sigma_i}(\phi_i)$, $G_{\sigma_i}(\psi_i)$, and $\phi_i \approx \psi_i$, $1 \leq i \leq n$.

We write $G(T)$ if $G_\sigma(T)$ for some σ.

6.27 Definition *For any term T of PV^ω, $G^*(T)$ iff $G(T[\vec{\phi}/\vec{X}])$ and $T[\vec{\phi}/\vec{X}] \approx T[\vec{\psi}/\vec{X}]$ for all $\vec{\phi} \equiv \phi_1, \ldots, \phi_n$ and $\vec{\psi} \equiv \psi_1, \ldots, \psi_n$ such that 1) $G(\phi_i), G(\psi_i), \phi_i \approx \psi_i$, $1 \leq i \leq n$, and 2) $T[\vec{\phi}/\vec{X}]$ is zero-order open.*

Lemma 6.24 and Theorem 6.17 now follow from the next lemma.

6.28 Lemma *$G^*(T)$ for all terms T of PV^ω.*

Proof: Induction on the definition 5.4 of terms T.

1) $G^*(X)$ clearly holds for all variables X.

2) $G^*(f)$, f any PV function symbol: It suffices to show $G(f)$, which is, by Cases 1 and 2 of Definition 6.14,

$$PV \vdash f(\phi_1^{PV}, \ldots, \phi_n^{PV}) = f(\psi_1^{PV}, \ldots, \psi_n^{PV}),$$

where $PV \vdash \phi_i^{PV} = \psi_i^{PV}$, $1 \leq i \leq n$. This follows from PV rule $R3$.

3) $G^*(\mathcal{R})$. It suffices to show $G(\mathcal{R})$, i.e. $\mathcal{R}(s, T, U, v) \approx \mathcal{R}(\hat{s}, \hat{T}, \hat{U}, \hat{v})$, assuming all terms in question satisfy G, and $s \approx \hat{s}, T \approx \hat{T}$, $U \approx \hat{U}$, and $v \approx \hat{v}$. Referring to Case 3 of Definition 6.14, it suffices (since $v \approx \hat{v}$) to show that

$$PV \vdash R[g, h, k](x, u_1^{PV}, \ldots, u_n^{PV}) = R[\hat{g}, \hat{h}, \hat{k}](x, \hat{u}_1^{PV}, \ldots, \hat{u}_m^{PV})$$

where the right hand side is $\{\mathcal{R}(\hat{s}, \hat{T}, \hat{U}, x)\}^{PV}$ (note that in general $n \neq m$). The above equation is proved in PV by induction on notation on x, using the defining equations for $R[\ ]$ and the PV identities $g(\vec{u}^{PV}) = \hat{g}(\vec{\hat{u}}^{PV})$, $h(x, \vec{u}^{PV}, z) = \hat{h}(x, \vec{\hat{u}}^{PV}, z)$, and $k(x, \vec{u}^{PV}) = \hat{k}(x, \vec{\hat{u}}^{PV})$. The identity for g is immediate from $s \approx \hat{s}$. The identity for h follows (using 6.15) from $\{T(x, z)\}^{PV} = \{\hat{T}(x, z)\}^{PV}$, which holds since $T \approx \hat{T}$. Similarly the identity for k holds in PV because $U \approx \hat{U}$.

4) $G^*(TU)$, assuming $G^*(T)$ and $G^*(U)$: Referring to the definition of G^*, let $T' \equiv T[\vec{\phi}/\vec{X}]$, $T'' \equiv T[\vec{\psi}/\vec{X}]$, $U' \equiv U[\vec{\phi}/\vec{X}]$, and $U'' \equiv U[\vec{\psi}/\vec{X}]$. We must show $G(T'U')$ and $T'U' \approx T''U''$. Since $T' \approx T''$, $G(U')$ and $U' \approx U''$ and $G(T'')$, we have $T'(U', \vec{\eta}) \approx T''(U', \vec{\eta}) \approx T''(U'', \vec{\eta})$, so $T'U' \approx T''U''$. Since also $G(T')$, we have (if $\vec{\eta} \approx \vec{\mu}$) $T'(U', \vec{\eta}) \approx T'(U', \vec{\mu})$, so $G(T'U')$.

5) $G^*(\lambda X.T)$, assuming $G^*(T)$. Let $T' \equiv T[\vec{\phi}/\vec{Y}]$ and $T'' \equiv T[\vec{\psi}/\vec{Y}]$ where X is not among the components of $\vec{Y}$. We must show $G(\lambda X.T')$ and $(\lambda X.T') \approx (\lambda X.T'')$. Since $T'[U/X] \approx T''[U/X]$, and $\{\cdot\}^{PV}$ applies to normal forms, we have $(\lambda X.T')(U, \vec{\phi}) \approx (\lambda X.T'')(U, \vec{\phi})$, so $(\lambda X.T') \approx (\lambda X.T'')$. Now suppose $U \approx V$ and $\vec{\eta} \approx \vec{\mu}$. Since $T'[U/X] \approx T'[V/X]$ and $G(T'[V/X])$, we have $T'[U/X](\vec{\eta}) \approx T'[V/X](\vec{\eta}) \approx T'[V/X](\vec{\mu})$, so $(\lambda X.T')(U, \vec{\eta}) \approx (\lambda X.T')(V, \vec{\mu})$, so $G(\lambda X.T')$. $\Box$

7 The System IPV^ω

The system IPV^ω is a quantified version of PV^ω, employing intuitionistic predicate logic. It is a conservative extension of both PV^ω (Theorem 10.1) and IPV (Theorem 7.7). IPV^ω plays a central role in the next two sections on realizability.

The terms of IPV^ω are the same as the terms of PV^ω. The predicate symbols are $=$ and $\leq$, and the atomic formulas are all formulas of the form $t = u$ or $t \leq u$, where t, u are any type 0 terms. The formulas are formed from atomic formulas using the connectives $\wedge, \vee, \rightarrow$, and the quantifiers $\forall X, \exists X$, where X is a variable of any type. Bounded quantifiers $(\exists x \leq t)A$ and $(\forall x \leq t)A$ are defined as in §1, where x is a type 0 variable and t is a type 0 term of PV^ω not containing a free occurrence of x. The logical axioms and rules and identity axioms are the same as for IS^1_2 and IPV, understood to apply to the many-sorted predicate calculus.

The nonlogical axioms for IPV^ω consist of all theorems of PV^ω, all nonlogical axioms of IPV, and the NP induction scheme of IPV is generalized to the $PIND^\omega$ axiom scheme

$$(A[0/x] \wedge \forall x(A[\lfloor \tfrac{1}{2}x \rfloor /x] \rightarrow A)) \rightarrow \forall x A,$$

where A has the form

$$(\exists y \leq t)u = v$$

with t zero-order open (see 6.13).

Notice that in the formula A of the $PIND^\omega$ scheme the terms u and v may have free higher type variables, but free variables of t must have type 0. The reason for the latter restriction is that we will require that each bounding term t is bounded by a monotone term (Proposition 7.5). On the other hand, to prove the results on realizability the terms u and v must have free occurrences of higher type variables.

Notice that we have placed all theorems of PV^ω as axioms of IPV^ω, rather than just the axioms of PV^ω (just the axioms of PV are needed as axioms for IPV). The difficulty lies with the powerful PV^ω rule $R3^\omega$. The analogous rule $R3$ of PV is a derived rule in IPV because of the identity axioms, but $R3^\omega$ does not follow from the identity axioms because s and t may have free variables which are bound in $u\{s\}$ and $u\{t\}$. We cannot translate the rule as an axiom scheme, say $s = t \rightarrow u\{s\} = u\{t\}$, because this is not sound. We could incorporate the rule $R3^\omega$ as a rule of IPV^ω, but then the deduction theorem would not hold in IPV^ω. To preserve the deduction theorem, we could add the more general rule

$$A \rightarrow s = t \vdash A \rightarrow u\{s\} = u\{t\},$$

subject to the restriction that every free variable of s or t which becomes bound in $u\{s\}$ or $u\{t\}$ has no free occurrence in A. This rule is sound, but (because of $HTLRN$) it would considerably complicate our proof that IPV^ω is a conservative extension of IPV.

7.1 Notation *(Compare 6.3.)* $A\{S\}$ *refers to a formula with a distinguished occurrence of a term* S. *Then* $A\{T\}$ *means* $A\{S\}$ *with the indicated occurrence of* S *replaced by* T. *In general* S *and* T *may have free variables which become bound in* $A\{S\}$ *and* $A\{T\}$.

Next we prove the analog of Theorem 6.10.

7.2 Theorem a) $IPV^\omega \vdash x = 0 \rightarrow (A\{Cond_\tau(x, S, T)\} \leftrightarrow A\{S\})$
b) $IPV^\omega \vdash x \neq 0 \rightarrow (A\{Cond_\tau(x, S, T)\} \leftrightarrow A\{T\})$,
provided the indicated occurrences of x *on the right are free (not bound by quantifiers or* λ*-terms in* $A\{\ \}$*).*

Proof: Induction on the logical structure of $A\{\ \}$. The base case, in which $A\{\ \}$ is atomic, follows from 6.10 and the axiom $x \leq y \leftrightarrow Lessequ(x, y) = 0$. $\square$

7.3 Theorem *If* A *is a* Σ_0^b *formula of* IPV^ω, *then there is a term* t^A *of* PV^ω *so that*

$$IPV^\omega \vdash A \leftrightarrow t^A = 0.$$

Proof: The same as the proof of 4.4. $\square$

7.4 Theorem *If* A *is a* Σ_0^b *formula of* IPV^ω *then* $IPV^\omega \vdash A \vee \neg A$.

Proof: By 7.3, 4.3, and T43. $\square$

7.5 Proposition *For each type 0 zero-order open term* s *and all type 0 variables* x, y *there is a term* t *of* PV *whose free variables are among those in* s *such that*

(a) $IPV^\omega \vdash s \leq t$

and

(b) $IPV^\omega \vdash x \leq y \rightarrow t \leq t[y/x]$.

Proof: By Theorem 6.16, $IPV^\omega \vdash s = s^{PV}$. The result now follows from T205 (Appendix), the IPV axiom 4.1.2 and 4.3. $\square$

Recall that $(\exists \vec{y} \leq \vec{t})$ stands for $(\exists y_1 \leq t_1) \cdots (\exists y_n \leq t_n)$.

7.6 Theorem *Each instance of the* $PIND^\omega$ *axiom scheme is a theorem of* IPV^ω *when* A *has the more general form*

$$(\exists \vec{y} \leq \vec{t})u = v$$

with each t_i *zero-order open.*

Proof: Similar to the corresponding proof for IPV, which occurs in the proof of Lemma 4.11. Proposition 7.5 is used in place of T205. $\square$

7.7 Theorem *IPV^ω is a conservative extension of IPV.*

Proof: This is the one place in the paper in which a cut elimination argument is used. In outline, we argue as follows. If A is a formula in IPV and $IPV^\omega \vdash A$, then using cut elimination we can find a proof of A in IPV^ω which does not involve higher type quantifiers. Next we argue that every zero-order open substitution instance of a line in this proof is a theorem of IPV^ω, and hence each line can be transformed to a theorem of IPV using the operation $\{\ \}^{PV}$ defined in Section 6 in the proof that PV^ω is conservative over PV.

To apply cut elimination, the systems IPV and IPV^ω must be reformulated in terms of Gentzen's sequent system LJ (see [2], [4] and [36]). In LJ, each node in a proof tree is a sequent of the form $A_1, \ldots, A_n \to B$, where possibly $n = 0$ or B is missing. The logical rules are those described in Chapter 1 of [36]. (Of course it is understood that formulas meet the syntax requirements for IPV or IPV^ω.) The logical axioms are all those of the form $A \to A$, where (consistent with [4] but contrary to [36]) we require that A be atomic. The set of axioms must be closed under substitution in order for cut elimination to work, so we take as a nonlogical axiom in the new formulation any sequent of the form $\to A$, where A is any substitution instance of a nonlogical axiom in the old formulation. It is important that no axiom involve higher type quantifiers, so we take as identity axioms every instance of any sequent $x = y \to (A \leftrightarrow A[y/x])$, where A is atomic.

Since LJ is equivalent to the logical system given in Section 1 (see for example [30], Appendix A), it is not hard to see that the resulting sequent systems for IPV and IPV^ω are equivalent to the original systems, in the sense that $\to A$ is a theorem of the sequent system iff A is a theorem of the original system.

For the rest of this section we assume that IPV and IPV^ω have their sequent formulations.

A *higher type quantifier* is one of the form $\forall X$ or $\exists X$, where X is a variable not of type 0.

7.8 Definition *An instance of the cut rule*

$$\frac{\Gamma \to A \qquad A, \Delta \to B}{\Gamma \to B}$$

is called high *if the cut formula A has higher type quantifiers.*

7.9 Lemma *If $\Gamma \to A$ has a proof in IPV^ω which has no high cut, then every substitution instance of $\Gamma \to A$ has such a proof.*

Proof: Similar to the proof Lemma 1.3.5 in [36], using the fact that the axioms are closed under substitution instances. □

7.10 Lemma *If $\Gamma \to A$ is a sequent of IPV and a theorem of IPV^ω, then $\Gamma \to A$ has a proof in IPV^ω which does not involve higher type quantifiers.*

Proof: Since $\Gamma \to A$ has no higher type quantifier, it suffices to find a proof of $\Gamma \to A$ which has no high cut. Since no axiom has higher type quantifiers, using Lemma 7.9 all high cuts can be eliminated as in the proof of the cut elimination theorem in Chapter 1 of [36]. In a nut shell, the high cuts (or high "mixes") are pushed toward the leaves of the proof tree until each hits a rule which generates the cut formula either by a weakening or by introducing a higher type quantifier. In either case, the high cut can be eliminated. $\Box$

We say that a formula or sequent of IPV^ω is *zero-order open* if all free variables have type 0 (see Definition 6.13). A *zero-order open instance* of a formula or sequent is a substitution instance that is zero-order open.

We now extend the transformation $\{\ \}^{PV}$ (see 6.14) to $\{\ \}^{IPV}$.

7.11 Definition *The transformation $A \rightsquigarrow \{A\}^{IPV}$ takes a zero-order open formula A of IPV^ω with no higher type quantifiers to an equivalent formula $\{A\}^{IPV}$ of IPV, and is defined inductively as follows:*

$$
\begin{aligned}
&\{t = u\}^{IPV} \stackrel{d}{\equiv} t^{PV} = u^{PV} \\
&\{t \le u\}^{IPV} \stackrel{d}{\equiv} t^{PV} \le u^{PV} \\
&\{AcB\}^{IPV} \stackrel{d}{\equiv} A^{IPV} c B^{IPV}, \textit{ where } c \textit{ is } \wedge,\ \vee,\ \textit{or} \to. \\
&\{\forall x A\}^{IPV} \stackrel{d}{\equiv} \forall x A^{IPV} \\
&\{\exists x A\}^{IPV} \stackrel{d}{\equiv} \exists x A^{IPV}
\end{aligned}
$$

We extend $\{\ \}^{IPV}$ to sequents by the obvious conventions $\{A_1, \ldots, A_n\}^{IPV} \stackrel{d}{\equiv} A_1^{IPV}, \ldots, A_n^{IPV}$, and $\{\Gamma \to A\}^{IPV} \stackrel{d}{\equiv} \Gamma^{IPV} \to A^{IPV}$. Note that the transformation $\{\ \}^{IPV}$ fixes objects in IPV. Note also that by induction on the above definition, using Lemma 6.15, $\{\ \}^{IPV}$ is transparent to substitution. That is, for every zero-order open type zero term t of PV^ω

$$\{S[t/x]\}^{IPV} \equiv S^{IPV}[t^{PV}/x],$$

where S is a sequent or formula of IPV^ω.

Theorem 7.7 now follows immediately from Lemma 7.10 and the following:

7.12 Lemma *If a proof P of IPV^ω does not involve higher type quantifiers, then $IPV \vdash \{S'\}^{IPV}$ for every zero-order open instance S' of any sequent S in P.*

Proof: Induction on the length of the longest path from S to a leaf (axiom) in P. For the base case S is an axiom. Then either S' is an axiom of IPV, so the result is obvious, or

S' is $\to A$ where A is a theorem of PV^{ω}, so the result follows from Theorem 6.17, or S is a $PIND^{\omega}$ axiom, so by transparency of substitution $\{S'\}^{IPV}$ is an IPV induction axiom.

For the induction step, if S is the consequence in P of applying a rule of LJ, then the induction hypothesis applies to the upper sequent(s) in the rule, and the result in every case follows from a simple argument. (In case the LJ rule is a quantifier inference, we use the assumption that the quantifiers are type 0.) $\square$

8 Realizability

In this section we present a form of Kreisel's modified realizability for the system IPV^ω. As applications, we show that decidable formulas represent polynomial time predicates, prove a version of Buss's main theorem for IS_2^1 relating existence proofs and polynomial time functions, and prove two of Buss's conjectures concerning IS_2^1.

Our version of realizability is a translation of IPV^ω into itself, associating with each formula A of IPV^ω a formula $\vec{X}ⓇA$ (read $\vec{X}$ realizes A) of IPV^ω, where $\vec{X}$ is a sequence of zero or more variables of types determined by the logical structure of A. Intuitively, $\vec{X}ⓇA$ says that the functions represented by the variables $\vec{X}$ help to explain why A is true. Our definition follows that presented in [37]. The main difference is the conjunct $A \to B$ in clause (iv) below, which assures that provably realizable formulas are provable, and allows us to prove Theorem 8.4 and Buss's conjectures. In standard realizability, a suitable version of the axiom of choice is provably realizable but not provable [37].

We use the notation $\vec{X}$ for $X_1, \ldots, X_n$ and $\vec{Y}$ for $Y_1, \ldots, Y_k$, $n, k \geq 0$, and $\vec{Y}(\vec{X})$ for $Y_1(\vec{X}), \ldots, Y_k(\vec{X})$, etc. Also Λ stands for the empty sequence of variables, and $Y(\Lambda) \stackrel{d}{\equiv} Y$.

8.1 Definition *$\vec{X}ⓇA$ is defined by induction on the logical structure of A. We assume that no variable in the list $\vec{X}$ occurs free in A. $\vec{T}ⓇA \stackrel{d}{\equiv} (\vec{X}ⓇA)[\vec{T}/\vec{X}]$; that is $\vec{X}ⓇA$ with the simultaneous substitution of T_i for X_i $(1 \leq i \leq n)$.*

(i) $\Lambda ⓇA \stackrel{d}{\equiv} A$, if A is atomic.

(ii) $\vec{X}, \vec{Y}Ⓡ(A \wedge B) \stackrel{d}{\equiv} \vec{X}ⓇA \wedge \vec{Y}ⓇB$.

(iii) $z, \vec{X}, \vec{Y}Ⓡ(A \vee B) \stackrel{d}{\equiv} (z = 0 \wedge \vec{X}ⓇA) \vee (z \neq 0 \wedge \vec{Y}ⓇB)$.

(iv) $\vec{Y}Ⓡ(A \to B) \stackrel{d}{\equiv} \forall \vec{X}(\vec{X}ⓇA \to \vec{Y}(\vec{X})ⓇB) \wedge (A \to B)$.

(v) $\vec{X}Ⓡ\forall Y A \stackrel{d}{\equiv} \forall Y(\vec{X}(Y)ⓇA)$.

(vi) $Z, \vec{X}Ⓡ\exists Y A \stackrel{d}{\equiv} \vec{X}ⓇA[Z/Y]$.

8.2 Proposition $\qquad \Lambda Ⓡ\neg A \equiv \forall \vec{Y}(\neg(\vec{Y}ⓇA)) \wedge \neg A.$

Proof: This is immediate from (i), (iv), and the definition $\neg A \equiv A \to 0 = 1$. □

8.3 Proposition a) *Each free variable of $\vec{X}ⓇA$ is either free in A or in the list $\vec{X}$.*

b) *If Y is not in the list $\vec{X}$, then $\vec{X}ⓇA[T/Y] \equiv (\vec{X}ⓇA)[T/Y]$.*

Proof: Induction on the logical structure of A. $\square$

8.4 Theorem *For every formula A of IPV^ω*

$$IPV^\omega \vdash \exists \vec{X}(\vec{X} \circledR A) \rightarrow A.$$

Proof: Induction on the logical structure of A. Each case follows easily from the corresponding clause of Definition 8.1. For clause (iv) we do not need the induction hypothesis. $\square$

8.5 Corollary $$IPV^\omega \vdash (\Lambda \circledR \neg A) \leftrightarrow \neg A.$$

Proof: This is an easy consequence of 8.2 and 8.4. $\square$

The above corollary shows that realizability sheds no light on negated formulas. (This is in contrast to the *Dialectica* interpretation in Section 9.)

8.6 Soundness Theorem *If $IPV^\omega \vdash A$ then $IPV^\omega \vdash \vec{T} \circledR A$ for some sequence $\vec{T}$ of terms whose free variables are among the free variables of A.*

Before giving the proof, consider as an illustration Fermat's "Little Theorem", which states that if $0 < a < n$ and n is prime, then $a^{n-1} \bmod n = 1$. Since $a^{n-1} \bmod n$ is a polynomial time computable function of a and n, the conclusion is an atomic formula of IPV, so our form of realizability gives no information. (However the *Dialectica* interpretation does give interesting information; see the end of §9.) On the other hand, suppose we let the formula B be the following form of the contrapositive:

8.7 $$\forall a \forall n[0 < a \wedge a < n \wedge a^{n-1} \bmod n \neq 1) \rightarrow \exists d(d|n \wedge d \neq 1 \wedge d \neq n)]$$

where $d|n$ is the polynomial time predicate "d divides n". Then if B is provable in IPV^ω, by the Soundness Theorem there is a term D such that $\vdash D \circledR B$. That is, the formula

8.8 $$\forall a \forall n[0 < a \wedge a < n \wedge a^{n-1} \bmod n \neq 1) \rightarrow (D(a,n)|n \wedge D(a,n) \neq 1 \wedge D(a,n) \neq n)]$$

would be a theorem of IPV^ω. For "most" composite numbers n a random number a such that $0 < a < n$ will satisfy the antecedent. Hence such a realizing function D might well provide a practical method for factoring large numbers. In any case the existence of such a polynomial time function D would represent a surprising and major result in complexity theory. We conjecture that B is not a theorem of IPV^ω.

Proof of 8.6: Induction on the IPV^ω proof of A. We first consider the logical axioms and rules, as presented in Section 1. The argument is straightforward in all cases (for example, $PIND^\omega$ axioms and $HTLRN$ are not needed), but in just one case, Axiom Scheme 2, the higher type conditional (6.8) is needed. We give three example axioms and two example rules, starting with the most interesting case.

Axiom Scheme 2. $\vec{W}®(A \vee A) \to A \equiv$

$$\forall z\vec{X}\vec{Y}[((z = 0 \wedge \vec{X}®A) \vee (z \neq 0 \wedge \vec{Y}®A)) \to \vec{W}(z, \vec{X}, \vec{Y})®A] \wedge ((A \vee A) \to A)$$

where $\vec{X} \equiv (X_i, \ldots, X_n), \vec{Y} \equiv (Y_1, \ldots, Y_n)$ and $\vec{W} \equiv (W_1, \ldots, W_n)$. Let

$$T_i \equiv \lambda z\vec{X}\vec{Y}Cond_{\tau_i}(z, X_i, Y_i) \quad (1 \leq i \leq n),$$

where τ_i is the type of X_i (and hence of Y_i). Then $IPV^\omega \vdash \vec{T}®((A \vee A) \to A)$ by repeated application of axiom scheme (β) and Theorem 7.2. (Recall that equations such as $Cond_\tau(0, X, Y) = X$ cannot be expressed in IPV^ω when $\tau \neq 0$, so Theorem 7.2 is needed.)

Axiom Scheme 4. $Z, \vec{V}, \vec{W}®B \to (A \vee B) \equiv$

$$\forall\vec{X}[\vec{X}®B \to (Z(\vec{X}) = 0 \wedge \vec{V}(\vec{X})®A) \vee (\vec{Z}(\vec{X}) \neq 0 \wedge \vec{W}(\vec{X})®B)] \wedge (B \to (A \vee B)).$$

Thus $IPV^\omega \vdash R, \vec{S}, \vec{T}®B \to (A \vee B)$, where $R \equiv \lambda\vec{X}.1$, $\vec{S} \equiv \lambda\vec{X}.\vec{U}$, and $\vec{T} \equiv \lambda\vec{X}.\vec{X}$, where $\vec{U}$ is any sequence of closed terms such that $\vec{U}®A$ is syntactically correct.

Axiom Scheme 7. $\vec{Z}®\forall YA \to A[T/Y] \equiv$

$$\forall\vec{X}(\forall Y(\vec{X}(Y)®A) \to \vec{Z}(\vec{X})®A[T/Y]) \wedge \forall YA \to A[T/Y].$$

Thus $IPV^\omega \vdash \vec{S}®\forall YA \to A[T/Y]$, where $S_i \equiv \lambda\vec{X}.X_i(T)$, $1 \leq i \leq n$. To carry out this IPV^ω proof, we use the fact (by 8.3b)) that $(\vec{X}(Y)®A)[T/Y] \equiv \vec{X}(T)®A[T/Y]$.

Rule 11. If $IPV^\omega \vdash \vec{S}®(A \to B)$ and $IPV^\omega \vdash \vec{T}®(B \to C)$ then $IPV^\omega \vdash \lambda\vec{X}.\vec{T}(\vec{S}(\vec{X}))$ $®(A \to C)$.

Rule 16. If $IPV^\omega \vdash \vec{S}®(B \to A)$ and Y does not occur free in A, then $IPV^\omega \vdash$ $\lambda Y\vec{X}.\vec{S}(\vec{X})®(\exists YB \to A)$.

For the equality axioms, we can apply 8.3 b) to show

Axiom Scheme 16. $IPV^\omega \vdash \lambda\vec{W}.\vec{W}, \lambda\vec{W}.\vec{W}®x = y \to (A \leftrightarrow A[y/x])$.

We next consider the nonlogical axioms of IPV^ω. The theorems of PV^ω are all atomic formulas, and hence 8.6 is trivial for them. Considering nonlogical axioms of IPV, the empty sequence realizes 4.1.2, $Parity(x)$ realizes 4.1.3, and x realizes 4.1.4.

It remains to prove 8.6 for the case of $PIND^\omega$ axioms. We use the notation $B(x,y)$ for the equation $u = v$ of PV^ω, where x, y are type 0 variables, and $B(s,t)$ denotes $u = v$ with the simultaneous substitution of s, t for x, y. Then a $PIND^\omega$ axiom has the form

$$I \overset{d}{\equiv} ((\exists y \le t'')B(0,y) \wedge \forall x J) \rightarrow \forall x(\exists y \le t)B(x,y)$$

where

$$\begin{aligned} &J \overset{d}{\equiv} ((\exists y \le t')B(\lfloor \tfrac{1}{2}x \rfloor, y) \rightarrow (\exists y \le t)B(x,y)), \\ &t'' \equiv t[0/x] \text{ and } t' \equiv t[\lfloor \tfrac{1}{2}x \rfloor / x], \end{aligned}$$

and t is zero-order open with no free occurrence of y. Our task is to find a term $\mathcal{S}$ of PV^ω such that $IPV^\omega \vdash \mathcal{S}Ⓡ I$. We have

$$\begin{aligned} \mathcal{S}Ⓡ I \quad \equiv \quad & \forall yY[(y \le t'' \wedge B(0,y) \wedge \forall x(Y(x)Ⓡ J)) \rightarrow \\ & \forall x(\mathcal{S}(y,Y,x) \le t \wedge B(x, \mathcal{S}(y,Y,x)))] \wedge I \end{aligned}$$

and

$$Y(x)Ⓡ J \equiv \forall y[y \le t' \wedge B(\lfloor \tfrac{1}{2}x \rfloor, y) \rightarrow Y(x,y) \le t \wedge B(x, Y(x,y))] \wedge J.$$

Intuitively, the function $\mathcal{S}(y,Y,x)$ supplies values for the existential quantifier in $\exists z B(x, z)$, given a value y satisfying $B(0,y)$ and given the type 1 function Y which takes values (w, y') satisfying $B(\lfloor \frac{1}{2}w \rfloor, y')$ to a value y'' satisfying $B(w, y'')$. When the hypotheses of $\mathcal{S}Ⓡ I$ are satisfied we would like

8.9 $$\mathcal{S}(y,Y,0) = y$$

8.10 $$\mathcal{S}(y,Y,x) = Y(x, \mathcal{S}(y,Y,\lfloor \tfrac{1}{2}x \rfloor)),\ x \ne 0,$$

and

8.11 $$\mathcal{S}(y,Y,x) \le t.$$

Proceeding formally, we let

$$\mathcal{S} \overset{d}{\equiv} \lambda yYx.\mathcal{R}(y, \lambda x'z.Y(x',z), \lambda x'.2 \cdot t[x'/x], x).$$

With this definition of $\mathcal{S}$ we have by $HTLRN$

8.12 $$IPV^\omega \vdash \mathcal{S}(y,Y,x) = \begin{cases} \textit{if } x = 0 \textit{ then } y \\ \quad \textit{else} \\ Cond(u \dot{-} 2 \cdot t, u, 2 \cdot t) \end{cases}$$

where $u \overset{d}{\equiv} Y(x, \mathcal{S}(y,Y,\lfloor \frac{1}{2}x \rfloor))$.

We would like to prove $\mathcal{S}\circledR I$ in IPV^ω by induction on x. In order to apply a suitable $PIND^\omega$ axiom, we must transform $\mathcal{S}\circledR I$ to be of the form $\exists y \leq t(u = v)$. First we drop the conjunct I from $\mathcal{S}\circledR I$, since I is already an axiom of IPV^ω, and we strengthen the assertion $\mathcal{S}\circledR I$ by dropping the conjunct J from $Y(x)\circledR J$. The result has the form

8.13
$$\forall y Y[(C \wedge \forall xyD) \rightarrow \forall xE],$$

where

$$C \stackrel{d}{\equiv} y \leq t'' \wedge B(0, y)$$
$$D \stackrel{d}{\equiv} y \leq t' \wedge B(\lfloor \tfrac{1}{2}x \rfloor, y) \rightarrow Y(x, y) \leq t \wedge B(x, Y(x, y))$$
$$E \stackrel{d}{\equiv} \mathcal{S}(y, Y, x) \leq t \wedge B(x, \mathcal{S}(y, Y, x)).$$

Note that C, D, and E are quantifier-free.

Let

$$A \stackrel{d}{\equiv} \exists x' \leq x \exists y' \leq u(C \wedge D'. \rightarrow E)$$

where $u \stackrel{d}{\equiv} t[\lfloor \frac{1}{2}x' \rfloor / x]$ and $D' \stackrel{d}{\equiv} D[x'y'/xy]$. Then A is a suitable formula for a $PIND^\omega$ axiom by 7.3 and 7.6. We use this fact to argue below that $IPV^\omega \vdash A$. From this, it is easy to see that $IPV^\omega \vdash \mathcal{S}\circledR I$. First note that using Theorem 4.3, the truth-functional connectives in A can be re-interpreted as the usual connectives $\wedge, \vee, \neg, \rightarrow$ in IPV^ω. Next, the bounds on the existential quantifiers in A can be dropped (they only strengthen the assertion) and standard intuitionistic reasoning then yields 8.13, and hence $IPV^\omega \vdash \mathcal{S}\circledR I$.

It remains to prove A by $PIND^\omega$ induction. For the base case, note that 8.9 is a theorem of IPV^ω, so that $\vdash A[0/x]$ with $x' = 0$ and $y' = 0$. For the induction step, assume $A[\lfloor \frac{1}{2}x \rfloor / x]$ and let x', y' satisfy the existential quantifiers. Since classical truth-functional reasoning can be carried out in PV^ω, we argue that either (x', y') satisfy $C \wedge D'$ or not. If not, then the same values x', y' satisfy the existential quantifiers in A (note that x does not occur free in u, C, or D'). Hence $A[\lfloor \frac{1}{2}x \rfloor / x] \rightarrow A$ in this case.

Now suppose x', y' satisfy $C \wedge D'$, so $E[\lfloor \frac{1}{2}x \rfloor / x]$ holds. Reassign $x' \leftarrow x$ and $y' \leftarrow \mathcal{S}(y, Y, \lfloor \frac{1}{2}x \rfloor)$. Then by $E[\lfloor \frac{1}{2}x \rfloor / x]$, $y' \leq t'$, so $y' \leq u$, so the existential bounds in A are satisfied. If $C \wedge D'$ is now false, then A follows and we are done. Now assume $C \wedge D'$ holds for these new values of x', y'. Then $E[\lfloor \frac{1}{2}x \rfloor / x]$ is equivalent to the antecedent of D', so the latter holds, so the consequent holds; that is

8.14
$$Y(x, \mathcal{S}(y, Y, \lfloor \tfrac{1}{2}x \rfloor)) \leq t \wedge B(x, Y(x, \mathcal{S}(y, Y, \lfloor \tfrac{1}{2}x \rfloor))).$$

Thus it suffices to show that 8.10 holds under the assumption 8.14. This follows from 8.12 and the simple fact

$$IPV \vdash x \leq y \rightarrow x \mathbin{\dot{-}} (2 \cdot y) = 0. \quad \square$$

One application of the Soundness Theorem 8.6 is to show that every formula which is decidable in IPV^ω provably represents a polynomial time predicate.

8.15 Proposition *If $IPV^\omega \vdash A(x) \vee \neg A(x)$, where $A(x)$ has only the type 0 variable x occurring free, then there is a PV function symbol f such that*

$$IPV^\omega \vdash f(x) = 0 \leftrightarrow A(x).$$

The same holds with IPV^ω replaced (twice) by IPV.

Proof: $T, \vec{U} \circledR \forall x (A(x) \vee \neg A(x)) \equiv$

$$\forall x [(T(x) = 0 \wedge \vec{U} \circledR A(x)) \vee (T(x) \neq 0 \wedge \Lambda \circledR \neg A(x))]$$

where T is closed and has type $0 \to 0$. Hence $\{T(x)\}^{PV}$ is a term of PV, and by 6.16 and 3.2.7 $\vdash f(x) = T(x)$, where $f \equiv [\lambda x. \{T(x)\}^{PV}]$. The result for IPV^ω now follows by 8.6 and 8.4. The result for IPV follows because IPV^ω is conservative over IPV (7.7). $\square$

Since every NP predicate can be represented by a Σ_1^b formula, it follows from 8.15 that not all such formulas are decidable in IPV unless $P = NP$. For example, suppose $satisfy(x, y)$ is a PV term which is 0 iff y codes a truth assignment which satisfies the propositional formula coded by x (under suitable encodings). If $(\exists y \leq x) satisfy(x, y) = 0$ is decidable in IPV, then $P = NP$.

As a second application of the Soundness Theorem we show that some existentially quantified variables can be replaced by PV^ω terms.

8.16 Theorem *Let $\forall \vec{X} \exists Y A(\vec{X}, Y)$ be a closed Theorem of IPV^ω. Then there is a closed term S of IPV^ω such that*

$$IPV^\omega \vdash \forall \vec{X} A(\vec{X}, S(\vec{X})).$$

Proof: Note that

$$Z, \vec{W} \circledR \forall \vec{X} \exists Y A(\vec{X}, Y) \equiv \forall \vec{X} (\vec{W}(\vec{X}) \circledR A(\vec{X}, Z(\vec{X}))).$$

Thus by Theorem 8.6 there are closed terms $S, \vec{T}$ such that

$$IPV^\omega \vdash \forall \vec{X} (\vec{T}(\vec{X}) \circledR A(\vec{X}, S(\vec{X})).$$

Hence by Theorem 8.4

$$IPV^\omega \vdash \forall \vec{X} A(\vec{X}, S(\vec{X})). \quad \square$$

8.17 Corollary *Let $\forall \vec{x} \exists y A(\vec{x}, y)$ be a closed theorem IPV^ω such that the variables $\vec{x}, y$ have type 0. Then there is a PV function symbol f such that*

$$IPV^\omega \vdash \forall \vec{x} A(\vec{x}, f(\vec{x})).$$

The same is true with IPV^ω replaced (twice) by IPV.

Proof: By Theorem 8.16 and 6.20

$$IPV^{\omega} \vdash \forall \vec{x} A(\vec{x}, \{S(\vec{x})\}^{PV})$$

for some type 1 term S. We then argue as in the proof of 8.15. $\square$

Buss's main result (Theorem 2) in [4] is closely related to the above Corollary. In fact, 8.17 together with 10.6 (stating that our formulation of IS_2^1 is equivalent to Buss's) and 4.12 imply Buss's Theorem 2 for the case of IS_2^1. Buss's Theorem 2 applies more generally to any system in his hierarchy IS_2^i, $i \geq 1$, but his conclusion is weaker since it asserts truth instead of provability.

The next result proves Buss's Conjecture 1 ([4], §7) for the case of the system IS_2^1. (For the rest of this section we use the notation $A(\vec{x}, y)$ to display *all* free variables in A.)

8.18 Corollary *If $IS_2^1 \vdash \exists y A(\vec{x}, y)$, then there is a Σ_1^{b+} formula $B(\vec{x}, y)$ such that IS_2^1 proves the following formulas:*

(1) $\forall \vec{x}\, \forall y (B(\vec{x}, y) \to A(\vec{x}, y))$

(2) $\forall \vec{x}\, \forall y \forall z (B(\vec{x}, y) \wedge B(\vec{x}, z) \to y = z)$

(3) $\forall \vec{x}\, \exists y B(\vec{x}, y)$

Proof: By 4.12 and 8.17, $IPV \vdash \forall \vec{x}\, \exists y A(\vec{x}, f(\vec{x}))$ for some PV function symbol f. By 2.7 there is a Σ_1^{b+} formula $B(\vec{x}, y)$ of IS_2^1 which defines f. Thus $IPV \vdash y = f(\vec{x}) \leftrightarrow B(\vec{x}, y)$, and it follows that (1), (2), (3) are theorems of IPV, and hence of IS_2^1. $\square$

The next result proves Conjecture 2 of Buss [4]. Here we let I_n be the binary numeral for $n \in \mathbb{N}$. Thus $I_0 \stackrel{d}{\equiv} 0$, $I_{2n+1} \stackrel{d}{\equiv} s_1(I_n)$, and for $n \neq 0$, $I_{2n} \stackrel{d}{\equiv} s_0(I_n)$. If $\vec{n} = n_1, \ldots, n_k$, then $I_{\vec{n}} \stackrel{d}{\equiv} I_{n_1}, \ldots, I_{n_k}$. Here also Gödel numbers must be efficient, in the sense that the length of the Gödel number of a proof must be bounded by a polynomial in the length (number of symbols) in the proof.

8.19 Corollary *If $IS_2^1 \vdash \exists y A(\vec{x}, y)$, then there are polynomial time functions f and g such that for all $\vec{n} \in \mathbb{N}^k$, $g(\vec{n})$ is the Gödel number of an IS_2^1 proof of $A(I_{\vec{n}}, I_{f(\vec{n})})$.*

Proof: The IS_2^1 proof of $A(I_{\vec{n}}, I_{f(\vec{n})})$ consists of (a) a proof of 8.18.1, followed by (b) a proof of $B(I_{\vec{n}}, I_{f(\vec{n})})$, followed by (c) an instantiation of 8.18.1 and modus ponens. Since (a) is a constant and (c) has a constant number of lines which are easily computed in polynomial time, it suffices to show that the proof of $B(I_{\vec{n}}, I_{f(\vec{n})})$ can be computed in time bounded by a polynomial in the length of $\vec{n}$. Referring to the proof of 2.7, this is shown by induction on the representation of f guaranteed by Cobham's theorem. Intuitively, the

IS_2^1 proof of $B(I_{\vec{n}}, I_{f(\vec{n})})$ mimics the evaluation of $f(\vec{n})$. Since the latter is polynomial time, the former can be generated in polynomial time. $\Box$

Conjecture 3 in Buss [4] asserts that the above argument can be formalized in IS_2^1. Since no infeasible concepts are involved in the argument, this amounts to an exercise in formalizing a proof, similar to the proof of Gödel's second incompleteness theorem (see Buss [2] for S_2^1 and Cook [11] for PV).

9 The *Dialectica* Interpretation

We define a second translation from IPV^ω into itself which follows the functional interpretation of Heyting arithmetic in Gödel [20]. The translation differs from that of the previous section in the treatment of implication. The more radical translation of implications in Gödel's interpretation allows elimination of logical operators.

The translation associates with each formula A of IPV^ω a formula $\exists\vec{x}\forall\vec{y}\, t_A = 0$, where $\vec{x}$ and $\vec{y}$ are finite sequences of variables of finite type and t_A is a term of PV^ω. The types of $\vec{x}$ and $\vec{y}$ depend only on the logical structure of A; the free variables of A are included in the free variables of t_A. In the definition below, the following notational conventions are used:

1. If $s = 0$ and $t = 0$ are equations of PV^ω, we write:

$$\begin{array}{ccc} (s = 0\ \&\ t = 0) & \text{for} & (s \& t = 0); \\ (s = 0 \vee t = 0) & \text{for} & (s \vee t = 0); \\ (s = 0 \supset t = 0) & \text{for} & (s \supset t = 0); \\ s \neq 0 & \text{for} & \sim s = 0. \end{array}$$

2. $\vec{x}, \vec{y}, \vec{u}, \vec{v}$ are finite sequences of distinct variables of finite type, while z is a numerical variable.

3. $\vec{U}$ is a sequence of variables whose number and types are determined by the fact that each of them can be applied to $\vec{x}$ as an argument sequence and that the sequence $\vec{U}(\vec{x})$ so obtained agrees with the sequence $\vec{u}$ with respect to the number and type of its members. If $\vec{u}$ is the empty sequence then $\vec{U}$ is empty; if $\vec{x}$ is empty then $\vec{U} = \vec{u}$.

4. One-element sequences are identified with their only elements.

9.1 Definition *If A is a formula of IPV^ω, the* Dialectica *translation of A, A^D, is defined by induction on the logical complexity of A:*

(i) *If A is an equation $u = v$ then A^D is $Equ(u, v) = 0$;*

(ii) *If A is an inequality $u \leq v$ then A^D is $Lessequ(u, v) = 0$;*

For the remaining cases, assume that

$A^D \stackrel{d}{\equiv} \exists\vec{x}\forall\vec{y}\, t_A = 0,$

$B^D \stackrel{d}{\equiv} \exists\vec{u}\forall\vec{v}\, t_B = 0.$

(iii) $(A \wedge B)^D \stackrel{d}{\equiv} \exists\vec{x}\vec{u}\forall\vec{y}\vec{v}[t_A = 0\ \&\ t_B = 0]$;

(iv) $(A \vee B)^D \stackrel{d}{\equiv} \exists z\vec{x}\vec{u}\forall\vec{y}\vec{v}[(z = 0\ \&\ t_A = 0) \vee (z \neq 0\ \&\ t_B = 0)]$;

(v) $(\exists w A)^D \stackrel{d}{\equiv} \exists w\vec{x}\forall\vec{y}\, t_A(\vec{x}, \vec{y}) = 0$;

(vi) $(\forall w A)^D \stackrel{d}{\equiv} \exists \vec{X} \forall w \vec{y}\, t_A(\vec{X}(w), \vec{y}) = 0$;

(vii) $(A \to B)^D \stackrel{d}{\equiv} \exists \vec{U}\vec{Y} \forall \vec{x}\vec{v}[t_A(\vec{x}, \vec{Y}(\vec{x}\,\vec{v})) = 0 \supset t_B(\vec{U}(\vec{x}), \vec{v}) = 0]$.

The definition of $\neg A$ *as* $(A \to (0 = 1))$ *leads to the equivalence:*

(viii) $(\neg A)^D \stackrel{d}{\equiv} \exists \vec{Y} \forall \vec{x}\, t_A(\vec{x}, \vec{Y}(\vec{x})) \neq 0$.

In the rules above, $t_A(\vec{s}, \vec{t})$ stands for $t_A[\vec{s}, \vec{t}/\vec{x}, \vec{y}]$. In applying rules (ii) to (vi), we assume that the bound variables of A^D and B^D are disjoint, and that the variables introduced in (iv), (vi) and (vii) are chosen to be mutually distinct and different from the variables in the formulas concerned.

9.2 Definition *We denote by* MP (Markov's Principle) *the scheme* $\neg\neg\exists\vec{x}A \to \exists\vec{x}A$, *where* A *is an atomic formula.*

We will use MP both in the contexts of IPV and IPV^ω. For IPV^ω, the variables in the sequence $\vec{x}$ are of arbitrary finite type.

We note that in general, instances of MP are not theorems of IPV^ω, nor of any of the usual formal theories of arithmetic employing intuitionistic logic. For example, let M be a machine which computes a function whose domain is not recursive, and let $A_M(x, y)$ assert that y codes a halting computation of M with input x. The $\forall x(\neg\neg\exists y A_M(x, y) \to \exists y A_M(x, y))$ cannot be realized (in the sense of §8) by any total recursive function.

However, it follows from Theorem 10.4 in the next section that if A is an atomic formula of IPV^ω and $IPV \vdash \neg\neg\exists\vec{x}A$, then $IPV \vdash \exists\vec{x}A$. Similarly for IPV^ω.

9.3 Theorem *(Soundness of* Dialectica *interpretation). If* $A(\vec{z})$ *is a formula of* IPV^ω*, whose free variables are contained in* $\vec{z}$*, such that* $IPV^\omega + MP \vdash A(\vec{z})$ *then there is a sequence* $\vec{S}$ *of closed terms of* PV^ω *so that*

$$PV^\omega \vdash t_A(\vec{S}(\vec{z}), \vec{y}, \vec{z}) = 0,$$

where $A^D \stackrel{d}{\equiv} \exists\vec{x}\forall\vec{y}\, t_A(\vec{x}, \vec{y}, \vec{z}) = 0$.

Proof: Induction on the proof of A. If A is a logical axiom, the proof is usually straightforward. The most difficult logical axiom to verify is the scheme: $A \to (A \wedge A)$ (see §1). The translation of this scheme takes the form:

$$\exists \vec{X}_1\vec{X}_2\vec{Y}\forall\vec{x}\vec{y}_1\vec{y}_2[t_A(\vec{x}, \vec{Y}(\vec{x}\vec{y}_1\vec{y}_2)) = 0 \,.\!\supset\!.\, t_A(\vec{X}_1(\vec{x}), \vec{y}_1) = 0 \;\&\; t_A(\vec{X}_2(\vec{x}), \vec{y}_2) = 0].$$

To find appropriate sequences $\vec{S}_1, \vec{S}_2, \vec{T}$ of PV^ω terms, set $\vec{S}_1(\vec{x}) = \vec{S}_2(\vec{x}) \stackrel{d}{\equiv} \vec{x}$, and for $1 \le i \le k$ where $\vec{y} = y_1, \ldots, y_k$,

$$T_i(\vec{x}\vec{y}_1\vec{y}_2) \stackrel{d}{\equiv} Cond(t_A(\vec{x}, \vec{y}_2), (\vec{y}_1)_i, (\vec{y}_2)_i).$$

By 6.10,

$$PV^\omega \vdash t_A(\vec{x},\vec{y}_2)=0 \supset t_A(\vec{x},\vec{T}(\vec{x}\vec{y}_1\vec{y}_2))=t_A(\vec{x},\vec{y}_1),$$

$$PV^\omega \vdash t_A(\vec{x},\vec{y}_2)\neq 0 \supset t_A(\vec{x},\vec{T}(\vec{x}\vec{y}_1\vec{y}_2))=t_A(\vec{x},\vec{y}_2).$$

By T43,

$$PV^\omega \vdash t_A(\vec{x},\vec{y}_2)=0 \vee t_A(\vec{x},\vec{y}_2)\neq 0,$$

so by classical truth-functional reasoning,

$$PV^\omega \vdash t_A(\vec{x},\vec{T}(\vec{x}\vec{y}_1\vec{y}_2))=0 \,.\supset.\, t_A(\vec{x},\vec{y}_1)=0 \;\&\; t_A(\vec{x},\vec{y}_2)=0.$$

The decidability of atomic formulas is essential to the soundness proof for this axiom scheme. For this reason, higher type equality statements are not included in the basis of PV^ω or IPV^ω. Although higher type equality can be treated as a decidable notion by giving an intensional interpretation to higher type identities (Tait [35]), the decision procedure is not elementary recursive by Statman [34]. The soundness proofs for the remaining logical axiom schemes are omitted.

The logical rules are also not difficult to verify. As an illustration, we verify the interpretation of Rule 11. Assume that there are sequences of terms $\vec{Q},\vec{R},\vec{S},\vec{T}$, so that:

$$PV^\omega \vdash t_A(\vec{x},\vec{R}(\vec{x}\vec{v}))=0 \supset t_B(\vec{Q}(\vec{x}),\vec{v})=0,$$

$$PV^\omega \vdash t_B(\vec{u},\vec{T}(\vec{u}\vec{z}))=0 \supset t_C(\vec{S}(\vec{u}),\vec{z})=0.$$

We wish to find sequences $\vec{N},\vec{P}$ of terms so that:

$$PV^\omega \vdash t_A(\vec{x},\vec{P}(\vec{x}\vec{z}))=0 \supset t_C(\vec{N}(\vec{x}),\vec{z})=0.$$

Substituting $\vec{Q}(\vec{x})$ for $\vec{u}$ and $\vec{T}(\vec{Q}(\vec{x})\vec{z})$ for $\vec{v}$, we obtain:

$$PV^\omega \vdash t_A(\vec{x},\vec{R}(\vec{x}\vec{T}(\vec{Q}(\vec{x})\vec{z})))=0 \supset t_B(\vec{Q}(\vec{x}),\vec{T}(\vec{Q}(\vec{x})\vec{z}))=0,$$

$$PV^\omega \vdash t_B(\vec{Q}(\vec{x}),\vec{T}(\vec{Q}(\vec{x})\vec{z}))=0 \supset t_C(\vec{S}(\vec{Q}(\vec{x})),\vec{z})=0.$$

Thus a solution is found by setting $\vec{N} \stackrel{d}{\equiv} \lambda\vec{x}.\vec{S}(\vec{Q}(\vec{x}))$, $\vec{P} \stackrel{d}{\equiv} \lambda\vec{x}\vec{z}.\vec{R}(\vec{x}\vec{T}(\vec{Q}(\vec{x})\vec{z}))$.

In the case of the non-logical axioms of IPV^ω, the equations $s=t$ and $Equ(s,t)=0$ are inter-deducible in PV^ω by the theorems $Equ(x,x)=0$ and $Cond(Equ(x,y),x,y)=y$. Thus 9.3 holds whenever A is a theorem of PV^ω. The translation of Axiom 4.1.2 of IPV takes the form $(B \leftrightarrow B)^D$; it is straightforward to show that 9.3 holds for Axioms 4.1.3 and 4.1.4 of IPV. It remains to prove 9.3 for the case of $PIND^\omega$ axioms.

An instance of the NP induction scheme of IPV^ω has the form:

$$(A[0/x] \wedge \forall x(A[\lfloor \tfrac{1}{2}x \rfloor/x] \to A)) \to \forall x A$$

where A has the form $(\exists y \leq t)u = v$, with t zero-order open. We shall write s as an abbreviation of the term $(Lessequ(y,t) \& Equ(u,v))$; we write $s(p,q)$ for $s[p,q/x,y]$. With this convention, the translation of the induction axiom is:

$$\begin{aligned}&\exists \mathcal{Y}_3 \mathcal{X}_1 \mathcal{Y}_1 \forall \vec{Z}[\{s(0,y_0) = 0 \,\& \\ &s(\lfloor \tfrac{1}{2}\mathcal{X}_1(\vec{Z})\rfloor, \mathcal{Y}_1(\vec{Z})) = 0 \supset s(\mathcal{X}_1(\vec{Z}), Y_2(\mathcal{X}_1(\vec{Z}), \mathcal{Y}_1(\vec{Z}))) = 0)\} \\ &\qquad\qquad\qquad \supset s(x, \mathcal{Y}_3(\vec{Z})) = 0],\end{aligned}$$

where $\vec{Z}$ abbreviates the vector $y_0 Y_2 x$. Thus we need to define terms T_3, S_1, T_1, so that when substituted for $\mathcal{Y}_3, \mathcal{X}_1, \mathcal{Y}_1$ in the matrix of the translation, the result is a theorem of PV^ω.

We define three terms, $\mathcal{T}_3, \mathcal{S}_1$ and an auxiliary term $\mathcal{U}$ by simultaneous recursion, using Theorem 6.12:

$$\mathcal{T}_3(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } y_0 \\ \quad \text{else} \\ Cond(< F, G, H > \dot{-} < t', x, 1 > 0, F, t') \end{cases}$$

$$\mathcal{S}_1(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ Cond(< F, G, H > \dot{-} < t', x, 1 > 0, G, x) \end{cases}$$

$$\mathcal{U}(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } s(0, y_0) \\ \quad \text{else} \\ Cond(< F, G, H > \dot{-} < t', x, 1 > 0, H, 1), \end{cases}$$

where F, G, H are defined as:

$$F \stackrel{d}{\equiv} \begin{cases} \text{If } \mathcal{U}(\lfloor \tfrac{1}{2}x \rfloor) = 0 \text{ then } \begin{cases} \text{if } s(x, Y_2(x\mathcal{T}_3(\lfloor \tfrac{1}{2}x \rfloor))) = 0 \\ \text{then } Y_2(x\mathcal{T}_3(\lfloor \tfrac{1}{2}x \rfloor)) \\ \text{else } \mathcal{T}_3(\lfloor \tfrac{1}{2}x \rfloor) \end{cases} \\ \quad \text{else } \mathcal{T}_3(\lfloor \tfrac{1}{2}x \rfloor) \end{cases}$$

$$G \stackrel{d}{\equiv} Cond(\mathcal{U}(\lfloor \tfrac{1}{2}x \rfloor), x, \mathcal{S}_1(\lfloor \tfrac{1}{2}x \rfloor))$$

$$H \stackrel{d}{\equiv} Cond(\mathcal{U}(\lfloor \tfrac{1}{2}x \rfloor), s(x, Y_2(x\mathcal{T}_3(\lfloor \tfrac{1}{2}x \rfloor))), 1).$$

The term t' stands for a term which can be proved to obey the properties $t \leq t'$, $y_0 \leq t'$, $x \leq y \supset t'(x) \leq t'(y)$ in PV^ω. Such a term can be proved to exist by T205 (Appendix). In the definitions above we have suppressed the free parameters y_0 and Y_2 in the definitions of $\mathcal{T}_3$, $\mathcal{S}_1$ and $\mathcal{U}$, to simplify notation.

Finally we define $\mathcal{T}_1(x) \stackrel{d}{\equiv} \mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor)$.

We first establish that the bounding condition may be dropped by proving the equation $< F, G, H > \dot{-} < t', x, 1 >= 0$. Here and in what follows we use the predicate symbols of IPV and classical propositional calculus, as we are entitled to do by T43, 4.3. It is easy to prove that $G \leq x$ and $H \leq 1$ by induction on x.

To prove $F \leq t'$, for $x > 0$, we first prove $\mathcal{T}_3(x) \leq t'$, which holds by the definition of $\mathcal{T}_3$. Now for $x > 0$, if $\mathcal{U}(\lfloor \frac{1}{2}x \rfloor) = 0$ and $s(x, Y_2(x\mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor))) = 0$ then $Y_2(x\mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor)) \leq t$, so $F \leq t$; in other cases, $F = \mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor) \leq t'(\lfloor \frac{1}{2}x \rfloor) \leq t'(x)$. It follows that $< F, G, H > \leq < t', x, 1 >$ so $< F, G, H > \dot{-} < t', x, 1 > 0 = 0$.

It is now straightforward to prove the following properties of the defined functions:

(A) $\mathcal{U}(x) = 0 \supset \mathcal{U}(\lfloor \frac{1}{2}x \rfloor) = 0$

(B) $\mathcal{U}(x) = 0 \vee \mathcal{U}(x) = 1$

(C) $\mathcal{U}(\lfloor \frac{1}{2}x \rfloor) = 0 \supset \mathcal{S}_1(x) = x$

(D) $\mathcal{U}(x) = 1 \supset \mathcal{T}_3(x) = \mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor)$.

The appropriate instance of 9.3 now follows from the implications:

(E) $\mathcal{U}(x) = 0 \supset s(x, \mathcal{T}_3(x)) = 0$

(F) $\mathcal{U}(x) = 1 \supset s(0, y_0) = 1 \vee$
$[s(\lfloor \frac{1}{2}\mathcal{S}_1(x) \rfloor, \mathcal{T}_1(x)) = 0 \wedge s(\mathcal{S}_1(x), Y_2(\mathcal{S}_1(x)\mathcal{T}_1(x))) = 1]$.

We prove (E) by induction on x. For $x = 0$, $\mathcal{T}_3(x) = y_0$ hence $s(x, \mathcal{T}_3(x)) = \mathcal{U}(x) = 0$. For $x > 0$, assume that (E) holds for $\lfloor \frac{1}{2}x \rfloor$. If $\mathcal{U}(x) = 0$ then $\mathcal{U}(\lfloor \frac{1}{2}x \rfloor) = 0$ by (A) hence $s(x, Y_2(x\mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor))) = \mathcal{U}(x) = 0$, so $s(x, \mathcal{T}_3(x)) = 0$, completing the proof of (E). To prove (F), we again induce on x. For $x = 0$, if $\mathcal{U}(x) = 1$ then $s(0, y_0) = 1$ by definition. For $x > 0$, assume $\mathcal{U}(x) = 1$ and $s(0, y_0) = 0$. Two cases arise. First, if $u(\lfloor \frac{1}{2}x \rfloor) = 0$ then by (E) $s(\lfloor \frac{1}{2}x \rfloor, \mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor)) = 0$, that is, $s(\lfloor \frac{1}{2}x \rfloor, \mathcal{T}_1(x)) = 0$. By (C), $\mathcal{S}_1(x) = x$, so $s(\lfloor \frac{1}{2}\mathcal{S}_1(x) \rfloor, \mathcal{T}_1(x)) = 0$. On the other hand, $s(\mathcal{S}_1(x), Y_2(\mathcal{S}_1(x)\mathcal{T}_1(x))) = s(x, Y_2(x\mathcal{T}_3(\lfloor \frac{1}{2}x \rfloor))) = \mathcal{U}(x) = 1$. Secondly, if $\mathcal{U}(\lfloor \frac{1}{2}x \rfloor) = 1$ then by inductive assumption, $s(\lfloor \frac{1}{2}\mathcal{S}_1(\lfloor \frac{1}{2}x \rfloor) \rfloor, \mathcal{T}_1(\lfloor \frac{1}{2}x \rfloor)) = 0$ and $s(\mathcal{S}_1(\lfloor \frac{1}{2}x \rfloor), Y_2(\lfloor \frac{1}{2}x \rfloor \mathcal{T}_1(\lfloor \frac{1}{2}x \rfloor))) = 1$. By definition, $\mathcal{S}_1(x) = \mathcal{S}_1(\lfloor \frac{1}{2}x \rfloor)$ and by (D), $\mathcal{T}_1(x) = \mathcal{T}_1(\lfloor \frac{1}{2}x \rfloor)$. Thus $s(\lfloor \frac{1}{2}\mathcal{S}_1(x) \rfloor, \mathcal{T}_1(x)) = 0$ and $s(\mathcal{S}_1(x), Y_2(\mathcal{S}_1(x)\mathcal{T}_1(x))) = 1$. This completes the verification of the NP-induction scheme.

In the case of Markov's Principle, MP, an instance of the scheme has the form $\neg\neg\exists\vec{x}A \rightarrow \exists\vec{x}A$, where A is an atomic formula of IPV^ω. Then $(\neg\neg\exists\vec{x}A)^D$ has the form $\exists\vec{x} \sim\sim t = 0$, while $(\exists\vec{x}A)^D$ is $\exists\vec{x}\, t = 0$, so 9.3 holds for MP, by T43. □

The above results on systems containing Markov's Principle are closely connected with some results of Sazonov. In [32], Sazonov defines the theory HPT which consists of all true universal sentences of IPV, together with intuitionistic predicate logic. The class of *Harrop formulas* is defined to be the smallest class $\mathcal{H}$ of formulas satisfying:

(1) All atomic formulas are in $\mathcal{H}$,

(2) $A, B \in \mathcal{H} \Rightarrow (A \wedge B) \in \mathcal{H}$,

(3) $A \in \mathcal{H} \Rightarrow \forall x A \in \mathcal{H}$,

(4) $B \in \mathcal{H} \Rightarrow (A \rightarrow B) \in \mathcal{H}$.

Sazonov defines a theory T to be *constructive* if whenever A is a Harrop formula, B an arbitrary formula and $T \vdash A(\vec{x}) \rightarrow \exists y B(\vec{x}, y)$ then there is a term t of T so that

$$T \vdash A(\vec{x}) \rightarrow B(\vec{x}, t(\vec{x})).$$

T is defined to be $\exists$-*constructive* if the same condition holds with the antecedent $A(\vec{x})$ omitted. Sazonov states as one of his main theorems [32] that $HPT+MP$ is $\exists$-constructive, and that $HPT + MP$ is constructive if and only if $P = NP$.

To illustrate Theorem 9.3 we return to Fermat's theorem (see after 8.6). We formulate this as the formula A:

9.4 $$\forall a \forall n [\forall d (1 \neq d \wedge d \neq n \rightarrow \neg d | n) \rightarrow (0 < a \wedge a < n \rightarrow a^{n-1} \bmod n = 1)]$$

The *Dialectica* translation A^D is essentially A with $\forall d$ removed and the remaining occurrences of d replaced by the term $D(a,n)$, where D has type $0 \rightarrow 0 \rightarrow 0$. Thus if $IPV \vdash A$, then by 9.3 a polynomial time function $D(a,n)$ can be found which satisfies A^D. This function supplies the same information as the term D in $D \circledR B$ (see 8.8). Thus for Fermat's theorem, the *Dialectica* translation is interesting for both the statement itself and its contrapositive, while the realizability of §8 is interesting only for the contrapositive.

In fact, the *Dialectica* interpretation shows

9.5 Proposition *Let A and B be as in 9.4 and 8.7 (Fermat's theorem and its contrapositive). Then $IPV \vdash A$ iff $IPV \vdash B$.*

Proof: By standard intuitionistic reasoning we have $IPV \vdash B \rightarrow A$. Conversely, if $IPV \vdash A$, then by 9.3, $PV^\omega \vdash A^D(D)$ for some closed term D of IPV^ω. It is easy to see that

$$IPV^\omega \vdash A^D(D) \rightarrow B$$

and hence $IPV^\omega \vdash B$. By 7.7, $IPV \vdash B$. $\Box$

10 Applications of Functional Interpretations

In this section we present some further applications of the functional interpretations of the preceding sections. These applications include: conservative extension results, analogues of the main results of §8 for systems based on classical logic, equivalence of IS_2^1 and IS_2^1B, and independence results relating to open questions in complexity theory.

10.1 Theorem *$IPV^\omega + MP$ is a conservative extension of PV^ω. IPV is a conservative extension of PV.*

Proof: The *Dialectica* translation of an equation $u = v$ of PV^ω is $Equ(u, v) = 0$, from which $u = v$ follows by DR126 (Appendix), so the first part of the theorem follows by 9.3. The second part follows from this and 6.18 (see Fig. 1). □

We shall denote by CPV the system obtained from IPV by adding all instances of the law of excluded middle $A \vee \neg A$. CPV^ω is obtained from IPV^ω in the same way. CPV is a conservative extension of S_2^1, by Theorem 4.12. (CPV is equivalent to $S_2^1(PV)$ in [2].)

10.2 Definition *Let A be a formula of IPV or of IPV^ω. The* negative translation *of A, $A^{\neg\neg}$, is defined by induction on the complexity of A as follows:*

(1) *If A is atomic, $A^{\neg\neg} \stackrel{d}{\equiv} A$;*

(2) $(A \wedge B)^{\neg\neg} \stackrel{d}{\equiv} (A^{\neg\neg} \wedge B^{\neg\neg})$;

(3) $(A \to B)^{\neg\neg} \stackrel{d}{\equiv} (A^{\neg\neg} \to B^{\neg\neg})$;

(4) $(\forall x A)^{\neg\neg} \stackrel{d}{\equiv} \forall x A^{\neg\neg}$;

(5) $(A \vee B)^{\neg\neg} \stackrel{d}{\equiv} \neg\neg(A^{\neg\neg} \vee B^{\neg\neg})$;

(6) $(\exists x A)^{\neg\neg} \stackrel{d}{\equiv} \neg\neg\exists x A^{\neg\neg}$.

10.3 Lemma *If $CPV \vdash A$ then $IPV + MP \vdash A^{\neg\neg}$. If $CPV^\omega \vdash A$ then $IPV^\omega + MP \vdash A^{\neg\neg}$.*

Proof: By induction on the length of the proof of A. For the logical axioms and rules of CPV and CPV^ω, this is a well known result, and MP is not necessary (see Troelstra [37] pp. 85–89). The nonlogical axioms other than induction CPV and CPV^ω are quantifier-free, and hence their negative translations are theorems of IPV and IPV^ω, respectively, by 4.5 and 7.4. Thus it suffices to show that the negative translation of the NP induction scheme is derivable in $IPV + MP$. (The argument for IPV^ω is the same.) Since $\neg\exists x F x$

and $\forall x \neg Fx$ are equivalent in intuitionistic logic, the negative translation of an instance of the scheme is equivalent in IPV to a formula of the form:

$$[\neg\neg A(0) \wedge \forall x(\neg\neg A(\lfloor \tfrac{1}{2}x \rfloor) \rightarrow \neg\neg A(x))] \rightarrow \forall z \neg\neg A(z),$$

where A is of the form $(\exists y \leq t)u = v$. By Theorem 4.3, $(\exists y \leq t)u = v$ is equivalent in IPV to the formula $\exists y(Lessequ(y,t) \& Equ(u,v) = 0)$, so that $\neg\neg A \leftrightarrow A$ by MP. Hence the negative translation of the NP induction scheme is derivable in $IPV + MP$. $\Box$

10.4 Theorem *For each Σ_0^b formula A, if $CPV \vdash \exists\vec{x}A$ then $IPV \vdash \exists\vec{x}A$, and if $CPV^\omega \vdash \exists\vec{x}A$ then $IPV^\omega \vdash \exists\vec{x}A$.*

Proof: If A is a Σ_0^b formula of CPV, then by 4.4 there is a term t so $IPV \vdash A \leftrightarrow t = 0$. Hence if $CPV \vdash \exists\vec{x}A$ then $CPV \vdash \exists\vec{x}(t = 0)$ so by 10.3, $IPV + MP \vdash (\exists\vec{x}(t = 0))^{\neg\neg}$. By induction on the number of variables in the list $\vec{x}$, we see that $MP \vdash (\exists\vec{x}(t = 0))^{\neg\neg} \leftrightarrow \exists\vec{x}(t = 0)$. Hence $IPV + MP \vdash \exists\vec{x}(t = 0)$. By Theorem 9.3, there are type 0 zero-order open terms $\vec{s}$ of PV^ω so that $PV^\omega \vdash t[\vec{s}/\vec{x}] = 0$. By 6.16, $PV^\omega \vdash t[\vec{s}^{PV}/\vec{x}] = 0$, so by 6.18, $PV \vdash t[\vec{s}^{PV}/\vec{x}] = 0$. Thus $IPV \vdash \exists\vec{x}(t = 0)$, so $IPV \vdash \exists\vec{x}A$.

The argument for CPV^ω and IPV^ω is the same, except 7.3 is used in place of 4.4, the terms $\vec{s}$ may not be of type 0, and no appeal is needed to results in §6. $\Box$

10.5 Corollary (Compare Buss [2]). *CPV is a conservative extension of PV. Also CPV^ω is a conservative extension of PV^ω.*

Proof: 10.4 and 10.1. $\Box$

The above Corollary can easily be strengthened to say that if B has the form $\forall\vec{x}\exists\vec{y}A$, where A is atomic, and $CPV \vdash B$, then B is a consequence via intuitionistic predicate calculus of the theorems of PV (the NP induction axioms are unnecessary). Hence there is a robustness to the notion of which $\forall\exists$ formulas can be proved using "polynomial time reasoning".

10.6 Corollary *The systems IS_2^1 and IS_2^1B are equivalent.*

Proof: We have already shown in Theorem 1.10 that the $H\Sigma_1^b - PIND$ scheme is derivable in IS_2^1. Thus it suffices to show that the remaining axioms of IS_2^1B are derivable in IS_2^1. Let $(A \rightarrow B)$ be a theorem of S_2^1, where both A and B are $H\Sigma_1^b$ formulas. By Lemma 1.9, we may assume that A and B are both Σ_1^{b+}. Now if $S_2^1 \vdash A \rightarrow B$ then $CPV \vdash A \rightarrow B$, and by 4.11, 4.3, A and B are provably equivalent in IPV to formulas of the form $(\exists x \leq t)(u = 0)$ and $(\exists y \leq t)(v = 0)$. Thus

$$CPV \vdash (x \leq t \wedge u = 0) \rightarrow (\exists y \leq t)(v = 0)$$

so by classical logic,

$$CPV \vdash \exists y[(x \leq t \wedge u = 0) \rightarrow (y \leq t \wedge v = 0)].$$

By 10.4, IPV proves the same formula, hence $IPV \vdash A \rightarrow B$. By 4.12, $IS_2^1 \vdash A \rightarrow B$. □

10.7 Theorem *Let A be a Σ_1^b formula such that $CPV \vdash \forall\vec{x}\exists y A(\vec{x}, y)$. Then there is an n-place function symbol f of PV so that $IPV \vdash A(x, f(\vec{x}))$.*

Proof: If $CPV \vdash \forall\vec{x}\exists y A(\vec{x}, y)$ then $CPV \vdash \exists y A'(\vec{x}, y)$, where $A'(\vec{x}, y)$ is $POS(A(\vec{x}, y))$, as defined in §1. By 4.11, $A'(\vec{x}, y)$ is equivalent in IPV to a formula of the form: $(\exists z \leq t)(u = v)$. Hence by 10.4, $IPV \vdash \forall\vec{x}\exists y A'(\vec{x}, y)$, so $IPV \vdash \forall\vec{x}\exists y A(\vec{x}, y)$ by 1.1. The conclusion of the theorem follows by 8.17. □

Theorem 10.7 is a reformulation of the main result of [2] for the case of the system S_2^1 (the main theorem of [2] also contains similar results for systems S_2^i which correspond to classes of predicates higher in the Meyer-Stockmeyer polynomial hierarchy).

As a final application of the functional interpretations, we prove an independence result relating to the complexity of proofs in classical propositional logic. This result follows as a corollary to the results of Cook [11] on the simulation of arithmetical theorems.

The following definitions are adapted from [16].

10.8 Definitions (1) *If Σ is a finite alphabet, Σ^* is the set of all finite strings over Σ; the length $|x|$ of a string x in Σ^* is the number of occurrences of symbols in x.*

(2) *For Σ_1, Σ_2 finite alphabets, $\mathcal{L}$ is the set of functions $f : \Sigma_1^* \rightarrow \Sigma_2^*$ computable by a deterministic Turing machine in time bounded by a polynomial in the length of the input.*

(3) *If $L \subseteq \Sigma^*$, a* proof system *for L is a function $f : \Sigma_1^* \rightarrow L$, so that f is in $\mathcal{L}$, and the range of f is L.*

(4) *If $f_1 : \Sigma_1^* \rightarrow L$ and $f_2 : \Sigma_2^* \rightarrow L$ are proof systems for L, then f_2* p-simulates *f_1 provided there is a function $g : \Sigma_1^* \rightarrow \Sigma_2^*$ such that g is in $\mathcal{L}$, and $f_2(g(x)) = f_1(x)$ for all x.*

(5) *A proof system f for L is* polynomially bounded *if there is a polynomial p so that for any $x \in L$, there is a y such that $f(y) = x$, and $|y| \leq p(|x|)$.*

The very general definitions just given allow the restatement of problems in complexity theory in proof-theoretic terms.

10.9 Theorem (1) *If a proof system f_2 for L p-simulates a polynomially bounded proof system for L, then f_2 is also polynomially bounded.*

(2) There is a polynomially bounded proof system for the classical tautologies $TAUT$ if and only if $NP = co\text{-}NP$.

Proof: See [16]. □

10.10 Definitions (1) *A* formula *of propositional logic is constructed from the connective set $\{\wedge, \vee, \neg, \rightarrow, \leftrightarrow\}$ and variables $P, Q, R, P_1, \ldots$ by the usual formation rules.*

(2) For formulas $A_1, \ldots, A_n, B$ we write $A_1, \ldots, A_n \models B$ if B is a logical consequence of $A_1, \ldots, A_n$ (i.e. every truth assignment satisfying $A_1, \ldots, A_n$ satisfies B).

(3) If $D_1, \ldots, D_k$ are formulas and $P_1, \ldots, P_k$ distinct propositional variables then $\sigma = (D_1, \ldots, D_k)/(P_1, \ldots, P_k)$ is a substitution *and σA is the formula $A[D_1, \ldots, D_k/ P_1, \ldots, P_k]$.*

(4) A Frege rule *is a rule of the form: from $\sigma A_1, \ldots, \sigma A_n$ infer σB, where $A_1, \ldots, A_n \models B$ and σ is any substitution. If $n = 0$, the rule is an* axiom scheme.

(5) If $\mathcal{F}$ is a set of Frege rules, a derivation of B from $A_1, \ldots, A_n$ in $\mathcal{F}$ is a finite sequence δ of formulas, each of which is either one of $A_1, \ldots, A_n$ or is derived by a rule in $\mathcal{F}$ from earlier formulas in δ and whose last formula is B. We write $A_1, \ldots, A_n \left|\frac{\delta}{\mathcal{F}}\right. B$ if these conditions hold. If $n = 0$ (that is, the derivation contains no hypotheses), δ is said to be a proof *of B, and we write $\left|\frac{\delta}{\mathcal{F}}\right. B$. If there is a derivation of B from $A_1, \ldots, A_n$ in $\mathcal{F}$ we write $A_1, \ldots, A_n \left|\frac{}{\mathcal{F}}\right. B$.*

(6) A Frege system *is a finite set of Frege rules which is inferentially complete, that is, if $A_1, \ldots, A_n \models B$ then $A_1, \ldots, A_n \left|\frac{}{\mathcal{F}}\right. B$.*

(7) If $\mathcal{F}$ is a Frege system, a derivation in $\mathcal{F}$ with extension *of B from $A_1, \ldots, A_n$ is a finite sequence of formulas whose last formula is B and such that each formula in the proof is either (1) one of $A_1, \ldots, A_n$, or (2) derived from earlier formulas in the sequence by one of the rules of $\mathcal{F}$, or (3) a formula of the form $P_k \leftrightarrow A$, where A is any formula, and P_k is a variable which does not occur in A, $A_1, \ldots, A_n$, B or any earlier formula in the sequence. In case (3), we say that the step is introduced by the* extension rule. *The resulting proof system we denote by $e\mathcal{F}$.*

(8) If $\mathcal{F}$ is a Frege system, a derivation in $\mathcal{F}$ with substitution *of B from $A_1, \ldots, A_n$ is a finite sequence of formulas whose last formula is B and such that each formula in the proof is either (1) one of $A_1, \ldots, A_n$, or (2) derived from earlier formulas in the sequence by one of the rules of $\mathcal{F}$ or (3) a formula of the form σA, where A is an earlier formula in the sequence, and σ is any substitution. In case (3), the formula A must satisfy the condition that none of the hypotheses $A_1, \ldots, A_n$ were used in the derivation of A; we say that σA is derived from A by the* rule of substitution. *We write the corresponding proof system as $s\mathcal{F}$.*

The proof systems defined in 10.10 do not fit the general form of 10.8 (3). To accomplish this, we must formulate the concepts of formula and derivations so that formulas and derivations are strings over finite alphabets (so that variables are written as P followed by a string in $\{0,1\}^*$); the function f corresponding to a conventional proof system is then definable as: $f(x) = B$ if x is a proof of B in the system, $f(x) = (P \to P)$ otherwise.

10.11 Theorem *(1) Any two Frege systems p-simulate each other.*

(2) Any two extended Frege systems p-simulate each other.

(3) Any two Frege systems with substitution p-simulate each other.

(4) Any Frege system with substitution p-simulates an extended Frege system.

Proof: For (1), (2), (4), see [16]. For (3) see Reckhow [31]. □

In view of Theorem 10.11, we shall suppose that a fixed Frege system F has been chosen; the corresponding systems with extension and substitution will be denoted by EF and SF. By formalizing the definitions of EF and SF as functions, we can find function symbols $EF(x)$ and $SF(x)$ of PV which express these functions (given a convention on the encoding of strings as binary numerals). We can also find a function symbol $TRUE(x,y)$ so that the equation $TRUE(x,y) = 0$ expresses "x is a formula which is true under truth assignment y."

The sense in which EF can simulate PV is given in the next theorem.

10.12 Theorem *If $t = u$ is an equation of PV, then there is a polynomially growing family of formulas $|t = u|_n$ so that:*

(1) $|t = u|_n$ is a tautology iff $t = u$ is true when restricted to numerals of length n or less;

(2) If $\vdash_{\overline{PV}} t = u$ then there is a polynomial $p(n)$ so that $|t = u|_n$ has an EF proof of length at most $p(n)$, for all n.

Proof: See Dowd [18]. □

Extended Frege systems can p-simulate any proof system whose soundness is provable in PV, as the next theorem shows.

10.13 Theorem *If $PV \vdash TRUE(P(x), y) = 0$, then there is a function symbol G of PV so that $PV \vdash EF(G(x)) = P(x)$.*

Proof: This is the main theorem of Cook [11]. See also Dowd [18]. □

Theorem 10.13 has the following surprising consequence.

10.14 Corollary *EF p-simulates SF.*

Proof: It is possible to prove in PV that $TRUE(SF(x), y) = 0$ by induction on the pair (x, y), so the corollary follows immediately. This result was first proved in Dowd [19]. A direct proof appears in [25]. $\Box$

Theorems 10.11 and 10.14 taken together show that if it could be proved that EF is not polynomially bounded, then the same result would follow for all standard propositional proof systems. This indicates that proving such a result for EF will be very difficult. This impression is reinforced by the result below, which shows that in a precisely defined sense there is no feasibly constructive proof that EF is not polynomially bounded.

It is not immediately obvious how to express this assertion in IPV, since only polynomial-time functions are definable in IPV. However, it is not difficult to see that the function $x^{|y|}$ is definable in PV, and we can use this fact to express the assertion.

10.15 Definition *Let F be a one-place function symbol of IPV. Then we formulate "F is not a polynomially bounded proof system for the tautologies" by:*

$$\neg PB(F) \stackrel{d}{\equiv} \forall x \exists y [TAUT(y) \wedge x \leq y \wedge \forall z (|z| \leq |y|^{|x|} \rightarrow F(z) \neq y)],$$

where $TAUT(y) \stackrel{d}{\equiv} \forall w TRUE(y, w) = 0.$

To confirm that this is a reasonable formulation of the assertion, we can show that various weak proof systems can be proved to be not polynomially bounded in IPV. Thus let $TT(x)$ be a function symbol of IPV so that $TT(x) = y$ if and only if x is an encoding of a truth table proof for a tautology y, or x is not an encoding of a truth table proof, and $y = \ulcorner P \vee \neg P \urcorner$, where $\ulcorner A \urcorner$ is the encoding of a formula A. If we encode a truth-table proof for an n-variable formula as n 1's followed by 0 followed by 2^n digits representing the truth-values of the formula under all assignments of truth-values to the variables, followed by an encoding for the formula, then the length of the encoding exceeds $2^n + 2n + 1$. To show that $\neg PB(TT)$ is provable in IPV, let P_m be the formula: $(P_0 \vee \neg P_0) \wedge (P_1 \vee \neg P_1) \wedge \cdots \wedge (P_m \vee \neg P_m)$. The encoding $\ulcorner P_m \urcorner$ of P_m has length $\Theta(m \log m)$, so there is a constant c so that $|\ulcorner P_m \urcorner| \leq cm^2$. Now for a constant d, we can find a constant f so that:

$$IPV \vdash x \geq f \rightarrow d|x|^4 < 2^{|x|}.$$

There is a PV function symbol G such that $G(n) = \ulcorner P_{|n|} \urcorner$ and

$$IPV \vdash |G(x)| \leq c|x|^2$$
$$IPV \vdash TAUT(G(x))$$

and

$$IPV \vdash TT(z) = G(x) \rightarrow 2^{|x|} < |z|.$$

Now taking $d = 2c$, let f be as above, assume $x \geq f$, and let $y = G(x\#x)$. Then we can prove $y \geq x$ in IPV, as well as

$$\begin{array}{rcl} |y|^{|x|} & \leq & c(|x\#x|^2)^{|x|} \\ & < & (2c|x|^4)^{|x|} \\ & < & 2^{|x|\cdot|x|} \\ & < & 2^{|x\#x|}. \end{array}$$

Thus if $|z| \leq |y|^{|x|}$, then $|z| < 2^{|x\#x|}$, so $TT(z) \neq y$.

It appears that similar proofs can be given in IPV for other weak proof systems such as analytic tableaux and regular resolution. The proof for the system of unrestricted resolution is more interesting. Haken [21] proved that for some $\varepsilon > 0$ if P_n is the tautology in disjunctive normal form of total length $\Theta(n^3)$ which asserts that $n+1$ pigeons cannot fit singly into n holes, then any resolution proof of P_n has length at least $2^{\varepsilon n}$. Thus to prove $\neg PB(RES)$ we can take y to be $P_{c|x|^2}$ for some constant c. This choice of y is a feasible function of x. Although it is not clear that the counting arguments in Haken's proof can be formalized in IPV, a feasibly constructive version using a greedy algorithm appears in [10]. By contrast, for the system EF, we have:

10.16 Theorem *The sentence $\neg PB(EF)$ is unprovable in IPV^ω.*

Proof: If $\neg PB(EF)$ were provable in IPV^ω, then by 8.16 there would be a PV function symbol F so that $IPV^\omega \vdash TAUT(F(x)) \wedge x \leq F(x)$ and

$$IPV^\omega \vdash |z| \leq |F(x)|^{|x|} \rightarrow EF(z) \neq F(x).$$

Thus $IPV^\omega \vdash TRUE(F(x), w) = 0$, hence by 10.1, $PV \vdash TRUE(F(x), w) = 0$. By 10.13, there is a function symbol G of PV so that $PV \vdash EF(G(x)) = F(x)$. Since G is a polynomial-time computable function and $x \leq F(x)$, we can find a k so that

$$|G(k)| \leq |F(k)|^{|k|}.$$

Substituting k for x and $G(k)$ for z in the earlier implication, we derive a contradiction in IPV^ω. $\square$

11 Postscript

After the preliminary version [9] of this paper appeared, some results have been generalized or strengthened by others.

In [2] and [3] Buss introduced systems S_2^i and IS_2^i for $i = 1, 2, \ldots$ in which the provably recursive functions are those polynomial time reducible to the $i-1$ level of the polynomial hierarchy. In the present paper we have considered only the case $i = 1$, but our treatment has been generalized by Harnik [22] to all levels of the hierarchy.

Buss [5] proved a stronger version of Theorem 10.16, but it is still open whether $CPV \vdash \neg PB(EF)$.

The complexity theory of higher type functions was developed in [15], [12] and [13].

APPENDIX

This appendix, which continues the formal development of PV in Section 3, is devoted to a series of theorems and definitions leading to the definition of a multi-variable recursor, the derived rule of multi-variable induction, and the development of the definitions and properties of the basic arithmetic functions. The proofs are somewhat more compressed than in the main text.

As an initial step, we generalize R5 to allow recursion on several digits at once.

D53: For m fixed, $I(m)$ is the term $s_1 \cdots s_1 0$ with value $2^m - 1$. $I(m)$ serves as a standard numeral of (binary) length m.

D54: $Equ\ell(x, m) \stackrel{d}{\equiv} [(x \dot{-} I(m)) \& (I(m) \dot{-} x)]$. $Equ\ell(x, m) = 0$ if and only if $|x|$ (the length of x in binary notation) is m.

T55: $I(m) \dot{-} x \neq 0 \;.\supset.\; xi \dot{-} I(m) = 0$.
Proof: The theorem is established by induction on m. For $m = 0$, T55 is equivalent to

$$Cond(0 \dot{-} x, 0, xi \dot{-} 0) = 0,$$

which follows from T18 and Axiom 3a. Now assume that T55 holds for m. For $m+1$, T55 is: $Cond(I(m)1 \dot{-} x, 0, xi \dot{-} I(m)1) = 0$. We proceed by induction on x:
(55.$\emptyset$): $Cond(I(m)1 \dot{-} 0, 0, i \dot{-} I(m)1)$
$= Cond(I(m)1, 0, 0)$ (T18, T20)
$= 0$ (T1);
(55.j): Assuming $x \neq 0$ if $j = 0$:

$$\begin{aligned} & Cond(I(m)1 \dot{-} xj, 0, xji \dot{-} I(m)1) \\ = \; & Cond(I(m) \dot{-} x, 0, xj \dot{-} I(m)) \qquad \text{(T20)} \\ = \; & 0 \qquad \text{(I.H.).} \quad \text{(See the comment after T72).} \quad \Box \end{aligned}$$

D56:

$$L_m(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(L_m) \dot{-} I(m-1) = 0 \text{ then } t(L_m) \text{ else } I(m-1), \end{cases}$$

$$t(L_m) \equiv Cond(I(m-1) \dot{-} L_m(\lfloor \tfrac{1}{2}x \rfloor), 0, L_m(\lfloor \tfrac{1}{2}x \rfloor)1).$$

$L_m(x)$ is $2^{|x| \bmod m} - 1$; thus $L_m(x)$ in binary notation is a string of 1's of length $|x| \bmod m$.

T57: $t(L_m) \dot{-} I(m-1) = 0$.
Proof: $t(L_m) \dot{-} I(m-1)$
$= Cond(I(m-1) \dot{-} L_m(\lfloor \frac{1}{2}x \rfloor), 0 \dot{-} I(m-1), L_m(\lfloor \frac{1}{2}x \rfloor)1 \dot{-} I(m-1))$ (T2)
$= 0$ (T18, T55). $\square$
This is the bounding inequality for D56.

T58: $x \neq 0 \ .\supset.\ (Equ\ell(x,k) \Rightarrow Equ\ell(xi, k+1)) = 0$.
Proof: Assuming $x \neq 0$:

$$\begin{aligned} & (Equ\ell(x,k) \Rightarrow Equ\ell(xi,k+1)) \\ = \ & [((x \dot{-} I(k))\&(I(k) \dot{-} x)) \Rightarrow ((xi \dot{-} I(k+1))\&(I(k+1) \dot{-} xi))] \\ = \ & [((x \dot{-} I(k))\&(I(k) \dot{-} x)) \Rightarrow ((x \dot{-} I(k))\&(I(k) \dot{-} x))] \qquad \text{(T20)} \\ = \ & 0 \qquad \text{(T43).} \quad \square \end{aligned}$$

T59: $[Equ\ell(L_m(x), 0) \vee \cdots \vee Equ\ell(L_m(x), m-1)] = 0$.
Proof: Abbreviate the L.H.S. as P. Then $P[0/x] = 0$ is immediate, and $(P \Rightarrow P[xi/x]) = 0$ follows from T58, T43 and DR46. Hence, T59 follows by DR52. $\square$

T60: $[Equ\ell(x,m) \Rightarrow \sim Equ\ell(x,n)] = 0$, $m \neq n$.
Proof: We may suppose $m < n$, otherwise we can contrapose the implication by T43. Now $Equ\ell(x,0) = sg(x)$, so the implication for $m = 0$ reduces to $sg(x) \Rightarrow \sim Equ\ell(x,n)$, which is 0 for $n > 0$, by DR14. The general case follows by induction on m. $\square$

DR61 (Bounded multi-digit recursion): For $m > 0$ and g, h, k n-place, $n+2$-place and $n+1$-place functions, there is an $n+1$-place function $R_m[g,h,k]$ so that:

$$R_m[g,h,k](x,\vec{y}) = \begin{cases} \text{If } x = 0 \text{ then } g(\vec{y}) \\ \quad \text{else} \\ \text{if } t(R_m) \dot{-} k(x,\vec{y}) = 0 \text{ then } t(R_m) \\ \qquad\qquad \text{else } k(x,\vec{y}), \end{cases}$$

$$t(R_m) \equiv h(x, \vec{y}, R_m[g,h,k](x \dot{-} I(m), \vec{y})).$$

Proof: We define functions $G_0 \cdots G_{m-1}$ by: $G_j(x,\vec{y}) \stackrel{d}{\equiv} R[g, h_j, k](x, \vec{y})$ where $h_j(x, \vec{y}, a) \stackrel{d}{\equiv}$ If $Equ\ell(L_m(x), j) = 0$ then $h(x,\vec{y},a)$ else a. We can now show that for any i, $0 \leq i < m$, $Equ\ell(L_m(x), i) \ .\supset.\ G_j(x,\vec{y}) = G_j(x \dot{-} I(q), \vec{y})$ where $i - q \equiv j \pmod m$, $0 \leq q < m$. In particular, since (assuming $x \neq 0$),

$$[Equ\ell(L_m(x), j) \Rightarrow Equ\ell(L_m(x \dot{-} 1), j - 1 \ (\mathrm{mod}\ m))] = 0,$$

$$\begin{aligned} Equ\ell(L_m(x), j) \ .\supset.\ G_j(x \dot{-} 1, \vec{y}) &= G_j((x \dot{-} 1) \dot{-} I(m-1), \vec{y}) \\ &= G_j(x \dot{-} I(m), \vec{y}). \end{aligned}$$

Now define:

$$\begin{array}{rcl} R_m[g,h,k](x,\vec{y}) & \stackrel{d}{\equiv} & \text{If } Equ\ell(L_m(x),0) \text{ then } G_0(x,\vec{y}) \\ & & \quad \text{else} \\ & & \text{if } Equ\ell(L_m(x),1) \text{ then } G_1(x,\vec{y}) \\ & & \quad \text{else} \\ & & \quad \vdots \\ & & \text{if } Equ\ell(L_m(x),m-2) \text{ then } G_{m-2}(x,\vec{y}) \\ & & \quad \text{else } G_{m-1}(x,\vec{y}). \end{array}$$

By T60,

$$Equ\ell(L_m(x),j) \;.\supset.\; R_m[g,h,k](x,\vec{y}) = G_j(x,\vec{y}).$$

Hence, assuming $x \neq 0$:

$$\begin{array}{rcl} Equ\ell(L_m(x),j) & .\supset. & R_m[g,h,k](x,\vec{y}) = G_j(x,\vec{y}) \\ & & = h(x,\vec{y},G_j(\lfloor \frac{1}{2}x \rfloor,\vec{y})) \\ & & = h(x,\vec{y},G_j(x \dot{-} I(m),\vec{y})) \\ & & = h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y})). \end{array}$$

By T59 and DR50, $R_m[g,h,k]$ satisfies the recursion equations for multi-digit recursion. □

DR62: Let $R_m[g,h,k]$ be defined as in DR61. If $h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y})) \dot{-} k(x,\vec{y}) = 0$ and $h(0,\vec{y},g(\vec{y})) = g(\vec{y})$ then $R_m[g,h,k](x,\vec{y}) = h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y}))$.
Proof: $R_m[g,h,k](x,\vec{y})$
$= Cond(x,g(\vec{y}),h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y})))$
$= Cond(x,h(0,\vec{y},g(\vec{y})),h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y})))$
$= h(x,\vec{y},R_m[g,h,k](x \dot{-} I(m),\vec{y}))$ (T6, T1) □

D63: $Bit_m(x) \stackrel{d}{\equiv} Parity(x \dot{-} I(m))$. $Bit_m(x)$ is the coefficient of 2^m in x's binary representation.

D64: $Tai\ell(x,s) \stackrel{d}{\equiv} sg^{d_0}(Bit_0(x))\&sg^{d_1}(Bit_1(x))\&\cdots\&sg^{d_{m-1}}(Bit_{m-1}(x))$, for fixed $s = d_{m-1}d_{m-2}\cdots d_0 \in \{0,1\}^m$, where $sg^0 \stackrel{d}{\equiv} sg$ and $sg^1 \stackrel{d}{\equiv} \overline{sg}$. $Tai\ell(x,s) = 0$ iff the last m bits of x's binary notation (padded with leading 0's if necessary) comprise s.

T65: Let $s_1,\ldots,s_{2^m}$ be the members of $\{0,1\}^m$. Then $[Tai\ell(x,s_1) \vee Tai\ell(x,s_2) \vee \cdots \vee Tai\ell(x,s_{2^m})] = 0$.
Proof: By T40, the left hand side is equivalent to the complete DNF tautology whose m atoms are $Bit_j(x)$, $0 \leq j < m$. The result follows by T43. □

T66: $Tai\ell(x,s) \mathrel{.\supset.} (x \dot{-} I(m))s = x$, for each $s \in \{0,1\}^m$.
Proof: By m nested applications of DR28. $\square$

DR67 (Proof by m-digit induction): For $m > 0$, if

$$t[0/x] = u[0/x]$$

and

$$\left.\begin{array}{l} t[xs/x] = v_s[t/a] \\ u[xs/x] = v_s[u/a] \end{array}\right\} \text{ all } s \in \{0,1\}^m$$

then

$$t = u.$$

Proof: For $0 \le j < m$ we want to define $t_j = t[x \dot{-} I(q)/x]$ and $u_j = u[x \dot{-} I(q)/x]$, where q is the least integer ≥ 0 such that either $|x \dot{-} I(q)| \equiv j \pmod{m}$ or $|x \dot{-} I(q)| = 0$. Thus

$$\begin{array}{rcl} t_j & \overset{d}{\equiv} & \text{if } Equ\ell(L_m(x), j) \ \text{ then } t \\ & & \quad \text{else} \\ & & \quad \vdots \\ & & \quad \text{else} \\ & & \text{if } Equ\ell(L_m(x), m-1) \ \text{ then } t[x \dot{-} I(m-j-1)/x] \\ & & \quad \text{else} \\ & & \text{if } Equ\ell(L_m(x), 0) \ \text{ then } t[x \dot{-} I(m-j)/x] \\ & & \quad \text{else} \\ & & \quad \vdots \\ & & \quad \text{else } t[x \dot{-} I(m-1)/x], \end{array}$$

and similarly for u_j.

We now establish $t_j = u_j$ by induction on x. For the base case, we have by the hypotheses for DR67

$$t_j[0/x] = t[0/x] = u[0/x] = u_j[0/x].$$

In general, for each ℓ, $0 \le \ell < m$, if $\ell \not\equiv j-1 \pmod{m}$ then

(1) $\qquad Equ\ell(L_m(x), \ell) \mathrel{.\supset.} t_j[xi/x] = t_j, \qquad i = 0,1.$

On the other hand, if $s_1, \ldots, s_k$, $k = 2^m$, are the members of $\{0,1\}^m$, and $\ell = (j-1) \bmod m$, then

$$(2) \qquad Equ\ell(L_m(x), \ell) \;.\supset.\; t_j[xi/x] = T_{ji}, \quad i = 0, 1,$$

where

$$\begin{array}{rcl} T_{ji} & \overset{d}{\equiv} & \text{if } Tai\ell(xi, s_1) \text{ then } v_{s_1}[t_j/a] \\ & & \quad \text{else} \\ & & \quad \vdots \\ & & \quad \text{else} \\ & & \text{if } Tai\ell(xi, s_{k-1}) \text{ then } v_{s_{k-1}}[t_j/a] \\ & & \quad \text{else } v_{s_k}[t_j/a]. \end{array}$$

Equation (2) is established as follows. Let $P \overset{d}{\equiv} Equ\ell(L_m(x), \ell)$ and $Q_v \overset{d}{\equiv} Tai\ell(xi, s_v)$, $1 \le v \le k = 2^m$. Then $(P\&Q_v) \;.\supset.\; t_j[xi/x] = T_{ji}$, by the hypotheses to DR67, using T66. Hence by DR50, $(P\&Q_1 \vee \cdots \vee P\&Q_k) \;.\supset.\; t_j[xi/x] = T_{ji}$, hence by T43 and T65 we have $P \;.\supset.\; t_j[xi/x] = T_{ji}$, which is (2).

The analogs of (1) and (2) for u_j also hold. Hence we can build a conditional term V_{ji}, based on the cases $Equ\ell(L_m(x), 0), \ldots, Equ\ell(L_m(x), m-1)$, and prove, using DR50, T59, and DR49, that $t_j[si/x] = V_{ji}[t_j/a]$ and $u_j[si/x] = V_{ji}[u_j/a]$, $i = 0, 1$. Hence $t_j = u_j$ by R5.

Now for $j = 0, 1, \ldots, m-1$,

$$\begin{array}{rcl} Equ\ell(L_m(x), j) \;.\supset.\; t & = & t_j \\ & = & u_j \\ & = & u. \end{array}$$

Thus $t = u$ by DR50, T59, and DR49. □

D68: $x \circledast y \overset{d}{\equiv} Cond(y, s_0(x), s_1(x))$.

D69 (Concatenation):

$$x * y = \begin{cases} \text{If } y = 0 \text{ then } x \\ \quad \text{else} \\ \text{if } t(*) \mathbin{\dot{-}} (x1 \boxplus y) = 0 \text{ then } t(*) \text{ else } (x1 \boxplus y), \end{cases}$$

$t(*) = (x * \lfloor \frac{1}{2} y \rfloor) \circledast Parity(y)$.

T70: $y \neq 0 \;.\supset.\; x * y = (x * \lfloor \frac{1}{2} y \rfloor) \circledast Parity(y)$.
Proof: We prove the bounding inequality $t(*) \mathbin{\dot{-}} (x1 \boxplus y) = 0$ by induction on y:

(70.∅): $(x * 0) \circledast Parity(0) \dot{-} (x1 \boxplus 0)$
$= (x0 \dot{-} x1) = 0$;
(70.i): $(x * \lfloor \frac{1}{2} yi \rfloor) \circledast Parity(yi) \dot{-} (x1 \boxplus yi)$
$= (x * y) \circledast i \dot{-} (x1 \boxplus y)0$
$= (x * y) \dot{-} (x1 \boxplus y)$ (T20)
$= 0$ (T23). □

Note that the equality of T70 fails for $y = 0$, since for $x \neq 0$, $x \circledast 0 = x0 \neq x * 0$.

In expressions involving binary operations, parentheses will be omitted with the understanding that they are to be restored by association to the left, so that $x \circledast y \circledast z$ (for example) stands for $(x \circledast y) \circledast z$.

D71: $2^{|x|+|y|} = 1 \boxplus x \boxplus y$.

T72: $x \boxplus y1 \boxplus z = x \boxplus y \boxplus z1$.
Proof: Induce on z:
(72.∅): $x \boxplus y1 \boxplus 0$
$= x \boxplus y1$ (Ax. 4a)
$= (x \boxplus y)0$ (Ax. 4b)
$= x \boxplus y \boxplus 1$ (Ax. 4b);
(72.i): Assuming $z \neq 0$ if $i = 0$:

$$\begin{aligned} x \boxplus y1 \boxplus zi &= (x \boxplus y1 \boxplus z)0 \\ &= (x \boxplus y \boxplus z1)0 \qquad \text{(Ind. Hyp.)} \\ &= (x \boxplus y \boxplus z)00 \\ &= (x \boxplus y \boxplus zi)0 \\ &= x \boxplus y \boxplus zi1. \quad \square \end{aligned}$$

In the preceding proof, the induction step took the form of deriving T72 with zi substituted for z by a chain of equalities in which one step was derived by using T72 itself. This mode of inference is justified by DR17, where $t_1 \equiv x \boxplus y1 \boxplus z$, $t_2 \equiv x \boxplus y \boxplus z1$, $w_i \equiv a0$. It will be employed repeatedly below; a similar justification using R5 or DR17 is possible in each such case.

T73: $x \boxplus y \boxplus z = x \boxplus z \boxplus y$.
Proof: By induction on z:
(73.∅): $x \boxplus y \boxplus 0$
$= x \boxplus y$

$= x \boxplus 0 \boxplus y;$
(73.i): Assuming $z \neq 0$ if $i = 0$:

$$\begin{array}{rcll} x \boxplus y \boxplus zi & = & (x \boxplus y \boxplus z)0 & \\ & = & (x \boxplus z \boxplus y)0 & \text{(Ind. Hyp.)} \\ & = & x \boxplus z \boxplus y1 & \\ & = & x \boxplus z1 \boxplus y & \text{(T72)} \\ & = & x \boxplus zi \boxplus y. \quad \square & \end{array}$$

T74: $2^{|x|+|y|} = 2^{|y|+|x|}$.
Proof: By T73. $\square$

T75: $x \neq 0 \;.\supset.\; x \boxplus y \neq 0$.
Proof: By induction on y:
(75.$\emptyset$): $Cond(x, 0, \overline{sg}(x \boxplus 0))$
$= Cond(x, 0, \overline{sg}(x))$
$= Cond(x, 0, 0)$ (T8)
$= 0;$
(75.i): Assuming $y \neq 0$ if $i = 0$:

$$\begin{array}{rll} & Cond(x, 0, \overline{sg}(x \boxplus yi)) & \\ = & Cond(x, 0, \overline{sg}((x \boxplus y)0)) & \\ = & Cond(x, 0, \overline{sg}(x \boxplus y)) & \text{(Ax. 3b)}. \quad \square \end{array}$$

T76: $2^{|x|+|y|} \neq 0$.
Proof: By T75, DR49, $1 \boxplus x \neq 0$, hence $1 \boxplus x \boxplus y \neq 0$. $\square$

T77: $(x \circledast y) \dot{-} 1 = x$.
Proof: By T27, T2, T1. $\square$

T78: $Bit_\ell(x \circledast y_k \circledast y_{k-1} \circledast \cdots \circledast y_0) = Parity(y_\ell)$, $\ell \leq k$.
Proof: $Bit_\ell(x \circledast y_k \circledast \cdots \circledast y_0)$
$= Parity(x \circledast y_k \circledast \cdots \circledast y_0 \dot{-} I(\ell))$ (D63)
$= Parity(x \circledast y_k \circledast \cdots \circledast y_\ell)$ (T77)
$= Parity[Cond(Parity(y_\ell), x \circledast y_k \circledast \cdots \circledast 0, x \circledast y_k \circledast \cdots \circledast 1)]$ (T29)
$= Cond(Parity(y_\ell), 0, 1)$ (T2, Ax. 1)
$= sg(Parity(y_\ell))$ (D31)
$= Parity(y_\ell)$ (T40). $\square$

T79: $Bit_m((x \dot{-} 1) \circledast i) = Bit_m(x)$, $m > 0$, $i = 0, 1$.
Proof: $Bit_m((x \dot{-} 1) \circledast i)$

$= Parity[((x \dot{-} 1) \circledast i) \dot{-} I(m)]$
$= Parity[(x \dot{-} 1)i \dot{-} I(m)]$ (D68, Ax. 3)
$= Parity[(x \dot{-} 1) \dot{-} I(m-1)]$ (T20)
$= Parity[\lfloor \frac{1}{2}x \rfloor \dot{-} I(m-1)]$ (T27)
$= Parity(x \dot{-} I(m))$ (T21, Ax. 5b)
$= Bit_m(x)$. $\square$

D80: For $k > 0$, we use DR61 to define

$$Puff_k(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(Puff_k) \dot{-} 2^{|x|+|x|} = 0 \text{ then } t(Puff_k) \\ \qquad\qquad \text{else } 2^{|x|+|x|}, \end{cases}$$

$$t(Puff_k) \equiv Puff_k(x \dot{-} I(k)) \circledast Bit_{k-1}(x) \circledast \cdots \circledast Bit_0(x) \circledast 0.$$

$Puff_k(x)$ is the result of "puffing up" the binary representation of x (padded with leading 0's if necessary) by interpolating a zero after each block of k digits. For example, if $x = 1111011$ then $Puff_2(x) = 10110100110$.

For $i = 0, 1$, and t a term, "ti^k" abbreviates t followed by k i's. For example, $I(k+1)$ is 1^{k+1}.

T81: $Puff_k(x) = Puff_k(x \dot{-} I(k)) \circledast Bit_{k-1}(x) \circledast \cdots \circledast Bit_0(x) \circledast 0$.
Proof: We prove $t(Puff_k) \dot{-} 2^{|x|+|x|} = 0$ by k-digit induction on x (DR67):
(81.∅): $Puff_k(0 \dot{-} I(k)) \circledast Bit_{k-1}(0) \circledast \cdots \circledast Bit_0(0) \circledast 0 \dot{-} 2^{|0|+|0|}$
$= 0 \dot{-} 1 = 0$;
(81.s): For $s \in \{0,1\}^k$,

$$\begin{array}{rll} & Puff_k(xs \dot{-} I(k)) \circledast Bit_{k-1}(xs) \circledast \cdots \circledast Bit_0(xs) \circledast 0 \dot{-} 2^{|xs|+|xs|} & \\ = & Puff_k(x) \circledast s_1 \circledast \cdots \circledast s_k \circledast 0 \dot{-} (2^{|xs|+|x|})0^k & \text{(T78, D71, Ax. 4)} \\ = & Puff_k(x) \circledast s_1 \dot{-} 2^{|x|+|xs|} & \text{(T76, T74, T20)} \\ = & (Puff_k(x) \circledast s_1) \dot{-} (2^{|x|+|x|})0^k & \\ = & (Puff_k(x) \dot{-} 2^{|x|+|x|}) \dot{-} I(k-1) & \text{(T76, T20, Ax. 5)} \\ = & 0 & \text{(D80, T22).} \quad \square \end{array}$$

D82: For $k > 1$,

$$Shift_k(x) \stackrel{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(Shift_k) \dot{-} (x * I(k)) = 0 \text{ then } t(Shift_k) \\ \qquad\qquad \text{else } x * I(k), \end{cases}$$

$$t(Shift_k) \equiv \lfloor \tfrac{1}{2} Shift_k(x \dot{-} I(k)) \rfloor \circledast Bit_0(x) \circledast Bit_{k-1}(x) \circledast \cdots \circledast Bit_1(x) \circledast 0.$$

$Shift_k(x)$ is obtained by shifting k places to the left all the digits in x which are coefficients of $2^0, 2^k, 2^{2k}, \ldots$, leaving 0 in the right-most place.

T83: $Shift_k(x) \dot{-} xs = 0$, for $s \in \{0,1\}^k$.
Proof: For $x = 0$,

$$Shift_k(x) \dot{-} xs = 0 \dot{-} xs = 0; \qquad \text{for } x \neq 0,$$

$$\begin{aligned} Shift_k(x) \dot{-} xs &= Shift_k(x) \dot{-} (x * I(k)) && \text{(T20)} \\ &= 0 && \text{(T23).} \quad \square \end{aligned}$$

T84: $Shift_k(x) = \lfloor \tfrac{1}{2} Shift_k(x \dot{-} I(k)) \rfloor \circledast Bit_0(x) \circledast Bit_{k-1}(x) \circledast \cdots \circledast Bit_1(x) \circledast 0$.
Proof: We prove $t(Shift_k) \dot{-} (x * I(k)) = 0$ by k-digit induction on x:
(84.∅): $t(Shift_k)[0/x] \dot{-} (0 * I(k)) = 0 \dot{-} I(k) = 0$;

(84.s): For $s \in \{0,1\}^k$ and $xs \neq 0$:

$$\begin{aligned} & \lfloor \tfrac{1}{2} Shift_k(xs \dot{-} I(k)) \rfloor \circledast Bit_0(xs) \circledast Bit_{k-1}(xs) \circledast \cdots \circledast Bit_1(xs) \circledast 0 \dot{-} (xs * I(k)) \\ = \; & (\lfloor \tfrac{1}{2} Shift_k(x) \rfloor \circledast s_k \circledast s_1 \circledast \cdots \circledast s_{k-1} \circledast 0) \dot{-} (xs * I(k)) \qquad \text{(T78)} \\ = \; & \lfloor \tfrac{1}{2} Shift_k(x) \rfloor \circledast s_k \dot{-} xs \\ = \; & \lfloor \tfrac{1}{2} Shift_k(x) \rfloor \circledast 0 \dot{-} xs \\ = \; & Shift_k(x) \dot{-} xs \\ = \; & 0 \qquad \text{(T83).} \end{aligned}$$

Hence, T84 follows by DR62. $\square$

D85: For $k > 0$, $2^{k|x|} = 1 \boxplus x \boxplus \cdots \boxplus x$ (pad on k x's).

T86: $(x \dot{-} (y \dot{-} 1)) \dot{-} 1 = Cond(y, x \dot{-} 1, x \dot{-} y)$.
Proof: By induction on y:
(86.∅): $(x \dot{-} (0 \dot{-} 1)) \dot{-} 1 = x \dot{-} 1 = Cond(0, x \dot{-} 1, x \dot{-} 0)$;
(86.i): Assuming $y \neq 0$ if $i = 0$:

$$\begin{aligned} (x \dot{-} (yi \dot{-} 1)) \dot{-} 1 &= (x \dot{-} y) \dot{-} 1 \\ &= x \dot{-} yi. \quad \square \end{aligned}$$

DR87: $t \dot{-} u = 0 \vdash (t \dot{-} x) \dot{-} (u \dot{-} x) = 0$.
Proof: By induction on x:

(87.$\emptyset$): $(t \dot{-} 0) \dot{-} (u \dot{-} 0) = t \dot{-} u = 0$;
(87.i): Assuming $x \neq 0$ if $i = 0$;

$$\begin{aligned} & (t \dot{-} xi) \dot{-} (u \dot{-} xi) \\ = \; & [(t \dot{-} x) \dot{-} 1] \dot{-} [(u \dot{-} x) \dot{-} 1] \\ = \; & [(t \dot{-} x) \dot{-} ((u \dot{-} x) \dot{-} 1)] \dot{-} 1 \qquad \text{(T21, T27)} \\ = \; & Cond(u \dot{-} x, (t \dot{-} x) \dot{-} 1, (t \dot{-} x) \dot{-} (u \dot{-} x)] \qquad \text{(T86)} \\ = \; & Cond(u \dot{-} x, ((t \dot{-} x) \dot{-} (u \dot{-} x)) \dot{-} 1, (t \dot{-} x) \dot{-} (u \dot{-} x)) \\ = \; & 0 \qquad \text{(Ind. Hyp.)}. \quad \square \end{aligned}$$

T88: $y \neq 0 \,.\!\supset\!.\; ((x \dot{-} 1) \circledast i \dot{-} y) = x \dot{-} y$, $i = 0, 1$.
Proof: Assuming $y \neq 0$:

$$\begin{aligned} ((x \dot{-} 1) \circledast i \dot{-} y) &= ((x \dot{-} 1) \circledast i \dot{-} (y \dot{-} 1)) \dot{-} 1 \qquad \text{(T86)} \\ &= ((x \dot{-} 1) \circledast i \dot{-} (y \dot{-} 1)1) \\ &= (x \dot{-} 1) \dot{-} (y \dot{-} 1) \qquad \text{(T20)} \\ &= (x \dot{-} (y \dot{-} 1)) \dot{-} 1 \qquad \text{(T21)} \\ &= x \dot{-} y \qquad \text{(T86)}. \quad \square \end{aligned}$$

T89: $y \neq 0 \,.\!\supset\!.\; (Shift_k(x) \dot{-} y0^k) \dot{-} (x \dot{-} y) = 0$.
Proof: $y \neq 0 \,.\!\supset\!.\; xs \dot{-} y0^k = x \dot{-} y$ (T20). Thus T89 follows from T83 and (with $x = y0^k$) DR87.

D90: For $k > 1$,

$$Interleave_k(x, y) \stackrel{d}{\equiv} \begin{cases} \text{If } y = 0 \text{ then } Puff_{k-1}(x) \\ \quad \text{else} \\ \text{if } t(Interleave_k) \dot{-} Puff_{k-1}(x) * 2^{k|y|} = 0 \\ \text{then } t(Interleave_k) \text{ else } Puff_{k-1}(x) * 2^{k|y|}, \end{cases}$$

$$t(Interleave_k) \equiv [Shift_k(Interleave_k(x, \lfloor \tfrac{1}{2} y \rfloor)) \dot{-} 1] \circledast Parity(y).$$

The function $Interleave_k(x, y)$ can be described informally as follows. If the digits of $Interleave_k(x, y)$ (padded with leading zeros if necessary) are broken into blocks of k digits, then the digits of y are formed by concatenating the last digits of the blocks, while the digits of x are formed by concatenating the remaining digits. The recursion equations formalize the process: first "puff up" x by forming $Puff_{k-1}(x)$, then shift the digits of y to the left into the zeros which were interpolated in $Puff_{k-1}(x)$.

T91: $y \neq 0 \mathrel{.\supset.} t(Interleave_k) \dot{-} Puff_{k-1}(x) * 2^{k|y|} = 0.$
Proof: Assuming $y \neq 0$:

$$\begin{aligned}
& \{[Shift_k(Interleave_k(x, \lfloor \tfrac{1}{2} yi \rfloor)) \dot{-} 1] \circledast Parity(yi)\} \dot{-} Puff_{k-1}(x) * 2^{k|yi|} \\
= \; & \{[Shift_k(Interleave_k(x, y)) \dot{-} 1] \circledast i\} \dot{-} (Puff_{k-1}(x) * 2^{k|y|})0^k \qquad \text{(T73)} \\
= \; & Shift_k(Interleave_k(x, y)) \dot{-} (Puff_{k-1}(x) * 2^{k|y|})0^k \qquad \text{(T88)} \\
= \; & [Shift_k(Interleave_k(x, y)) \dot{-} (Puff_{k-1}(x) * 2^{k|y|})0^k] \dot{-} \\
& [Interleave_k(x, y) \dot{-} (Puff_{k-1}(x) * 2^{k|y|})] \qquad \text{(T23)} \\
= \; & 0 \qquad \text{(T89)}. \quad \square
\end{aligned}$$

T92: For $k > 0$,

$$Shift_{k+1}(Puff_k(x)) = Puff_k(x).$$

Proof: By k-digit induction on x:
(92.$\emptyset$): $Shift_{k+1}(Puff_k(0))$
$= Shift_{k+1}(0)$
$= Puff_k(0)$;
(92.s): For $s \in \{0, 1\}^k$:

$$Puff_k(xs) = Puff_k(x) \circledast s_1 \circledast \cdots \circledast s_k \circledast 0 \qquad \text{(T81)}.$$

Hence,

$$\begin{aligned}
& Shift_{k+1}(Puff_k(xs)) \\
= \; & \lfloor \tfrac{1}{2} Shift_{k+1}(Puff_k(x)) \rfloor \circledast 0 \circledast s_1 \circledast \cdots \circledast s_k \circledast 0 \qquad \text{(T81, T84)} \\
= \; & \lfloor \tfrac{1}{2} Puff_k(x) \rfloor \circledast 0 \circledast s_1 \circledast \cdots \circledast s_k \circledast 0 \qquad \text{(I.H.)} \\
= \; & Puff_k(x) \circledast s_1 \circledast \cdots \circledast s_k \circledast 0 \qquad \text{(T81)} \\
= \; & Puff_k(xs) \qquad \text{(T81)}. \quad \square
\end{aligned}$$

T93: For $k > 1$,

$$Interleave_k(x, y) = [Shift_k(Interleave_k(x, \lfloor \tfrac{1}{2} y \rfloor)) \dot{-} 1] \circledast Parity(y).$$

Proof: For $y = 0$:

$$\begin{aligned}
& [Shift_k(Interleave_k(x, \lfloor \tfrac{1}{2} 0 \rfloor)) \dot{-} 1] \circledast Parity(0) \\
= \; & [Shift_k(Puff_{k-1}(x)) \dot{-} 1] \circledast 0 \\
= \; & [Puff_{k-1}(x) \dot{-} 1] \circledast 0 \qquad \text{(T92)} \\
= \; & Puff_{k-1}(x) \qquad \text{(T81)} \\
= \; & Interleave_k(x, 0).
\end{aligned}$$

The theorem follows by DR16, since T91 provides the bounding inequality for D90. $\square$

T94: For $k > 1$, $i, j \in \{0, 1\}$,

$$[Interleave_k(x, yi) \dot{-} 1] \circledast j = Interleave_k(x, yj).$$

Proof: $[Interleave_k(x, yi) \dot{-} 1] \circledast j$
$= [\{[Shift_k(Interleave_k(x, y)) \dot{-} 1] \circledast i\} \dot{-} 1] \circledast j$ (T93)
$= [Shift_k(Interleave_k(x, y)) \dot{-} 1] \circledast j$
$= Interleave_k(x, yj)$ (T93). $\square$

T95: $Bit_\ell(Shift_k(x)) = Bit_\ell(x)$, $0 < \ell < k$.
Proof: $Bit_\ell(Shift_k(x))$
$= Bit_\ell\{\lfloor \frac{1}{2} Shift_k(x \dot{-} I(k)) \rfloor \circledast Bit_0(x) \circledast Bit_{k-1}(x) \circledast \cdots \circledast Bit_1(x) \circledast 0\}$ (T84)
$= Parity(Bit_\ell(x))$ (T78)
$= Bit_\ell(x)$ (D63, T30). $\square$

T96: $Bit_0(Interleave_k(x, y)) = Bit_0(y)$,

$$Bit_\ell(Interleave_k(x, y)) = Bit_{\ell-1}(x),\ 0 < \ell < k.$$

Proof: By induction on y:

(96.∅): $Interleave_k(x, 0)$

$= Puff_{k-1}(x)$ (D90)

$= Puff_{k-1}(x \dot{-} I(k-1)) \circledast Bit_{k-2}(x) \circledast \cdots \circledast Bit_0(x) \circledast 0$ (T81),

hence by T78,

$$Bit_0(Interleave_k(x, 0)) = 0 = Bit_0(0),$$
$$Bit_\ell(Interleave_k(x, 0)) = Bit_{\ell-1}(x);$$

(96.j): $Interleave_k(x, yj)$

$= [Shift_k(Interleave_k(x, y)) \dot{-} 1] \circledast Parity(yj)$ (T93),

hence

$$Bit_0(Interleave_k(x, yj)) = j = Bit_0(yj),$$

and

$$\begin{aligned} & Bit_\ell(Interleave_k(x, yj)) \\ = \ & Bit_\ell\{[Shift_k(Interleave_k(x, y)) \dot{-} 1] \circledast j\} && \text{(T93)} \\ = \ & Bit_\ell\{Shift_k(Interleave_k(x, y))\} && \text{(T79)} \\ = \ & Bit_\ell\{Interleave_k(x, y)\} && \text{(T95)} \\ = \ & Bit_{\ell-1}(x) && \text{(I.H.).} \quad \square \end{aligned}$$

T97: $Shift_k(Interleave_k(xs, y)) = Interleave_k(x, y)s0$, $s \in \{0,1\}^{k-1}$, $k > 1$.
Proof: By induction on y:
(97.$\emptyset$): $Shift_k(Interleave_k(xs, 0))$
$= Shift_k(Puff_{k-1}(xs))$ (D90)
$= Puff_{k-1}(xs)$ (T92)
$= Puff_{k-1}(x)s0$ (T81)
$= Interleave_k(x, 0)s0$ (D90);
(97.i): $Shift_k(Interleave_k(xs, yi))$
$= Shift_k\{Shift_k(Interleave_k(xs, y)) \dot{-} 1 \circledast i\}$ (T93)
$= Shift_k\{Interleave_k(x, y)s0 \dot{-} 1 \circledast i\}$ (I.H.)
$= Shift_k\{Interleave_k(x, y)si\}$
$= \{Shift_k(Interleave_k(x, y)) \dot{-} 1\} \circledast i \circledast s_1 \circledast \cdots \circledast s_{k-1}0$ (T84)
$= Interleave_k(x, yi)s0$ (T93). $\Box$

T98: $Interleave_k(xi_1 \cdots i_{k-1}, yi_k) = Interleave_k(x, y)i_1 \cdots i_k$.
Proof: $Interleave_k(xi_1 \cdots i_{k-1}, yi_k)$
$= [Shift_k(Interleave_k(xi_1 \cdots i_{k-1}, y)) \dot{-} 1] \circledast i_k$ (T93)
$= [Interleave_k(x, y)i_1 \cdots i_{k-1}0 \dot{-} 1] \circledast i_k$ (T97)
$= Interleave_k(x, y)i_1 \cdots i_k$. $\Box$

Theorem 98 is the basic recursion equation for $Interleave_k$; with its help, we can reach our first goal, the definition of the interleaved n-tuple functions.

D99: $< x > \stackrel{d}{\equiv} x$.
$< x_1, \ldots, x_k > \stackrel{d}{\equiv} Interleave_k(< x_1, \ldots, x_{k-1} >, x_k), k > 1$.

T100: $< 0, \ldots, 0 > = 0$.
Proof: By induction on k. For $k = 1$, it is true by definition; for $k > 1$,

$$\begin{aligned} < 0, \ldots, 0, 0 > &= Interleave_k(< 0, \ldots, 0 >, 0) \\ &= Interleave_k(0, 0) \qquad \text{(I.H.)} \\ &= Puff_{k-1}(0) \qquad \text{(D90)} \\ &= 0. \quad \Box \end{aligned}$$

T101: $< x_1i_1, \ldots, x_ki_k > = < x_1, \ldots, x_k > i_1 \cdots i_k$.
Proof: By induction on k. For $k > 1$:

$$\begin{aligned} & < x_1i_1, \ldots, x_ki_k > \\ = \;& Interleave_k(< x_1i_1, \ldots, x_{k-1}i_{k-1} >, x_ki_k) \qquad \text{(D99)} \\ = \;& Interleave_k(< x_1, \ldots, x_{k-1} > i_1 \cdots i_{k-1}, x_ki_k) \qquad \text{(I.H.)} \\ = \;& < x_1, \ldots, x_k > i_1 \cdots i_k \qquad \text{(T98, D99).} \quad \Box \end{aligned}$$

This completes the proof of the basic equalities for the n-tuple functions. We now proceed to the definition of the corresponding projection operators. We begin by defining the auxiliary functions Π^m_L and Π^m_R by m-digit recursion.

D102: For $m > 1$:

$$\Pi^m_L(x) \overset{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(\Pi^m_L) \dot{-} x = 0 \text{ then } t(\Pi^m_L) \text{ else } x, \end{cases}$$

$$t(\Pi^m_L) = \Pi^m_L(x \dot{-} I(m)) \circledast Bit_{m-1}(x) \circledast \cdots \circledast Bit_1(x);$$

$$\Pi^m_R(x) \overset{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(\Pi^m_R) \dot{-} x = 0 \text{ then } t(\Pi^m_R) \text{ else } x, \end{cases}$$

$$t(\Pi^m_R) \equiv \Pi^m_R(x \dot{-} I(m)) \circledast Bit_0(x).$$

T103: $\Pi^m_L(x) = \Pi^m_L(x \dot{-} I(m)) \circledast Bit_{m-1}(x) \circledast \cdots \circledast Bit_1(x)$.
Proof: We prove $t(\Pi^m_L) \dot{-} x = 0$ by m-digit induction (DR67:
(103.∅): $[\Pi^m_L(0 \dot{-} I(m)) \circledast Bit_{m-1}(0) \circledast \cdots \circledast Bit_1(0)]$
$= 0 \dot{-} 0 = 0$;
(103.s): For $s \in \{0, 1\}^m$:

$$\begin{aligned} & [\Pi^m_L(xs \dot{-} I(m)) \circledast Bit_{m-1}(xs) \circledast \cdots \circledast Bit_1(xs)] \dot{-} xs \\ = \; & [\Pi^m_L(x) \circledast s_1 \circledast \cdots \circledast s_{m-1}] \dot{-} xs \\ = \; & \Pi^m_L(x) \dot{-} x \dot{-} 1 \qquad \text{(T20)} \\ = \; & 0 \qquad \text{(T23)}. \quad \Box \end{aligned}$$

T104: $\Pi^m_R(x) = \Pi^m_R(x \dot{-} I(m)) \circledast Bit_0(x)$.
Proof: Similar to T103. □

T105: $\Pi^m_L((x \dot{-} 1) \circledast i) = \Pi^m_L(x)$.
Proof: This follows from T103 by T88. □

T106: $\Pi^m_R((x \dot{-} 1) \circledast i) = (\Pi^m_R(x) \dot{-} 1) \circledast i$.
Proof: By T104 and T88. □

T107: $\Pi^m_L(Shift_m(x)) = \Pi^m_L(x)$.
Proof: By m-digit induction:
(107.∅): $\Pi^m_L(Shift_m(0)) = \Pi^m_L(0)$;
(107. $\mathbf{i}_1 \cdots \mathbf{i}_m$): $\Pi^m_L(Shift_m(xi_1 \cdots i_m))$
$= \Pi^m_L\{(Shift_m(x) \dot{-} 1) \circledast i_m \circledast i_1 \circledast \cdots \circledast i_{m-1} \circledast 0\}$ (T84)

$= \Pi_L^m\{(Shift_m(x) \dot{-} 1) \circledast i_m\} \circledast i_1 \circledast \cdots \circledast i_{m-1}$ (T103)
$= \Pi_L^m\{Shift_m(x)\} \circledast i_1 \circledast \cdots \circledast i_{m-1}$ (T105)
$= \Pi_L^m(x) \circledast i_1 \circledast \cdots \circledast i_{m-1}$ (I.H.)
$= \Pi_L^m(xi_1 \cdots i_m \dot{-} I(m)) \circledast i_1 \circledast \cdots \circledast i_{m-1}$
$= \Pi_L^m(xi_1 \cdots i_m)$ (T103). □

T108: $\Pi_R^m(Shift_m(x)) = \Pi_R^m(x) \circledast 0$.
Proof: By m-digit induction:
(108.$\emptyset$): $\Pi_R^m(Shift_m(0)) = \Pi_R^m(0) = \Pi_R^m(0) \circledast 0$;
(108. $\mathbf{i}_1 \cdots\ \mathbf{i}_m$): $\Pi_R^m(Shift_m(xi_1 \cdots i_m))$
$= \Pi_R^m\{(Shift_m(x) \dot{-} 1) \circledast i_m \circledast i_1 \circledast \cdots \circledast i_{m-1} \circledast 0\}$ (T84)
$= \Pi_R^m\{(Shift_m(x) \dot{-} 1) \circledast i_m\} \circledast 0$ (T104)
$= (\Pi_R^m\{Shift_m(x)\} \dot{-} 1) \circledast i_m \circledast 0$ (T106)
$= ((\Pi_R^m(x) \circledast 0) \dot{-} 1) \circledast i_m \circledast 0$ (I.H.)
$= \Pi_R^m(x) \circledast i_m \circledast 0$
$= \Pi_R^m(xi_1 \cdots i_m \dot{-} I(m)) \circledast i_m \circledast 0$
$= \Pi_R^m(xi_1 \cdots i_m) \circledast 0.$ (T104). □

T109: $\Pi_L^m(x \dot{-} I(m)) = \Pi_L^m(x) \dot{-} I(m-1)$.
Proof: $\Pi_L^m(x \dot{-} I(m))$
$= [\Pi_L^m(x \dot{-} I(m)) \circledast Bit_{m-1}(x) \circledast \ldots \circledast Bit_1(x)] \dot{-} I(m-1)$
$= \Pi_L^m(x) \dot{-} I(m-1)$ (T103). □

T110: $\Pi_R^m(x \dot{-} I(m)) = \Pi_R^m(x) \dot{-} 1$.
Proof: $\Pi_R^m(x \dot{-} I(m))$
$= [\Pi_R^m(x \dot{-} I(m)) \circledast Bit_0(x)] \dot{-} 1$
$= \Pi_R^m(x) \dot{-} 1$ (T104). □

T111: $\Pi_L^m(Puff_{m-1}(x)) = x$.
Proof: By $(m-1)$-digit induction:
(111.$\emptyset$): $\Pi_L^m(Puff_{m-1}(0)) = \Pi_L^m(0) = 0$;
(111. $\mathbf{i}_1 \cdots\ \mathbf{i}_{m-1}$): $\Pi_L^m(Puff_{m-1}(xi_1 \cdots i_{m-1}))$
$= \Pi_L^m[Puff_{m-1}(x) \circledast i_1 \circledast \cdots \circledast i_{m-1} \circledast 0]$ (T81)
$= \Pi_L^m(Puff_{m-1}(x)) \circledast i_1 \circledast \cdots \circledast i_{m-1}$ (T103)
$= xi_1 \cdots i_{m-1}$ (I.H.) □

T112: $\Pi_R^m(Puff_{m-1}(x)) = 0$.
Proof: By induction on x:
(112.$\emptyset$): $\Pi_R^m(Puff_{m-1}(0)) = \Pi_R^m(0) = 0$;
(112. $\mathbf{i}_1 \cdots\ \mathbf{i}_{m-1}$): $\Pi_R^m(Puff_{m-1}(xi_1 \cdots i_{m-1}))$
$= \Pi_R^m[Puff_{m-1}(x) \circledast i_1 \circledast \cdots \circledast i_{m-1} \circledast 0]$ (T81)

$= \Pi_R^m(Puff_{m-1}(x)) \circledast 0$ (T104)
$= 0 \circledast 0$ (I.H.)
$= 0.$ $\square$

T113: $\Pi_L^m[Interleave_m(x,y)] = x.$
Proof: By induction on y:
(113.$\emptyset$): $\Pi_L^m[Interleave_m(x,0)]$
$= \Pi_L^m[Puff_{m-1}(x)]$ (D90)
$= x$ (T111);
(113.i): $\Pi_L^m[Interleave_m(x,yi)]$
$= \Pi_L^m[\{Shift_m(Interleave_m(x,y)) \dot{-} 1\} \circledast i]$ (T93)
$= \Pi_L^m[Shift_m(Interleave_m(x,y))]$ (T105)
$= \Pi_L^m[Interleave_m(x,y)]$ (T107)
$= x$ (I.H.). $\square$

T114: $\Pi_R^m[Interleave_m(x,y)] = y.$
Proof: Induce on y:
(114.$\emptyset$): $\Pi_R^m(Interleave_m(x,0))$
$= \Pi_R^m(Puff_{m-1}(x))$
$= 0$ (T112);
(114.i): $\Pi_R^m(Interleave_m(x,yi))$
$= \Pi_R^m[\{Shift_m(Interleave_m(x,y) \dot{-} 1\} \circledast i]$ (T93)
$= \{\Pi_R^m[Shift_m(Interleave_m(x,y))] \dot{-} 1\} \circledast i$ (T106)
$= \Pi_R^m[Interleave_m(x,y)] \circledast i$ (T108)
$= yi$ (I.H.) $\square$

Theorems 113 and 114 are the basic equalities for Π_L^m and Π_R^m; with these two equalities we can define the general projection operators which operate on interleaved m-tuples.

D115: $\Pi_1^1(x) \stackrel{d}{\equiv} x,$
$\Pi_k^m(x) \stackrel{d}{\equiv} \Pi_k^{m-1}(\Pi_L^m(x))$ for $0 < k < m,$
$\Pi_m^m(x) \stackrel{d}{\equiv} \Pi_R^m(x).$

T116: $\Pi_k^m(< x_1, \ldots, x_m >) = x_k,\ 1 \leq k \leq m.$
Proof: By induction on m.

$$\Pi_1^1(< x >) = \Pi_1^1(x) = x;$$

for $k < m$,

$$\begin{aligned}
\Pi_k^m(< x_1, \ldots, x_m >) &= \Pi_k^m[Interleave_m(< x_1, \ldots, x_{m-1} >, x_m)] && \text{(D99)} \\
&= \Pi_k^{m-1}\Pi_L^m[Interleave_m(< x_1, \ldots, x_{m-1} >, x_m)] && \text{(D115)}
\end{aligned}$$

$$\begin{aligned} &= \Pi_k^{m-1}(< x_1, \ldots, x_{m-1} >) && \text{(T113)} \\ &= x_k && \text{(I.H.)}; \end{aligned}$$

$$\begin{aligned} \Pi_m^m(< x_1, \ldots, x_m >) &= \Pi_R^m(Interleave_m(< x_1, \ldots, x_{m-1} >, x_m)) \\ &= x_m \qquad \text{(T114)}. \quad \Box \end{aligned}$$

T117: For $s \in \{0,1\}^n$ and $1 \leq k \leq n$,

$$\Pi_k^n(xs) = \Pi_k^n(x)s_k.$$

Proof: By induction on n:

$$\Pi_1^1(xs_1) = xs_1 = \Pi_1^1(x)s_1;$$

assuming T117 for $n-1$, and for $k < n$,

$$\begin{aligned} \Pi_k^n(xs) &= \Pi_k^{n-1}\Pi_L^n(xs) && \text{(D115)} \\ &= \Pi_k^{n-1}[\Pi_L^n(x)s_1 \cdots s_{n-1}] && \text{(T103)} \\ &= \Pi_k^{n-1}(\Pi_L^n(x))s_k && \text{(I.H.)} \\ &= \Pi_k^n(x)s_k; \end{aligned}$$

$$\begin{aligned} \Pi_n^n(xs) &= \Pi_R^n(xs) \\ &= \Pi_R^n(x)s_n && \text{(T104)} \\ &= \Pi_n^n(x)s_n. \quad \Box \end{aligned}$$

This completes the basic theorems needed concerning the n-tuple and projection functions. It is now straightforward to prove the derived rules of multi-variable induction and multi-variable recursion.

DR118 (Multi-variable induction)**:** From

$$\left.\begin{array}{l} t[0/x_1 \cdots 0/x_n] = u[0/x_1 \cdots 0/x_n] \\ t[x_1s_1/x_1 \cdots x_ns_n/x_n] = v_s[t/a] \\ u[x_1s_1/x_1 \cdots x_ns_n/x_n] = v_s[u/a] \end{array}\right\} \forall s \in \{0,1\}^n, s = s_1 \cdots s_n,$$

infer $t = u$.

Proof: Let w be a variable not occurring in t or u. Define:

$$t' \equiv t[\Pi_1^n(w)/x_1 \cdots \Pi_n^n(w)/x_n],$$
$$u' \equiv u[\Pi_1^n(w)/x_1 \cdots \Pi_n^n(w)/x_n].$$

Then:

$$
\begin{aligned}
t'[0/w] &= t[\Pi^n_1(0)/x_1 \cdots \Pi^n_n(0)/x_n] \\
&= t[0/x_1 \cdots 0/x_n] && \text{(T100, T116)} \\
&= u[0/x_1 \cdots 0/x_n] \\
&= u'[0/w];
\end{aligned}
$$

for $s \in \{0,1\}^n$,

$$
\begin{aligned}
t'[ws/w] &= t[\Pi^n_1(ws)/x_1 \cdots \Pi^n_n(ws)/x_n] \\
&= t[\Pi^n_1(w)s_1/x_1 \cdots \Pi^n_n(w)s_n/x_n] && \text{(T117)} \\
&= v'_s[t'/a],
\end{aligned}
$$

where $v'_s \equiv v_s[\Pi^n_1(w)/x_1 \ldots \Pi^n_n(w)/x_n]$, and similarly, $u'[ws/w] = v'_s[u'/a]$. Hence, by multi-digit recursion on the variable w (DR67), $t' = u'$. Hence,

$$
\begin{aligned}
t &= t[\Pi^n_1(< x_1, \ldots, x_n >)/x_1 \cdots \Pi^n_n(< x_1, \ldots, x_n >)/x_n] && \text{(T116)} \\
&= t'[< x_1, \ldots, x_n > /w] \\
&= u'[< x_1, \ldots, x_n > /w] \\
&= u[\Pi^n_1(< x_1, \ldots, x_n >)/x_1 \cdots \Pi^n_n(< x_1, \ldots, x_n >)/x_n] \\
&= u. \quad \square
\end{aligned}
$$

T119: $< x_1, \ldots, x_n > \dot{-} I(n) = < \lfloor \frac{1}{2}x_1 \rfloor \cdots \lfloor \frac{1}{2}x_n \rfloor >$.
Proof: By multi-variable induction on $x_1 \cdots x_n$:
(119.∅): $< 0, \ldots, 0 > \dot{-} I(n)$
$= 0 \dot{-} I(n)$
$= 0$
$= < 0, \ldots, 0 >$ (T100)
$= < \lfloor \frac{1}{2}0 \rfloor \cdots \lfloor \frac{1}{2}0 \rfloor >$;
(119.s): $< x_1 s_1, \ldots, x_n s_n > \dot{-} I(n)$
$= < x_1, \ldots, x_n > s \dot{-} I(n)$ (T101)
$= < x_1, \ldots, x_n >$
$= < \lfloor \frac{1}{2}x_1 s_1 \rfloor \cdots \lfloor \frac{1}{2}x_n s_n \rfloor >$. $\square$

DR120 (Bounded multi-variable recursion): If g, h and k are n-place, $(m+n+1)$-place and $(m+n)$-place functions respectively, there is an $(m+n)$-place function $R^m[g,h,k]$ which satisfies:

$$
R^m[g,h,k](\vec{x},\vec{y}) = \begin{cases} \text{If } < x_1, \ldots, x_m > = 0 \text{ then } g(\vec{y}) \\ \quad \text{else} \\ \text{if } t \dot{-} k(\vec{x},\vec{y}) = 0 \text{ then } t \\ \quad \text{else } k(\vec{x},\vec{y}), \end{cases}
$$

$$t \equiv h(\vec{x}, \vec{y}, R^m[g,h,k](\lfloor \tfrac{1}{2}x_1 \rfloor \cdots \lfloor \tfrac{1}{2}x_m \rfloor, \vec{y})).$$

Proof: Define:

$$h'(w, \vec{y}, a) \equiv h(\Pi_1^m(w), \ldots, \Pi_m^m(w), \vec{y}, a),$$
$$k'(w, \vec{y}) \equiv k(\Pi_1^m(w), \ldots, \Pi_m^m(w), \vec{y}).$$

Let $R_m[g, h', k']$ be defined by bounded multi-digit recursion (DR61). Then we have:

$$R_m[g,h',k'](w,\vec{y}) = \begin{cases} \text{If } w = 0 \text{ then } g(\vec{y}) \\ \quad \text{else} \\ \text{if } t' \dot{-} k'(w,\vec{y}) = 0 \text{ then } t' \\ \quad \text{else } k'(w,\vec{y}), \end{cases}$$

$$t' \equiv h'(w, \vec{y}, R_m[g,h',k'](w \dot{-} I(m), \vec{y})).$$

Now define:

$$R^m[g,h,k](\vec{x},\vec{y}) \overset{d}{\equiv} R_m[g,h',k'](< x_1, \ldots, x_m >, \vec{y}).$$

Substituting in the previous equality, we obtain:

$$R^m[g,h,k](\vec{x},\vec{y}) = \begin{cases} \text{If } < x_1, \ldots, x_m > = 0 \text{ then } g(\vec{y}) \\ \quad \text{else} \\ \text{if } t'[< x_1, \ldots, x_m > / w] \dot{-} k'(< x_1, \ldots, x_m >, \vec{y}) = 0 \\ \text{then } t'[< x_1, \ldots, x_m > / w] \text{ else } k'(< x_1, \ldots, x_m >, \vec{y}). \end{cases}$$

Now we compute:

$$\begin{array}{ll} & t'[< x_1, \ldots, x_m > / w] \\ = & h'(< x_1, \ldots, x_m >, \vec{y}, R_m[g,h',k'](< x_1, \ldots, x_m > \dot{-} I(m), \vec{y})) \\ = & h(x_1, \ldots, x_m, \vec{y}, R_m[g,h',k'](< \lfloor \tfrac{1}{2}x_1 \rfloor, \ldots, \lfloor \tfrac{1}{2}x_m \rfloor >, \vec{y})) \qquad \text{(T116, T119)} \\ = & h(x_1, \ldots, x_m, \vec{y}, R^m[g,h,k](\lfloor \tfrac{1}{2}x_1 \rfloor, \ldots, \lfloor \tfrac{1}{2}x_m \rfloor, \vec{y})), \end{array}$$

and

$$\begin{array}{ll} & k'(< x_1, \ldots, x_m >, \vec{y}) \\ = & k(\Pi_1^m(< x_1, \ldots, x_m >) \cdots \Pi_m^m(< x_1, \ldots, x_m >), \vec{y}) \\ = & k(x_1, \ldots, x_m, \vec{y}) \qquad \text{(T116)}, \end{array}$$

so $R^m[g,h,k]$ satisfies the required recursion equation. □

The remainder of the Appendix is devoted to introducing basic arithmetical functions and proving their fundamental properties, using the tools of multi-variable recursion and induction. The equality functional with which we begin is a good example of a function which is easy to define by double induction, while no direct definition by single-variable induction is apparent.

D121 (Equality):

$$Equ(x,y) \stackrel{d}{\equiv} \begin{cases} \text{If } < x,y >= 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(Equ) \dot{-} 1 = 0 \text{ then } t(Equ) \text{ else } 1, \end{cases}$$

$$t(Equ) \equiv [Equ(\lfloor \tfrac{1}{2}x \rfloor, \lfloor \tfrac{1}{2}x \rfloor)\&(Parity(x) \Leftrightarrow Parity(y))].$$

The bounding inequality for D121 is immediate from T45.

T122: $Equ(x,x) = 0$.
Proof: By induction on x:
(122.∅): $Equ(0,0) = 0$;
(122.i): $Equ(xi,xi)$
$= [Equ(x,x)\&(i \Leftrightarrow i)]$ (I.H.)
$= 0$. □

T123: $Equ(x,y) = Equ(y,x)$.
Proof: By induction on x and y:
(123.∅): $Equ(0,0) = 0$;
(123.ij): $Equ(xi,yj)$
$= Equ(x,y)\&(Parity(xi) \Leftrightarrow Parity(yj))$
$= Equ(y,x)\&(Parity(yj) \Leftrightarrow Parity(xi))$ (I.H., T43)
$= Equ(yj,xi)$. □

T124: $[(Equ(x,y)\&Equ(y,z)) \supset Equ(x,z)] = 0$.
Proof: By induction on x, y and z, using T43. □

T125: $Equ(x,y) \,.\!\supset\!.\; x = y$.
Proof: By induction on x and y:
(125.∅): $Cond(Equ(0,0),0,0)$
$= Cond(0,0,0)$
$= 0$;
(125.ij): For $i \neq j$,

$$\begin{aligned} & Cond(Equ(xi,yj),xi,yj) \\ = \; & Cond(1,xi,yj) \\ = \; & yj; \end{aligned}$$

for $i = j$,

$$\begin{aligned} & Cond(Equ(xi,yj),xi,yj) \\ = \; & Cond(Equ(x,y),xi,yi) \\ = \; & [Cond(Equ(x,y),x,y)]i \\ = \; & yi \qquad \text{(I.H.).} \quad \square \end{aligned}$$

DR126: $Equ(t,u) = 0 \vdash t = u$.
Proof: By T125. $\square$

DR127: $t = u \vdash Equ(t,u) = 0$.
Proof: By T122. $\square$

T128: $[Equ(x,y) \Rightarrow Equ(f(x,\vec{z}), f(y,\vec{z})] = 0$.
Proof: $[Equ(x,y) \Rightarrow Equ(f(x,\vec{z}), f(y,\vec{z})]$
$= [Equ(x,y) \Rightarrow Equ(f(Cond(Equ(x,y),y,x),\vec{z}), f(y,\vec{z})]$ (T123, T125)
$= [Equ(x,y) \Rightarrow Equ(f(y,\vec{z}), f(y,\vec{z})]$ (D35, T7)
$= 0$. $\square$

The theorems and derived rules T122 to T128 show that the equality function Equ has all the classical properties of the identity relation. This together with T43 and DR49 makes possible the employment of classical truth-functional reasoning with propositional formulas built from equalities in proving theorems of PV. In the remaining proofs, such reasoning will be employed without special mention.

DR129: From $P[0/x_1 \cdots 0/x_n] = 0$ and $[P \Rightarrow P[x_1s_1/x_1 \cdots x_ns_n/x_n]] = 0$, $\forall s \in \{0,1\}^n$ infer $P = 0$.
Proof: This is proved in the same way as DR52, using DR118 in place of single-variable induction. $\square$

D130: $x \sqsubseteq y \overset{d}{\equiv} Equ(x \dot{-} y, 0)$. This is the characteristic function of the relation $|x| \leq |y|$.

T131: $[((x \sqsubseteq y) \& (y \sqsubseteq z)) \Rightarrow (x \sqsubseteq z)] = 0$.
Proof: By induction on x, y and z, using T20. $\square$

T132: $x \neq 0 \;.\!\supset\!.\; (S(x \dot{-} 1) \dot{-} x) = 0$.
Proof: By induction on x. Assuming $xi \neq 0$:

$$\begin{array}{ll} & S(xi \dot{-} 1) \dot{-} xi \\ = & Sx \dot{-} xi \\ = & Cond(Parity(x), s_1(\lfloor \frac{1}{2}x \rfloor) \dot{-} xi, s_0(S(\lfloor \frac{1}{2}x \rfloor)) \dot{-} xi) \qquad \text{(T26, T3)} \\ = & Cond(Parity(x), \lfloor \frac{1}{2}x \rfloor \dot{-} x, S(\lfloor \frac{1}{2}x \rfloor) \dot{-} x) \qquad \text{(T20)} \\ = & 0 \qquad \text{(I.H., DR15).} \quad \square \end{array}$$

D133:

$$|x| \overset{d}{\equiv} \begin{cases} \text{If } x = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } S(|\lfloor \frac{1}{2}x \rfloor|) \dot{-} x = 0 \text{ then } S(|\lfloor \frac{1}{2}x \rfloor|) \text{ else } x. \end{cases}$$

T134: $x \neq 0 \ .\supset.\ S(|\lfloor \frac{1}{2}x \rfloor|) \dot{-} x = 0.$
Proof: By induction on x:
(134.i): Assume $xi \neq 0$. Then:

$$\begin{array}{ll} & S(|\lfloor \frac{1}{2}xi \rfloor|) \dot{-} xi \\ = & S(|x|) \dot{-} xi \\ = & Cond(Parity(|x|), s_1(|x| \dot{-} 1) \dot{-} xi, s_0 S(|x| \dot{-} 1) \dot{-} xi) \qquad \text{(T26, T3)} \\ = & Cond(Parity(|x|), (|x| \dot{-} 1) \dot{-} x, S(|x| \dot{-} 1) \dot{-} x). \qquad \text{(T20)} \end{array}$$

By T23, $|x| \sqsubseteq x$, so by T131, T132, if $|x| \neq 0$ then $S(|x| \dot{-} 1) \sqsubseteq |x| \sqsubseteq x$; also, $(|x| \dot{-} 1) \dot{-} x = (|x| \dot{-} x) \dot{-} 1 = 0 \dot{-} 1 = 0$. Hence, $S(|\lfloor \frac{1}{2}xi \rfloor|) \dot{-} xi = 0$ by DR15. □

T135: $|x| = Cond(x, 0, S(|\lfloor \frac{1}{2}x \rfloor|)).$
Proof: By D133, T134. □

D136:

$$Less(x,y) \overset{d}{\equiv} \begin{cases} \text{If } x = y = 0 \text{ then } 1 \\ \quad \text{else} \\ Cond(t(Less) \dot{-} 1, t(Less), 1), \end{cases}$$

$$t(Less) \equiv [Equ(\lfloor \tfrac{1}{2}x \rfloor, \lfloor \tfrac{1}{2}y \rfloor) \& \sim (Parity(x) \Rightarrow Parity(y)) \ .\vee.\ Less(\lfloor \tfrac{1}{2}x \rfloor, \lfloor \tfrac{1}{2}y \rfloor)].$$

The bounding inequality for D136 is immediate by T45.

D137: $Lessequ(x,y) \overset{d}{\equiv} [Less(x,y) \vee Equ(x,y)].$

T138: $[Less(x,y) \& Less(y,z) \ .\Rightarrow.\ Less(x,z)] = 0.$
Proof: By induction on x, y and z. □

T139: $[Lessequ(x,y) \& Lessequ(y,z) \ .\Rightarrow.\ Lessequ(x,z)] = 0.$
Proof: By T124, T138. □.

T140: $Less(x,x) = 1.$
Proof: By induction on x. □

T141: $[Lessequ(x,y) \& Lessequ(y,x) \ .\Rightarrow.\ Equ(x,y)] = 0.$
Proof: Assume $Lessequ(x,y) = Lessequ(y,x) = 0$. If $Less(x,y) = Less(y,x) = 0$ then $Less(x,x) = 0$ by T138, contradicting T140. Hence, $Equ(x,y) = 0$. □

T142: $[Lessequ(x,y) \vee Lessequ(y,x)] = 0.$
Proof: Induction on x, y. □

T143: $x \neq 0 \ .\Rightarrow.\ Less(0,x) = 0.$
Proof: By induction on x. Assume $x \neq 0$. If $\lfloor \frac{1}{2}x \rfloor = 0$ then $x = 1$, so $Less(0,x) = 0$; if $\lfloor \frac{1}{2}x \rfloor \neq 0$ then $Less(0, \lfloor \frac{1}{2}x \rfloor) = 0$ (I.H.) so $Less(0,x) = 0$. □

T144: $Lessequ(0, x) = 0$.
Proof: By T143, DR16. $\square$

T145: $Lessequ(Sx, y) = Less(x, y)$.
Proof: By induction on x and y. $\square$

T146: $Less(x, Sy) = Lessequ(x, y)$.
Proof: By induction on x and y. $\square$

T147: $[Lessequ(x, y) \;.\!\Rightarrow\!.\; Equ(x, y) \vee Lessequ(Sx, y)] = 0$.
Proof: By D137, T145. $\square$

T148: $Equ(x, Sx) = 1$.
Proof: If $Equ(x, Sx) = 0$ then $Lessequ(Sx, x) = 0$, hence $Less(x, x) = 0$ by T145; this contradicts T140. $\square$

T149: $Less(Sx, Sy) = Less(x, y)$.
Proof: By T145, T146. $\square$

T150: $Lessequ(x, 0) = Equ(x, 0)$.
Proof: By T144, T141. $\square$

T151: $[Lessequ(x, y) \Rightarrow Lessequ(|x|, |y|)] = 0$.
Proof: By induction on x and y. The case $x = y = 0$ is easy. For the induction step consider xi, yj. If $xi = 0$ then $Lessequ(|xi|, |yj|) = 0$ by T144; if $yj = 0$ then if $Lessequ(xi, yj) = 0$, $Equ(xi, yj) = 0$ by T150, hence $Lessequ(|xi|, |yj|)$ by T122. Thus, we can assume $xi \neq 0$, $yj \neq 0$, $Lessequ(xi, yj)$. Then $Lessequ(x, y) = 0$, so $Lessequ(|x|, |y|) = 0$ by I.H. Thus $Lessequ(S(|x|), S(|y|)) = 0$ by T149, so $Lessequ(|xi|, |yj|) = 0$. $\square$

T152: $y \neq 0 \;.\!\supset\!.\; (x0 \boxplus y) = x \boxplus y0$.
Proof: By induction on y. $\square$

D153 (Addition):

$$x + y \overset{d}{\equiv} \begin{cases} \text{If } x = y = 0 \text{ then } 0 \\ \quad \text{else} \\ \text{if } t(+) \mathbin{\dot{-}} (1 \boxplus x \boxplus y) = 0 \text{ then } t(+) \text{ else } 1 \boxplus x \boxplus y, \end{cases}$$

$$t(+) \equiv \begin{cases} \text{If } Parity(x) = 0 \text{ then } (\lfloor \tfrac{1}{2}x \rfloor + \lfloor \tfrac{1}{2}y \rfloor) \circledast Parity(y) \\ \quad \text{else} \\ \text{if } Parity(y) = 0 \text{ then } (\lfloor \tfrac{1}{2}x \rfloor + \lfloor \tfrac{1}{2}y \rfloor)1 \\ \quad \text{else } S(\lfloor \tfrac{1}{2}x \rfloor + \lfloor \tfrac{1}{2}y \rfloor)0. \end{cases}$$

T154: $t(+) \dot{-} (1 \boxplus x \boxplus y) = 0$.
Proof: By induction on x and y. The case $x = y = 0$ is easy.
(154.00): The expression simplifies to: $(x+y) \circledast 0 \dot{-} (1 \boxplus x0 \boxplus y0)$. If $x = 0$ and $y = 0$, this is $0 \dot{-} 1 = 0$; if $x \neq 0$ and $y = 0$, this is:

$$\begin{array}{rcll} (x+0) \circledast 0 \dot{-} (1 \boxplus x)0 & = & (x+0) \dot{-} (1 \boxplus x) & \\ & = & 0 & \text{(I.H.)}. \end{array}$$

So, we can assume $y \neq 0$. Hence:

$$\begin{array}{rll} & (x+y) \circledast 0 \dot{-} (1 \boxplus x0 \boxplus y0) & \\ = & (x+y) \circledast 0 \dot{-} (1 \boxplus x0 \boxplus y)0 & \\ = & (x+y) \dot{-} (1 \boxplus x0 \boxplus y) & \\ = & (x+y) \dot{-} (1 \boxplus x \boxplus y)0 & \text{(T152)} \\ = & 0 \qquad \text{(T23)}; & \end{array}$$

(154.01): Similarly;
(154.10): Similarly;
(154.11): We need to prove:

$$S(x+y)0 \dot{-} (1 \boxplus x1 \boxplus y1) = 0,$$

that is, $S(x+y) \dot{-} (1 \boxplus x \boxplus y)0 = 0$. We have

$$\begin{array}{rl} & s_1(x+y) \dot{-} (1 \boxplus x \boxplus y)0 \\ = & (x+y) \dot{-} (1 \boxplus x \boxplus y) \\ = & 0 \qquad \text{(T23)}, \end{array}$$

i.e. $s_1(x+y) \sqsubseteq (1 \boxplus x \boxplus y)0$, hence

$$\begin{array}{rcll} S(x+y) & \sqsubseteq & s_1(x+y) & \text{(T25)} \\ & \sqsubseteq & (1 \boxplus x \boxplus y)0, & \end{array}$$

so $S(x+y) \dot{-} (1 \boxplus x \boxplus y)0 = 0$. □

T154 is the bounding inequality for addition, so the recursion equations for + hold in the unconditional form. In what follows, proofs will simply be omitted if they are straightforward inductions.

T155: $x + 0 = x$. □

T156: $Sx = x + 1$.
Proof: By T155, T26. □

T157: $x + y = y + x$. $\square$

T158: $x + Sy = S(x + y)$. $\square$

T159: $(x + y) + z = x + (y + z)$.
Proof: By induction on x, y and z, using T157, T158. $\square$

T160: $[Equ(x + z, y + z) \Leftrightarrow Equ(x, y)] = 0$.
Proof: By induction on x, y and z, using $Equ(Sx, Sy) = Equ(x, y)$, which follows from T149. $\square$

T161: $Less(x + z, y + z) = Less(x, y)$.
Proof: By induction on x, y and z, using T146, T145 and T149. $\square$

T162: $Lessequ(x + z, y + z) = Lessequ(x, y)$.
Proof: By T160, T161. $\square$

T163: $y \neq 0 \;.\supset.\; x \dot{-} (y \boxplus z) = (x \dot{-} y) \dot{-} z$. $\square$

T164: $(x \sqsubseteq x) = 0$. $\square$

T165: $x \neq 0 \;.\supset.\; Equ(x, 0) = 1$. $\square$

T166: $sg(x) = Equ(x, 0)$. $\square$

In view of T166, $(x \sqsubseteq y) = Equ(x \dot{-} y, 0)$; hence we can use $(x \sqsubseteq y)$ as an atomic formula interchangeable with $x \dot{-} y = 0$ when reasoning in propositional logic.

T167: $[x \sqsubseteq y \;.\Rightarrow.\; x \dot{-} z \sqsubseteq y \dot{-} z] = 0$.
Proof: By induction on z:
(167.i): Assume that $x \sqsubseteq y$ and that T167 holds for z. If $y \dot{-} zi = 0$ then $y \sqsubseteq zi$, hence $x \sqsubseteq zi$ (T131), so that $x \dot{-} zi = 0$, hence $(x \dot{-} zi) \dot{-} (y \dot{-} zi) = 0$. If $y \dot{-} zi \neq 0$ then:

$$\begin{array}{rcl} (x \dot{-} zi) \dot{-} (y \dot{-} zi) & = & (x \dot{-} z) \dot{-} \lfloor \frac{1}{2}(y \dot{-} z) \rfloor 1 \\ & = & (x \dot{-} z) \dot{-} (y \dot{-} z) \\ & = & 0 \qquad\qquad \text{(I.H.)}. \quad \square \end{array}$$

T168: $[x \sqsubseteq y \;.\Rightarrow.\; s_i(x) \sqsubseteq s_i(y)] = 0$.
Proof: Assume $x \sqsubseteq y$. If $y = 0$ then $x = 0$ (T166), so $s_i(x) = s_i(0)$. If $y \neq 0$, then since $x \dot{-} y = 0$, $s_i(x) \dot{-} s_i(y) = x \dot{-} y = 0$, so $s_i(x) \sqsubseteq s_i(y)$. $\square$

T169: $[Equ(x0, 0) \Rightarrow Equ(x, 0)] = 0$.
Proof: By T128. $\square$

T170: $[Equ(x \boxplus y, 0) \Rightarrow Equ(x, 0)] = 0$. $\square$

T171: $Equ(x \# y, 0) = 1$.
Proof: By induction on y, using T170. $\square$

T172: $x \boxplus y1 = x0 \boxplus y$. $\square$

T173: $sg(Sx) = 1$. $\square$

T174: $sg(|x|) = sg(x)$. $\square$

T175: $(x \sqsubseteq y) = Lessequ(|x|, |y|)$.
Proof: By induction on x and y. For $y = 0$,

$$\begin{aligned}
(x \sqsubseteq 0) &= sg(x \dot{-} 0) \\
&= sg(x) \\
&= sg(|x|) \qquad \text{(T174)} \\
&= Equ(|x|, 0) \qquad \text{(T166)} \\
&= Lessequ(|x|, 0);
\end{aligned}$$

similarly if $x = 0$.
(175.ij): Assuming $yj \neq 0$, $xi \neq 0$:

$$\begin{aligned}
(xi \sqsubseteq yj) &= sg(xi \sqsubseteq yj) \qquad \text{(T39)} \\
&= sg(x \dot{-} y) \\
&= x \sqsubseteq y \\
&= Lessequ(|x|, |y|) \qquad \text{(I.H.)} \\
&= Lessequ(S(|x|), S(|y|)) \qquad \text{(T145, T146)} \\
&= Lessequ(|xi|, |yj|). \quad \square
\end{aligned}$$

T176: $y \neq 0 \; . \supset . \; x \dot{-} (y1 \boxplus z) = x \dot{-} (y \boxplus z)1$.
Proof: By T163. $\square$

T177: $x \neq 0 \; . \supset . \; (x + y) \dot{-} (x \boxplus y) = 0$.
Proof: By induction on x and y. The case $y = 0$ is trivial.
(177.00): $(x0 + y0) \dot{-} (x0 \boxplus y0)$
$= (x + y)0 \dot{-} (x \boxplus y00)$ (T152)
$= (x + y)0 \dot{-} (x \boxplus y)00$
$= (x + y) \dot{-} (x \boxplus y)0$ (T75)
$= 0$ (I.H.);

(177.01): Similarly;
(177.10): $(x1+y0) \dot{-} (x1 \boxplus y0)$
$= (x+y)1 \dot{-} (x1 \boxplus y)0$
$= (x+y) \dot{-} (x1 \boxplus y)$ (T75)
$= ((x+y) \dot{-} (x \boxplus y)) \dot{-} 1$ (T176)
$= 0$ (I.H.);
(177.11): $(x1+y1) \dot{-} (x1 \boxplus y1)$
$= S(x+y)0 \dot{-} (x1 \boxplus y)0$
$= S(x+y) \dot{-} (x1 \boxplus y)$ (T75)
$= S(x+y) \dot{-} (x \boxplus y)1$ (T176);
by T25, $S(x+y) \sqsubseteq (x+y)1$, so by T167,

$$\begin{array}{rcl} S(x+y) \dot{-} (x \boxplus y)1 & \sqsubseteq & (x+y)1 \dot{-} (x \boxplus y)1 \\ & = & (x+y) \dot{-} (x \boxplus y) \\ & = & 0 \qquad \text{(I.H.)} \quad \Box \end{array}$$

T178: $Equ(Sx, 0) = 1$.
Proof: By T166, T173. □

T179: $[Equ(x+y,0) \;.\!\Rightarrow\!.\; Equ(y,0)] = 0$.
Proof: By induction on x and y, using T178. □

T180: $x \neq 0 \;.\!\supset\!.\; |x \boxplus y| = |x| + |y|$.
Proof: By induction on y, using T75 and T158. □

D181: (Multiplication):

$$x \cdot y \stackrel{d}{\equiv} \begin{cases} \text{If } y = 0 \text{ then } 0 \\ \quad \text{else} \\ Cond(t(\cdot) \dot{-} (x1\#y), t(\cdot), x1\#y), \end{cases}$$

$$t(\cdot) \equiv Cond(Parity(y), (x \cdot \lfloor \tfrac{1}{2}y \rfloor)0, (x \cdot \lfloor \tfrac{1}{2}y \rfloor)0 + x).$$

T182: $[Equ(x \cdot y, 0) \;.\!\Rightarrow\!.\; (Equ(x,0) \vee Equ(y,0))] = 0$.
Proof: By induction on y. The bounding inequality for D181 may be assumed in the induction steps since $(x1\#y) \neq 0$ by T171. Hence, assuming T178 for x and y and that $y \neq 0$:
(182.0): If $(x \cdot y0) = 0$ then $(x \cdot y)0 = 0$ so $x \cdot y = 0$, hence $x = 0$ by I.H.;
(182.1): If $(x \cdot y1) = 0$ then $(x \cdot y)0 + x = 0$, so $x = 0$ by T179. □

T183: $x \cdot y = Cond(Parity(y), (x \cdot \lfloor \frac{1}{2}y \rfloor)0, (x \cdot \lfloor \frac{1}{2}y \rfloor)0 + x)$.
Proof: We prove $t(\cdot) \dot{-} (x1\#y) = 0$ by induction on y:
(183.$\emptyset$): $(x \cdot 0)0 \dot{-} (x1\#0) = 0 \dot{-} 1 = 0$;
(183.0): Assuming $y \neq 0$:

$$\begin{array}{rlr} & (x \cdot \lfloor \frac{1}{2}y0 \rfloor)0 \dot{-} (x1\#y0) & \\ = & (x \cdot y)0 \dot{-} [(x1\#y) \boxplus x1] & \text{(Ax. 6a)} \\ = & (x \cdot y)0 \dot{-} [(x1\#y)0 \boxplus x] & \text{(T172)} \\ = & [(x \cdot y)0 \dot{-} (x1\#y)0] \dot{-} x & \text{(T163, T171)} \\ = & [(x \cdot y) \dot{-} (x1\#y)] \dot{-} x & \text{(T171, T20)} \\ = & 0 \qquad \text{(I.H.);} & \end{array}$$

(183.1): Assuming $y \neq 0$:

$$\begin{array}{rlr} & [(x \cdot \lfloor \frac{1}{2}y1 \rfloor)0 + x] \dot{-} (x1\#y1) & \\ = & [(x \cdot y)0 + x] \dot{-} [(x1\#y) \boxplus x1] & \text{(Ax. 6b)} \\ = & [(x \cdot y)0 + x] \dot{-} [(x1\#y)0 \boxplus x] & \text{(T172).} \end{array}$$

By T23, $(x \cdot y) \dot{-} (x1\#y) = 0$, so $(x \cdot y)0 \dot{-} (x1\#y)0 = 0$ by T171, T20. If $x = 0$, then $(x \cdot y)0 + x = (x \cdot y)0$ and $(x1\#y)0 \boxplus x = (x1\#y)0$, so $[(x \cdot y)0 + x] \dot{-} [(x1\#y)0 \boxplus x] = 0$. If $x \neq 0$ then $x \cdot y \neq 0$ by T182, so

$$\begin{array}{rclr} (x \cdot y)0 + x & \sqsubseteq & (x \cdot y)0 \boxplus x & \text{(T177)} \\ & \sqsubseteq & (x1\#y)0 \boxplus x & \text{(T180, T162, T175).} \quad \square \end{array}$$

T184: $x \cdot 1 = x$. $\square$

T185: $s_0(x) = x + x = x \cdot 2$. $\square$

T186: $x \cdot (y + 1) = (x \cdot y) + x$. $\square$

T187: $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$. $\square$

T188: $x0 \cdot y = (x \cdot y)0$. $\square$

T189: $x1 \cdot y = (x \cdot y)0 + y$. $\square$

T190: $0 \cdot x = 0$. $\square$

T191: $x \cdot y = y \cdot x$.
Proof: By induction on y, using T188, T189, T190. $\square$

T192: $[Lessequ(1,x) \mathrel{.\Rightarrow.} Equ(|2 \cdot x|, S(|x|))] = 0$.
Proof: For $x = 0$, $Lessequ(1,x) = Lessequ(1,0) = 1$; for $x \neq 0$,

$$\begin{array}{rcll} |2 \cdot x| & = & |x \cdot 2| & \text{(T191)} \\ & = & |s_0(x)| & \text{(T185)} \\ & = & S(|x|) & \text{(T135).} \quad \square \end{array}$$

T193: $|S(2 \cdot x)| = S(|x|)$.
Proof: By T185, T26, T135. $\square$

T194: $|x \# y| = S(|x| \cdot |y|)$.
Proof: By induction on y, using T75, T180, T158. $\square$

T195: $1 \# 1 = 2$. $\square$

T196: $0 \# x = 1$. $\square$

T197: $xi \neq 0 \mathrel{.\supset.} xi \# y = (x \# y) \boxplus y$.
Proof: By induction on y:
(197.∅): $xi \# 0 = 1 = (x \# 0) \boxplus 0$;
(197.j): $xi \# yj$
$= (xi \# y) \boxplus xi$
$= ((x \# y) \boxplus y) \boxplus xi$ (I.H.)
$= ((x \# y) \boxplus xi) \boxplus y$ (T73)
$= ((x \# y) \boxplus x) \boxplus yj$ (T152, T172)
$= (x \# yj) \boxplus yj$. $\square$

T198: $x \# y = y \# x$.
Proof: By induction on y, using T197. $\square$

T199: $(x \boxplus y) \cdot z = (x \cdot z) \boxplus y$.
Proof: By induction on y, using T188. $\square$

T200: $x \neq 0 \mathrel{.\supset.} (x \boxplus y) \# z = (x \# z) \cdot (y \# z)$.
Proof: By induction on y:
(200.∅): $(x \boxplus 0) \# z$
$= (x \# z)$
$= (x \# z) \cdot (0 \# z)$ (T196);
(200.i): $(x \boxplus yi) \# z$
$= (x \boxplus y) 0 \# z$
$= [(x \boxplus y) \# z] \boxplus z$ (T197)
$= [(x \# z) \cdot (y \# z)] \boxplus z$ (I.H.)

$= [(y\#z)\cdot(x\#z)] \boxplus z$ (T191)
$= [(y\#z)\boxplus z]\cdot(x\#z)$ (T199)
$= (x\#z)\cdot(yi\#z)$. $\square$

T201: $[Equ(|x|,|y|) \Rightarrow Equ(x\#z, y\#z)] = 0$.
Proof: Induction on x, y:
(201.ij): If $|xi| = |yj|$ then $|x| = |y|$, hence

$$\begin{aligned} xi\#z &= (x\#z)\boxplus z && (\text{T197, } xi \neq 0) \\ &= (y\#z)\boxplus z && (\text{I.H.}) \\ &= yj\#z && (\text{T197, } yj \neq 0). \quad \square \end{aligned}$$

T202: $[Equ(|x|, |u|+|v|) \Rightarrow Equ(x\#y, (u\#y)\cdot(v\#y))] = 0$.
Proof: Assuming $|x| = |u|+|v|$, $u \neq 0$, then $|x| = |u \boxplus v|$ by T180, so $x\#y = (u\boxplus v)\#y = (u\#v)\cdot(v\#y)$ by T201, T200, T196, T184, T191. $\square$

T203: $[Equ(x, \lfloor\frac{1}{2}x\rfloor + \lfloor\frac{1}{2}x\rfloor) \vee Equ(x, S(\lfloor\frac{1}{2}x\rfloor + \lfloor\frac{1}{2}x\rfloor))] = 0$. $\square$

DR204: Let $f(x,\vec{y})$ be an $n+1$-place function symbol of PV. Then there are $n+1$-place function symbols $f^{\exists}$ and $f^{\forall}$ of PV so that the following are theorems of PV:

(1) $f^{\exists}(0,\vec{y}) = sg(f(0,\vec{y}))$
(2) $f^{\exists}(a,\vec{y}) = f^{\exists}(\lfloor\frac{1}{2}a\rfloor,\vec{y}) \vee f(|a|,\vec{y})$
(3) $f^{\forall}(0,\vec{y}) = sg(f(0,\vec{y}))$
(4) $f^{\forall}(a,\vec{y}) = f^{\forall}(\lfloor\frac{1}{2}a\rfloor,\vec{y}) \& f(|a|,\vec{y})$.

Proof: The function $f^{\exists}$ can be defined by:

$$f^{\exists}(a,\vec{y}) \stackrel{d}{\equiv} \begin{cases} \text{If } a = 0 \text{ then } sg(f(0,\vec{y})) \\ \quad \text{else} \\ \text{if } t(f^{\exists}) \mathbin{\dot{-}} 1 = 0 \text{ then } t(f^{\exists}) \text{ else } 1, \end{cases}$$

$$t(f^{\exists}) \equiv f^{\exists}(\lfloor\tfrac{1}{2}a\rfloor,\vec{y}) \vee f(|a|,\vec{y})$$

The bounding inequality is immediate, by T45. The function $f^{\forall}$ is defined similarly. $\square$

Finally, we prove that any function of PV is provably dominated by a monotone function.

T205: Let f be an n-place function symbol of PV. Then there is an n-place function symbol f^M of PV for which the theorems are provable:

(1) $Lessequ(f(x_1, \ldots, x_n), f^M(x_1, \ldots, x_n)) = 0$

(2) $[(Lessequ(x_1, y_1) \& \cdots \& Lessequ(x_n, y_n)) \Rightarrow$
$Lessequ(f^M(x_1, \ldots, x_n), f^M(y_1, \ldots, y_n))] = 0.$

Proof: By induction on the complexity of the definition of the function symbol f (Def. 3.1). It is straightforward to show that any function is provably dominated by a term built from $0, s_0, s_1, Pad$ and $\#$. Since all of these functions are monotone, so is the function f^M defined by the term. $\Box$

Acknowledgement

Our thanks to Sam Buss and Victor Harnik for carefully reading and commenting upon the earlier version of this paper. Our thanks to Vicky Shum for typesetting the many versions of the manuscript with great skill and speed.

References

[1] Errett Bishop. Mathematics as a numerical language. In J. Myhill A. Kino and R.E. Vesley, editors, *Intuitionism and Proof Theory: Proceedings of the Summer Conference at Buffalo, New York, 1968*, pages 53–71. North-Holland, Amsterdam, 1970.

[2] Samuel R. Buss. *Bounded Arithmetic*. Ph.D. dissertation, Princeton University 1985; reprinted Bibliopolis, Naples, 1986. Page references are to the reprinting.

[3] Samuel R. Buss. A note on bootstrapping intuitionistic bounded arithmetic. Unpublished manuscript.

[4] Samuel R. Buss. The polynomial hierarchy and intuitionistic bounded arithmetic. *Structure in Complexity Theory*, Springer-Verlag Lecture Notes in Computer Science No. 223, pages 77–103, 1986.

[5] Samuel R. Buss. On model theory for intuitionistic bounded arithmetic with applications to independence results. In S.R. Buss and P.J. Scott, editors, *Feasible Mathematics*, pages 27–47. Birkhauser, 1990.

[6] Alonzo Church. *Introduction to Mathematical Logic*. Princeton U. P., Princeton, N.J., 1956.

[7] Peter Clote and Gaisi Takeuti. Exponential time and bounded arithmetic. *Structure in Complexity Theory*, Springer-Verlag Lecture Notes in Computer Science No. 223, pages 125–143, 1986.

[8] Alan Cobham. The intrinsic computational difficulty of functions. In Y. Bar-Hillel, editor, *Proc. Int. Conf. Logic, Methodology, and Philosophy Sci.*, pages 24–30. North Holland, Amsterdam, 1965.

[9] S. Cook and A. Urquhart. *Functional Interpretations of Feasibly Constructive Arithmetic*. Technical Report 210/88, University of Toronto, 1988. Extended Abstract in *Proceedings 21st ACM Symposium on Theory of Computing*, May 1989, pp. 107–112.

[10] Stephen Cook and Toniann Pitassi. A feasibly constructive lower bound for resolution proofs. *Information Processing Letters*, 34:81–85, 1990.

[11] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus. In *Proc. 7th A.C.M. Symposium on the Theory of Computation*, pages 83–97, 1975.

[12] Stephen A. Cook. Computability and complexity of higher type functions. In *Proc. MSRI Workshop on Logic from Computer Science*. Springer-Verlag, (to appear).

[13] Stephen A. Cook. Computability complexity of higher type functions. In *Proc. International Congress of Mathematicians 1990.* Springer-Verlag, (to appear).

[14] Stephen A. Cook et al. Course notes for CSC 2429S, lectures 15–25, 1986. Spring.

[15] Stephen A. Cook and Bruce M. Kapron. Characterizations of the basic feasible functionals of finite type. In S.R. Buss and P.J. Scott, editors, *Feasible Mathematics*, pages 71–95. Birkhauser, 1990.

[16] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symbolic Logic*, 44(1):36–50, March 1979.

[17] Martin Dowd. *Primitive recursive arithmetic with recursion on notation and boundedness.* M.Sc. thesis, University of Toronto, 1976. Department of Computer Science Technical Report #88.

[18] Martin Dowd. *Propositional representation of arithmetic proofs.* Ph.D. dissertation, University of Toronto, 1979. Department of Computer Science Technical Report #132/79.

[19] Martin Dowd. Model theoretic aspects of P $\neq$ NP. Unpublished manuscript, 1985.

[20] Kurt Gödel. Über eine bisher noch nicht benützte erweiterung des finiten standpunktes. *Dialectica* 12, 280-287. English translation: *J. of Philosophical Logic* 9 (1980), 133–142. A revised and expanded English translation is in *Gödel's Works*, Volume II, pages 271–280. Oxford U.P. 1990.

[21] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[22] Victor Harnik. Provably total functions of intuitionistic bounded arithmetic. *J. Symbolic Logic*, (to appear).

[23] J. Roger Hindley and Jonathan P. Seldin. *Introduction to Combinators and λ-calculus.* Cambridge University Press, 1986.

[24] Stephen C. Kleene. *Introduction to Metamathematics.* Van Nostrand, Princeton, N.J., 1952.

[25] Jan Krajiček and Pavel Pudlak. Propositional proof systems, the consistency of first order theories and the complexity of computations. *J. Symbolic Logic*, 54(3):1063–1079, Sept. 1989.

[26] J. Lambek and P.J. Scott. *Introduction to Higher Order Categorical Logic.* Cambridge University Press, 1986.

[27] Daniel Leivant. Syntactic translations and provably recursive functions. *J. Symbolic Logic*, 50:51–68, 1985.

[28] Edward Nelson. *Predicative Arithmetic.* Princeton U. P., Princeton, N.J., 1986.

[29] Toniann Pitassi. Feasibly constructive proof of Fermat's Last Theorem for $n = 3$. Manuscript, 1988.

[30] D. Prawitz. *Natural Deduction, A Proof-Theoretic Study.* Alimquist & Wiksell, 1965.

[31] Robert A. Reckhow. *On the lengths of proofs in the propositional calculus.* Ph.D. dissertation, University of Toronto, 1975.

[32] V.Yu. Sazonov. An equivalence between polynomial constructivity of Markov's principle and the equality $P = NP$. In P.P. Petkov, editor, *Mathematical Logic*, pages 351–360. Plenum Press, New York, 1990.

[33] Raymond M. Smullyan. *Theory of Formal Systems.* Annals of Mathematics Studies, no. 47, Princeton U. P., Princeton, N.J., 1961.

[34] Richard Statman. *The typed λ-calculus is not elementary recursive. Theoretical Computer Science*, 9:73–81, 1980.

[35] W.W. Tait. *Intensional interpretations of functionals of finite type I. J. Symbolic Logic*, 32:198–212, 1967.

[36] Gaisi Takeuti. *Proof Theory.* North-Holland, second edition, 1987.

[37] A.S. Troelstra. Metamathematical investigation of intuitionistic arithmetic and analysis. Springer-Verlag Lecture Notes in Mathematics, No. 344, 1973.