Herbrand Analysis of Some Second-order Theories with Weak Set Existence Principles

Chung Tong Lee[1]
Aleksandar Ignjatović[2]

[1] University of New South Wales, NSW 2052, Australia
ctlee@cse.unsw.edu.au
[2] University of New South Wales and NICTA, NSW 2052, Australia
ignjat@cse.unsw.edu.au

Technical Report
UNSW-CSE-TR-0808
April 2008

THE UNIVERSITY OF
NEW SOUTH WALES

School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia

Abstract

We present a proof-theoretic analysis of some second-order theories of binary strings which were introduced in [5]. The core of these theories contains, besides finitely many open axioms for basic operations on strings, only a weak comprehension axiom schema. In such theories, a collection **W** can be defined to play the role of natural numbers. **W** is given as the intersection of all sets containing the empty string and closed under the two successor functions S^0 and S^1. We characterize the classes of functions which provably map **W** into itself and whose graphs are defined by formulas of an appropriate bounded quantifier complexity. For theories with weak comprehension schemas, this notion corresponds naturally to that of provably recursive functions for arithmetic theories. The techniques of Herbrand analysis developed by Sieg in [8] and [9] allow us to prove that these classes match up with levels of the polynomial-time hierarchy.

1 Languages and Theories

Second-order theories with weak set existence principles, suitable for a treatment of computational complexity classes, were introduced by Leivant in [6]. He defined a theory with a comprehension schema for quantifier-free *positive* formulas, denoted by $L_2(QF^+)$, and a collection of strings which plays the role of natural numbers via the usual impredicative definition. Leivant showed that $L_2(QF^+)$ proves that a function f maps the collection of "natural numbers" into itself, just in case f is a polynomial-time computable function. In [5] a more standard approach was taken. Algorithms are treated as definable partial functions in theories with a comprehension principle for formulas with limited quantifier complexity, but with no restriction on the propositional connectives. We review briefly the theories $\mathbf{C}^b(\Sigma_i^b)$ from [5]. The languages we consider are formulated in Ferreira's [3].

The language of $\mathbf{C}^b(\Sigma_i^b)$ is denoted by $\mathbf{L}^b$. Its first-order variables x, y, z, ... range over binary strings, while its second-order variables X, Y, Z, ... range over sets of binary strings. The non-logical vocabulary of $\mathbf{L}^b$ consists of the symbols ε, S^0, S^1, $\oplus$, $\otimes$, $\upharpoonright$, $=$, $\sqsubseteq$, $\preccurlyeq$ and $\in$. The meaning is indicated in the following table:

Symbols	Meanings
ε	constant symbol for the empty string
$S^0(x), S^1(x)$	concatenate 0, 1 to the end of x
$x \oplus y$	concatenate y to the end of x
$x \otimes y$	concatenate x to itself length of y many times
$x \upharpoonright y$	the initial segment of x with length equal to that of y, or just x if the length of x is smaller than that of y
$x = y$	x equals y
$x \sqsubseteq y$	x is an initial segment of y
$x \preccurlyeq y$	the length of x is smaller than or equal to that of y
$x \in X$	x is a member of X

In analogy to the language for bounded arithmetic in [2], we introduce bounded and sharply bounded quantifiers $(\forall x \preccurlyeq t)$, $(\exists x \preccurlyeq t)$, respectively $(\forall x \sqsubseteq t)$, $(\exists x \sqsubseteq t)$. The hierarchy of bounded formulas, Σ_i^b, is defined as usual. The class of formulas $\boldsymbol{\Sigma}_\mathbf{0}^\mathbf{b}$ is defined similarly to Σ_0^b, except that formulas in $\boldsymbol{\Sigma}_\mathbf{0}^\mathbf{b}$ may contain second-order variables. The class of all open formulas is denoted by **QF**. All languages in this paper extend $\mathbf{L}^b$ only by symbols for functions or functionals whose values are strings, not sets of strings. When referring to a formula class of a particular language, we add the language symbol in parentheses after the class symbol, e.g. $\Sigma_1^b(\mathbf{L}^b)$.

The theory *BASIC* contains only axioms expressing elementary properties of the functions and relations of $\mathbf{L}^b$; its second-order extensions $\mathbf{C}^b$ and

$\mathbf{C}^b(\Sigma_i^b)$ in $\mathbf{L}^b$ are defined as in [5] with comprehension schema Φ–CA given in the form:

$$(\forall\vec{y})(\forall\vec{Y})(\exists X)(\forall x)\left((x \in X) \leftrightarrow \varphi(x, \vec{y}, \vec{Y})\right),$$

where Φ is a class of formulas of the underlying language and φ is a formula of that class not containing the variable X.

Definition 1.

1. *$\mathbf{C}^b \overset{df}{=} BASIC + \mathbf{\Sigma_0^b}(\mathbf{L}^b)$–$CA$*

2. *For all $i \geq 1$, $\mathbf{C}^b(\Sigma_i^b) \overset{df}{=} \mathbf{C}^b + \Sigma_i^b(\mathbf{L}^b)$–$CA$.*

2 Inductive Sets and Definable Functions

Let $\mathcal{M}$ be a model of a theory $\mathbf{T} \supseteq \mathbf{C}^b$. An element in the second-order domain of $\mathcal{M}$ is *inductive* if it contains the empty string and is closed under the two successor functions. The intersection $\mathbf{W}$ of all inductive elements in $\mathcal{M}$ is defined by the formula $W(x)$ which is given by the following formulas:

$$Ind(X) \overset{df}{=} (\varepsilon \in X) \wedge (\forall y)\left((y \in X) \rightarrow \left((S^0(y) \in X) \wedge (S^1(y) \in X)\right)\right);$$
$$W(x) \overset{df}{=} (\forall X)\left(Ind(X) \rightarrow (x \in X)\right).$$

$\mathbf{W}$ has many properties analogous to those of the set $\mathbb{N}$ of natural numbers as defined in, say, ZF set theory. For example, the following two facts are proved in [5] and are significant in the proof of the Corollary 5 below:

Lemma 1. $\mathbf{C}^b \vdash (\forall x)(\forall y \preccurlyeq x)(W(x) \rightarrow W(y))$.

Lemma 2. $\mathbf{W}$ *is closed for all functions with polynomial growth rate.*

For an arbitrary formula φ, φ^W denotes the formula obtained from φ by relativizing all first order quantifiers of φ to $\mathbf{W}$. Thus, if φ is of the form $(\forall x)\psi$ or $(\exists x)\psi$, then φ^W is $(\forall x)(W(x) \rightarrow \psi^W)$, respectively $(\exists x)(W(x) \wedge \psi^W)$. $\mathcal{B}^1$ is the standard structure consisting of the set of all finite binary strings $B = \{0,1\}^*$ together with its usual operations and relations. In [5] an appropriate notion of "Φ–definable function" was introduced:

Definition 2. *A function $f(\vec{x}) : B^k \mapsto B$ is Φ-definable in the theory $\mathbf{T}$ which extends $\mathbf{C}^b$, if there is a formula $\varphi_f(\vec{x}, y) \in \Phi$ such that:*

$$\mathbf{T} \vdash \left((\forall\vec{x})(\exists! y)\varphi_f(\vec{x}, y)\right)^W,$$
$$\mathcal{B}^1 \models (\forall\vec{x})\varphi_f(\vec{x}, f(\vec{x})).$$

We take over definitions of some additional notions from bounded arithmetic, but replace reference to numbers by that to binary strings. In particular, $\square^p_{i+1}$ is the collection of all functions computable in polynomial time with a Σ^b_i-oracle. Buss proved in [2] that the functions in $\square^p_i$ are those which are Σ^b_i-definable in S^i_2 for the number version, and Ferreira established the binary string version of this theorem in [4]. Buchholz and Sieg in [1] reached the latter result via a different route, namely, Herbrand analysis.

Through an interpretability result, with comprehension in place of induction, it was shown in [5] that:

Lemma 3. *All functions from* $\square^p_i$ *are* Σ^b_i*-definable in the theory* $\mathbf{C}^b(\Sigma^b_i)$.

The converse of Lemma 3 was proved in [5] by a model-theoretic argument and Buss' corresponding theorem for S^i_2. Thus, $\mathbf{C}^b(\Sigma^b_1)$ plays the role of the theory S^1_2 in bounded arithmetic: its Σ^b_1-definable functions are exactly the polynomial-time computable functions. Here we prove the converse of Lemma 3 directly by an Herbrand analysis of the theories $\mathbf{C}^b(\Sigma^b_i)$. We follow the method presented in [9], in particular section 1.3.

3 Proof System

We work with a Tait-style calculus that is slightly different from the calculus used in [9], since our second-order objects are sets, rather than functions. Quantifiers are the standard ones: $\forall$ and $\exists$. Connectives for formulas are $\wedge$, $\vee$ and $\neg$ with their usual meanings. However, negations are allowed only in front of *atomic* formulas. Thus, $\neg\varphi$, where φ is an arbitrary formula, stands for the equivalent formula with negations pushed in front of the atomic subformulas. Similarly, $\rightarrow$ and $\leftrightarrow$ are abbreviations for the corresponding equivalent formulas built up with $\wedge$ and $\vee$ from (negated) atomic formulas.

In this calculus we derive finite sets of formulas. Such sets are denoted by capital Greek letters; Γ, Δ stands for $\Gamma \cup \Delta$ and Γ, φ stands for $\Gamma \cup \{\varphi\}$. The free variables of the *end set* of a derivation are called the *parameters* of that derivation. The *initial sets* of derivations are either logical axioms of the form $\Gamma, \varphi, \neg\varphi$, where φ is an atomic formula, or axioms of the form Γ, Φ^*, where Φ^* is a substitution instance of Φ which represents either an equality axiom or a non-logical axiom of a theory $\mathbf{T}$ extending $\mathbf{C}^b$. The inference rules of the calculus are as follows:

$$(\wedge) \quad \frac{\Gamma, \varphi \qquad \Gamma, \psi}{\Gamma, \varphi \wedge \psi}$$

$$(\vee) \quad \frac{\Gamma, \varphi}{\Gamma, \varphi \vee \psi}\,, \qquad \frac{\Gamma, \psi}{\Gamma, \varphi \vee \psi}$$

$$(\forall^1) \quad \frac{\Gamma, \varphi(a)}{\Gamma, (\forall x)\varphi(x)} \qquad (a \text{ is not free in any formula in } \Gamma)$$

$$(\exists^1) \quad \frac{\Gamma, \varphi(t)}{\Gamma, (\exists x)\varphi(x)}$$

$$(\forall^2) \quad \frac{\Gamma, \varphi(A)}{\Gamma, (\forall X)\varphi(X)} \qquad (A \text{ is not free in any formula in } \Gamma)$$

$$(\exists^2) \quad \frac{\Gamma, \varphi(A)}{\Gamma, (\exists X)\varphi(X)}$$

$$(C) \quad \frac{\Gamma, \varphi \qquad \Gamma, \neg\varphi}{\Gamma} \qquad (\varphi \text{ is called the cut formula})$$

The comprehension principle is incorporated in the calculus via a comprehension rule:

$$(\Phi\text{–}CR) \quad \frac{\Gamma, \psi(V_\varphi)}{\Gamma, (\exists X)\psi(X)} \quad (X \text{ does not occur in } \psi(V_\varphi) \text{ and } \varphi \in \Phi.)$$

V_φ is called the *abstract* of φ and is an auxiliary technical notion (not part of the formal proof-system): $\psi(V_\varphi)$ is the formula obtained from $\psi(X)$ by replacing all occurrences of atomic formulas $t \in X$ in $\psi(X)$ by $\varphi(t)$ where t is a term of the language (see [2] or [10]). The formula $\psi(V_\varphi)$ is the *minor formula* of Φ–CR, while the formula $(\exists X)\psi(X)$ is the *principal formula* of this rule.

It is well known (see e.g. [10]) that the proof system with Φ–CA is equivalent to that with Φ–CR, when Φ is closed under substitution (as just described) of second-order variables by the abstracts of formulas from Φ. All theories in our paper have this property. We call a derivation **D** *normal* if and only if

- every cut formula in **D** is a member of a set of formulas which represents an instance of an axiom;
- every free variable that appears in **D**, but is not among the parameters of **D**, is used as the eigenvariable of exactly one $\forall$–inference and occurs in **D** only above its corresponding rule;
- the eigenvariables $\vec{b} = b_0, \ldots, b_m$ are enumerated in such a way that if b_i is quantified below b_j (on the same thread), then $i < j$.

For every derivation of Γ in the proof-systems with Φ–CR there exists a normal derivation of Γ in the same system, i.e., the cut elimination theorem holds for these systems.

4 Functionals and Skolemization

By adding suitable functions to a language, we can perform term extraction from a derivation of $(\exists y)\varphi(\vec{x}, y)$ which expresses a functional dependence of

y on $\vec{x}$. Term extraction is the central idea of Herbrand analysis, it requires a delicate balance between mathematical proof principles (with existential import like induction for existentially quantified formulas) and definition principles for functions (like primitive recursion). The latter are used to "functionally" analyze the formulas. In our case, we are dealing with a weak set-existence principle and consider the limited recursion principle to define polynomial-time computable functions. In the classical investigations of Σ_1^0–IND, the primitive recursively defined functions allow the perfectly balanced analysis; in this case, we are considering the polynomial-time computable functions play the analogous role.

Definition 3. *A function $f(\vec{x})$ is defined by* limited recursion *from $g(\vec{x})$, $h_0(\vec{x}, y)$, $h_1(\vec{x}, y)$ and $k(\vec{x}, y)$ if:*

$$\begin{aligned} f(\vec{x}, \varepsilon) &= g(\vec{x}) \upharpoonright k(\vec{x}, \varepsilon); \\ f(\vec{x}, S^0(y)) &= h_0(\vec{x}, y, f(\vec{x}, y)) \upharpoonright k(\vec{x}, S^0(y)); \\ f(\vec{x}, S^1(y)) &= h_1(\vec{x}, y, f(\vec{x}, y)) \upharpoonright k(\vec{x}, S^1(y)). \end{aligned}$$

This definition of *limited recursion* is a slight modification of that given in [5]; it guarantees that $(\forall \vec{x})\Big((\forall y)\big(f(\vec{x}, y) \preccurlyeq k(\vec{x}, y)\big)\Big)^W$ in our theories. The first expansion of $\mathbf{L}^b$ is obtained by adding symbols for functions defined by composition and limited recursion from functions of $\mathbf{L}^b$; it is naturally called $\mathbf{L}^p$. The suitable theory for $\mathbf{L}^p$ was introduced in [5]: $BASIC^p$ contains $BASIC$ and the defining equations for all functions of $\mathbf{L}^p$.

In our further investigations, we call a purely first-order mapping a *function*, and a mapping with second-order arguments (but first-order values only) a *functional*. For example, $\mathbf{Mem}(X, x)$ is the functional defined by

$$\mathbf{Mem}(X, x) = \begin{cases} \varepsilon & \text{if } x \in X \\ S^0(\varepsilon) & \text{otherwise.} \end{cases}$$

The notions of composition, limited recursion and definable function can be extended to functionals. The language $\mathbf{L^p}$ is obtained from $\mathbf{L}^p$ by adding symbols for all functionals defined by composition and limited recursion from $\mathbf{Mem}(X, x)$ and functions in $\mathbf{L}^p$. The corresponding theory $BASIC^{\mathbf{p}}$ contains $BASIC^p$, the axioms for $\mathbf{Mem}(X, x)$, i.e., $\{x \notin X, \mathbf{Mem}(X, x) = \varepsilon\}$ and $\{x \in X, \mathbf{Mem}(X, x) = S^0(\varepsilon)\}$, and the defining equations for all functionals of $\mathbf{L^p}$ as just described. Using comprehension, the theory $\mathbf{C^p}$ is defined as $BASIC^{\mathbf{p}} + \mathbf{\Sigma_0^b}(\mathbf{L^p})$–$CR$.

A functional $F(\vec{X}, \vec{x})$ is *polynomial-time computable* when its values can be computed by a polynomial-time Turing machine which uses oracles to decide membership in the sets $\vec{X}$. The argument for Theorem 2 in the first chapter of [2] yields:

Theorem 4. *The class of polynomial-time computable functionals is exactly the class of functionals of* $\mathbf{L^p}$*.*

For clarity's sake, we denote by $\vec{\mathfrak{X}}$ the sequence of variables $\vec{X}$ and $\vec{x}$. The next fact is a consequence of Lemma 1 and Lemma 2.

Corollary 5. *Let* $\mathbf{T}$ *be a theory in the language* $\mathbf{L} \supseteq \mathbf{L}^b$ *such that* $\mathbf{T}$ *extends* $\mathbf{C}^b$*. For every functional* F *of* $\mathbf{L}$*, if there is a term* t_F *of* $\mathbf{L}^b$ *such that* $\mathbf{T} \vdash \Big((\forall \vec{\mathfrak{X}})(F(\vec{\mathfrak{X}}) \preccurlyeq t_F(\vec{x}))\Big)^W$*, then* $\mathbf{T} \vdash \neg W(\vec{x}), W(F(\vec{\mathfrak{X}}))$*.*

The polynomial induction principle, Φ–$PIND$, is given by the schema:

$$\Big(\varphi(\varepsilon, \vec{\mathfrak{X}}) \wedge (\forall y)\Big(\varphi(y, \vec{\mathfrak{X}}) \rightarrow \Big(\varphi(S^0(y), \vec{\mathfrak{X}}) \wedge \varphi(S^1(y), \vec{\mathfrak{X}})\Big)\Big)\Big) \rightarrow (\forall y)\varphi(y, \vec{\mathfrak{X}})$$

where $\varphi(y, \vec{\mathfrak{X}})$ is any formula in Φ. In the theory $\mathbf{C^p}$, we not only have the appropriately limited recursion, but also the $\mathbf{W}$-relativized Φ–$PIND$ schema for the formula class $\mathbf{\Sigma_0^b}(\mathbf{L^p})$. The proof for this schema in $\mathbf{C^p}$ is essentially the same as that for $(\mathbf{\Sigma_0^b}(\mathbf{L}^b) - PIND)^W$ in $\mathbf{C}^b$; c.f. [5].

Lemma 6. $\mathbf{C^p} \vdash (\mathbf{\Sigma_0^b}(\mathbf{L^p})$–$PIND)^W$*.*

With this principle and by induction on the complexity of formulas, we can establish the following: for any open formula φ of $\mathbf{L^p}$, there is a functional F_φ of the same language such that

$$\mathbf{C^p} \vdash \neg W(\vec{x}), \varphi(\vec{\mathfrak{X}}) \leftrightarrow \Big(F_\varphi(\vec{\mathfrak{X}}) = \varepsilon\Big).$$

This is an important fact for our applications of Herbrand analysis, because it allows us to define functionals by cases with open formulas as conditions. It follows that for every open formula φ of the language $\mathbf{L^p}$ and every sequence of functionals F_i, $i \leq k$, there exists a single functional F such that

$$\mathbf{C^p} \vdash \neg W(\vec{x}), \bigvee_{i \leq k} \varphi(\vec{\mathfrak{X}}, F_i(\vec{\mathfrak{X}})) \leftrightarrow \varphi(\vec{\mathfrak{X}}, F(\vec{\mathfrak{X}})). \tag{4.1}$$

This fact will be used at the very end of the proof of Lemma 10.

Theorem 7. *For any open formula* φ *and any term* t *of* $\mathbf{L^p}$*, there is a functional* F_φ *of the same language such that*

$$\mathbf{C^p} \vdash \neg W(\vec{x}), (\exists y \sqsubseteq t(\vec{\mathfrak{X}}))\varphi(\vec{\mathfrak{X}}, y) \rightarrow \Big(\varphi(\vec{\mathfrak{X}}, F_\varphi(\vec{\mathfrak{X}})) \wedge (\forall v \sqsubset F_\varphi(\vec{\mathfrak{X}}))\neg\varphi(\vec{\mathfrak{X}}, v)\Big)$$

where $x \sqsubset y$ *stands for* $(x \sqsubseteq y) \wedge \neg(x = y)$*.*

Proof. Let $\psi(\vec{\mathfrak{x}}, u)$ be the formula

$$(\exists y \sqsubseteq u)\varphi(\vec{\mathfrak{x}}, y) \to \Big(\varphi(\vec{\mathfrak{x}}, F(\vec{\mathfrak{x}}, u)) \wedge (\forall v \sqsubset F(\vec{\mathfrak{x}}, u))\neg\varphi(\vec{\mathfrak{x}}, v)\Big)$$

where $F(\vec{\mathfrak{x}}, u)$ is a search functional defined by composition and limited recursion: starting from ε, F looks for the shortest witness $y_0 \sqsubseteq u$ for $\varphi(\vec{\mathfrak{x}}, y_0)$ and gives y_0. If there is no such witness, $F(\vec{\mathfrak{x}}, u)$ equals $S^0(u)$. As $\mathbf{C^p}$ proves $\psi(\vec{\mathfrak{x}}, \varepsilon)$ and $\psi(\vec{\mathfrak{x}}, u) \to \Big(\psi(\vec{\mathfrak{x}}, S^0(u)) \wedge \psi(\vec{\mathfrak{x}}, S^1(u))\Big)$, Lemma 6 yields a derivation of $\neg W(u), \psi(\vec{\mathfrak{x}}, u)$. Now, define $F_\varphi(\vec{\mathfrak{x}}) = F(\vec{\mathfrak{x}}, t(\vec{\mathfrak{x}}))$, substitute u by $t(\vec{\mathfrak{x}})$ and apply cut with a derivation $\neg W(\vec{x}), W(t(\vec{\mathfrak{x}}))$ from Corollary 5 to complete the argument. □

This theorem has as an immediate consequence that every $\mathbf{\Sigma_0^b}$-formula is equivalent to an open formula when relativized to $\mathbf{W}$, provably in $\mathbf{C^p}$.

Corollary 8. *For every $\mathbf{\Sigma_0^b}$-formula φ of $\mathbf{L^p}$, there exists an open formula φ^* of the same language such that*

$$\mathbf{C^p} \vdash \neg W(\vec{x}), \varphi(\vec{\mathfrak{x}}) \leftrightarrow \varphi^*(\vec{\mathfrak{x}}). \tag{4.2}$$

However, for the Herbrand analysis of $\mathbf{C}^b(\Sigma_i^b)$, we need a theory with a comprehension rule for open formulas φ with no restriction that its first-order variables must range over $\mathbf{W}$. The Skolem theory we introduce in the next definition addresses this problem. It is a version of the Skolem operator theory presented in section 2 of [8].

Definition 4.

1. *$\mathbf{L}_0^{Sk} \overset{df}{=} \mathbf{L^p}$, $BASIC_0^{Sk} \overset{df}{=} \varnothing$.*
2. *For every open formula φ and every term t of $\mathbf{L}_i^{Sk}$, there exists a functional symbol $S_{\varphi,t}$ in $\mathbf{L}_{i+1}^{Sk}$ such that*

   $$(\forall\vec{\mathfrak{x}})\big((\exists y \sqsubseteq t(\vec{\mathfrak{x}}))\varphi(\vec{\mathfrak{x}}, y) \to \varphi(\vec{\mathfrak{x}}, S_{\varphi,t}(\vec{\mathfrak{x}}))\big).$$

   Using the transformation for universal axioms in [9], the following sets are added to the theory $BASIC_i^{Sk}$ to obtain $BASIC_{i+1}^{Sk}$:

   $$\begin{gathered}\neg(y \sqsubseteq t(\vec{\mathfrak{x}})), \neg\varphi(\vec{\mathfrak{x}}, y), \varphi(\vec{\mathfrak{x}}, S_{\varphi,t}(\vec{\mathfrak{x}})) \\ S_{\varphi,t}(\vec{\mathfrak{x}}) \sqsubseteq t(\vec{\mathfrak{x}}).\end{gathered} \tag{4.3}$$

3. *$\mathbf{L}^{Sk} = \bigcup\limits_{i\in\omega} \mathbf{L}_i^{Sk}$, $BASIC^{Sk} = \bigcup\limits_{i\in\omega} BASIC_i^{Sk}$.*

Paralleling the developments in [8], we can show that for every $\mathbf{\Sigma_0^b}$-formula φ of $\mathbf{L^P}$, there exists an open formula φ^s of $\mathbf{L}^{Sk}$ such that

$$BASIC^{Sk} \vdash \varphi(\vec{\mathfrak{X}}) \leftrightarrow \varphi^s(\vec{\mathfrak{X}}). \tag{4.4}$$

The "converse" of this claim also holds, namely, for the formula φ^s in the defining equation (4.3) of an arbitrary $\mathbf{\Sigma_0^b}$-Skolem function $S_{\varphi^s,t}$, there is a $\mathbf{\Sigma_0^b}$-formula φ such that (4.4) holds. (This can be shown by induction on i of $\mathbf{L}_i^{Sk}$.) Combining the "converse" with (4.2) and Theorem 7, we have

$$BASIC^{Sk} \vdash (\exists y \sqsubseteq t(\vec{\mathfrak{X}}))\varphi^s(\mathfrak{X}, y) \to \varphi^s(\mathfrak{X}, S_{\varphi^s,t}(\mathfrak{X}))$$
$$\mathbf{C^P} \vdash \neg W(\vec{x}), (\exists y \sqsubseteq t(\vec{\mathfrak{X}}))\varphi^*(\mathfrak{X}, y) \to \varphi^*(\mathfrak{X}, F_{\varphi^*}(\mathfrak{X})).$$

This yields a natural correspondence between $S_{\varphi^s,t}$ of $\mathbf{L}^{Sk}$ and F_{φ^*} of $\mathbf{L^P}$. For each $\mathbf{\Sigma_0^b}$-Skolem function S, we denote the corresponding functional by F_S. We combine these Skolemizations by letting $BASIC^{\mathbf{QF}}$ contain $BASIC^{\mathbf{P}}$ and $BASIC^{Sk}$, as well as the axiom $\{\neg W(\vec{x}),\ S(\vec{\mathfrak{X}}) = F_S(\vec{\mathfrak{X}})\}$ for each $\mathbf{\Sigma_0^b}$-Skolem function S.

Definition 5. $\mathbf{C^{QF}} \overset{df}{=} BASIC^{\mathbf{QF}} + \mathbf{QF}(\mathbf{L}^{Sk})$–$CR$.

Only open formulas are eligible for CR in $\mathbf{C^{QF}}$, and yet $\mathbf{C^{QF}}$ is powerful enough to derive all theorems of $\mathbf{C}^b$. The reason is this: we can replace any abstract (in $\mathbf{L}^b$) of a $\mathbf{\Sigma_0^b}$–CR in $\mathbf{C}^b$ derivations with an open formula (in $\mathbf{L}^{Sk}$) and perform a $\mathbf{QF}$–CR in a $\mathbf{C^{QF}}$ derivation to get the same result. The general facts about Skolem theories and equation (4.4) imply:

Corollary 9. *The theory* $\mathbf{C^{QF}}$ *is a conservative extension of the theory* $\mathbf{C}^b$.

By applying the universal axioms transformation in section 1.1 of [9] we can assume: open formulas in sets that represent axioms do not contain $\vee$ and $\wedge$ symbols. Further, all quantified formulas in the axiom sets are of the form $\neg W(x)$. Following the proof of Lemma (2.5) in [7], it can be shown that $\forall$, $\vee$ and $\wedge$–inversion lemmas hold for our systems. We will use this fact and the properties of normal derivations in our proof-theoretic arguments.

5 Derivations in $\mathbf{C^{QF}}(s\text{-}\Sigma_1^b)$ and $\mathbf{C^{QF}}(\Sigma_1^b)$

We now formulate and prove one of our main proof-theoretic lemmas. It is concerned with the class s-Σ_1^b of strict Σ_1^b-formulas of $\mathbf{L}^{Sk}$. This class consists of all purely first-order formulas that start with exactly one bounded existential quantifier $(\exists x \preccurlyeq t)$ followed by an open formula. The theory $\mathbf{C^{QF}}(\Phi)$ is obtained from $\mathbf{C^{QF}}$ by extending the comprehension rule for all formulas in Φ. This crucial lemma combines the $\exists$-inversion Theorem

(Proposition 1.2.3) and Lemma 1.3.4 of [9]. The proof techniques used are similar to those from section C of [1], adapting them from induction to comprehension.

Lemma 10. *Let φ_i be open formulas of the language $\mathbf{L}^{Sk}$ for any $k \in \mathbb{N}$ and $0 < i \leq k$. If $\mathbf{D}$ is a normal derivation in $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma_1^b)$ of the set*

$$\Gamma, \neg W(\vec{x}), (\exists y)\varphi_1(\vec{\mathfrak{X}}, y), \ldots, (\exists y)\varphi_k(\vec{\mathfrak{X}}, y) \tag{5.1}$$

where Γ is purely existential with no second-order quantifier and $\vec{x}$ is the list of all first-order parameters of $\mathbf{D}$, then there are functionals F_{φ_i} of $\mathbf{L}^{\mathbf{P}}$ such that

$$\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma_1^b) \vdash \Gamma, \neg W(\vec{x}), \varphi_1(\vec{\mathfrak{X}}, F_{\varphi_1}(\vec{\mathfrak{X}})), \ldots, \varphi_k(\vec{\mathfrak{X}}, F_{\varphi_k}(\vec{\mathfrak{X}})).$$

Proof. We proceed by induction on the height of normal derivations. The crucial case is when the last inference is a CR–instance. Since Γ contains no second-order existential quantifier, the CR–inference must introduce one of the $\neg W(x_j)$ such that $x_j \in \vec{x}$. Weakening with $\neg W(x_j)$, we have a normal derivation of the set:

$$\begin{aligned}&\Gamma, \neg W(\vec{x}), (\exists y)\varphi_1(\vec{\mathfrak{X}}, y), \ldots, (\exists y)\varphi_k(\vec{\mathfrak{X}}, y),\\ &\qquad \theta(\vec{\mathfrak{X}}, \varepsilon) \wedge (\forall y)\big(\neg\,\theta(\vec{\mathfrak{X}}, y) \vee (\theta(\vec{\mathfrak{X}}, S^0(y)) \wedge \theta(\vec{\mathfrak{X}}, S^1(y)))\big) \wedge \neg\,\theta(\vec{\mathfrak{X}}, x_j)\end{aligned}$$

where θ is the abstract of the CR–instance. When the context is clear, we suppress variables $\vec{\mathfrak{X}}$ and $\vec{x}$ to simplify the notation. With $\wedge$– and $\forall$–inversions, an eigenvariable b is obtained. Further inversions and weakening with $\neg W(b)$, we can obtain normal derivations in $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma_1^b)$ of the following sets of formulas:

$$\begin{aligned}&\Gamma, \neg W(\vec{x}), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), \theta(\varepsilon)\\ &\Gamma, \neg W(\vec{x}), \neg W(b), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), \neg\,\theta(b), \theta(S^0(b))\\ &\Gamma, \neg W(\vec{x}), \neg W(b), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), \neg\,\theta(b), \theta(S^1(b))\\ &\Gamma, \neg W(\vec{x}), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), \neg\,\theta(x_j)\end{aligned}$$

The formula θ is either a s-Σ_1^b-formula or an open formula. We will discuss the case of s-Σ_1^b-formula first. θ is of the form $(\exists z)\theta^*(\vec{x}, b, z)$ where θ^* is open. The $\forall$–inversion of $\neg\,\theta$ yields an eigenvariable a. Weakening with $\neg W(a)$, we obtain derivations of the following sets:

$$\begin{aligned}&\Gamma, \neg W(\vec{x}), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), (\exists z)\theta^*(\varepsilon, z)\\ &\Gamma, \neg W(\vec{x}), \neg W(b), \neg W(a), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y),\\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \neg\,\theta^*(b, a), (\exists z)\theta^*(S^0(b), z)\\ &\Gamma, \neg W(\vec{x}), \neg W(b), \neg W(a), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y),\\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \neg\,\theta^*(b, a), (\exists z)\theta^*(S^1(b), z)\\ &\Gamma, \neg W(\vec{x}), \neg W(a), (\exists y)\varphi_1(y), \ldots, (\exists y)\varphi_k(y), \neg\,\theta^*(x_j, a)\end{aligned}$$

As all derivations of the above sets are of strictly smaller height than $\mathbf{D}$ and of the form of (5.1), the induction hypothesis is applicable. Hence there are functionals $G_{\varphi_i}(\vec{\mathfrak{x}})$, $H^*_{\varphi_i}(\vec{\mathfrak{x}}, a, b)$, $M^*_{\varphi_i}(\vec{\mathfrak{x}}, a, b)$, $N^*_{\varphi_i}(\vec{\mathfrak{x}}, a)$, $G_{\theta^*}(\vec{\mathfrak{x}})$, $H^*_{\theta^*}(\vec{\mathfrak{x}}, a, b)$, and $M^*_{\theta^*}(\vec{\mathfrak{x}}, a, b)$ of $\mathbf{L^p}$ and derivations of the following sets:

$$\begin{aligned}
&\Gamma, \neg W(\vec{x}), \varphi_1(G_{\varphi_1}(\vec{\mathfrak{x}})), \dots, \varphi_k(G_{\varphi_k}(\vec{\mathfrak{x}})), \theta^*(\varepsilon, G_{\theta^*}(\vec{\mathfrak{x}}))\\
&\Gamma, \neg W(\vec{x}), \neg W(b), \neg W(a), \varphi_1(H^*_{\varphi_1}(a,b)), \dots, \varphi_k(H^*_{\varphi_k}(a,b)),\\
&\qquad\qquad\qquad\qquad\qquad\qquad \neg\, \theta^*(b,a), \theta^*(S^0(b), H^*_{\theta^*}(a,b))\\
&\Gamma, \neg W(\vec{x}), \neg W(b), \neg W(a), \varphi_1(M^*_{\varphi_1}(a,b)), \dots, \varphi_k(M^*_{\varphi_k}(a,b)),\\
&\qquad\qquad\qquad\qquad\qquad\qquad \neg\, \theta^*(b,a), \theta^*(S^0(b), M^*_{\theta^*}(a,b))\\
&\Gamma, \neg W(\vec{x}), \neg W(a), \varphi_1(N^*_{\varphi_1}(a)), \dots, \varphi_k(N^*_{\varphi_k}(a)), \neg\, \theta^*(x_i, a)
\end{aligned}$$

Combining these derivations using $\vee$, $\wedge$-rules and weakening, we have a derivation of

$$\begin{aligned}
&\Gamma, \neg W(\vec{x}), \neg W(b), \neg W(a),\\
&\qquad \varphi_1(G_{\varphi_1}(\vec{\mathfrak{x}})), \varphi_1(H^*_{\varphi_1}(a,b)), \varphi_1(M^*_{\varphi_1}(a,b)), \varphi_1(N^*_{\varphi_1}(a)),\\
&\qquad\qquad\qquad\qquad\vdots\\
&\qquad \varphi_k(G_{\varphi_k}(\vec{\mathfrak{x}})), \varphi_k(H^*_{\varphi_k}(a,b)), \varphi_k(M^*_{\varphi_k}(a,b)), \varphi_k(N^*_{\varphi_k}(a)),\\
&\qquad \theta^*(\varepsilon, G_{\theta^*}(\vec{\mathfrak{x}})) \wedge \neg\, \theta^*(x_i, a)\\
&\qquad \wedge \left(\neg\, \theta^*(b,a) \vee \left(\theta^*(S^0(b), H^*_{\theta *}(a,b)) \wedge \theta^*(S^1(b), M^*_{\theta *}(a,b))\right)\right). \qquad (5.2)
\end{aligned}$$

Now, consider a functional $L_{\theta^*}(\vec{\mathfrak{x}}, y)$ s.t.

$$L_{\theta^*}(\vec{\mathfrak{x}}, y) = \begin{cases} \varepsilon & \text{if } \neg\, \theta^*(\varepsilon, G_{\theta^*}(\vec{\mathfrak{x}}))\\ \text{shortest } z & \text{else if } (\exists z \sqsubseteq y)\vartheta\\ y & \text{otherwise} \end{cases}$$

where ϑ is an open formula defined as:

$$\begin{aligned}
&\vartheta \overset{df}{=} \theta^*(z, G_{\theta^*}(\vec{\mathfrak{x}})) \wedge\\
&\qquad \Big(\left(\neg\, \theta^*(S^0(z), H^*_{\theta^*}(G_{\theta^*}(\vec{\mathfrak{x}}), z))\right) \vee \left(\neg\, \theta^*(S^1(z), M^*_{\theta^*}(G_{\theta^*}(\vec{\mathfrak{x}}), z))\right)\Big).
\end{aligned}$$

By Theorem 7, L_{θ^*} is a functional in $\mathbf{L^p}$. By its definition, we can prove in $\mathbf{C^{QF}}$ that:

$$\begin{aligned}
&\neg W(\vec{x}), \neg\, \theta^*(\varepsilon, G_{\theta^*}(\vec{\mathfrak{x}})), \theta^*(x_i, G_{\theta^*}(\vec{\mathfrak{x}})),\\
&\theta^*(L_{\theta^*}(x_i), G_{\theta^*}(\vec{\mathfrak{x}})) \wedge \begin{pmatrix} & \neg\, \theta^*(S^0(L_{\theta^*}(x_i)), H^*_{\theta^*}(G_{\theta^*}(\vec{\mathfrak{x}}), L_{\theta^*}(x_i)))\\ \vee & \neg\, \theta^*(S^1(L_{\theta^*}(x_i)), M^*_{\theta^*}(G_{\theta^*}(\vec{\mathfrak{x}}), L_{\theta^*}(x_i))) \end{pmatrix}
\end{aligned} \qquad (5.3)$$

Substituting a by $G_{\theta^*}(\vec{\mathfrak{x}})$ and b by $L(x_i)$ in (5.2) allows us to eliminate $\neg W(a)$ and $\neg W(b)$ (formally by cuts and Lemma 1); replace $H^*_{\varphi_i}(a,b)$ with $H_{\varphi_i}(\vec{\mathfrak{x}})$, $M^*_{\varphi_i}(a,b)$ with $M_{\varphi_i}(\vec{\mathfrak{x}})$ and $N^*_{\varphi_i}(a)$ with $N_{\varphi_i}(\vec{\mathfrak{x}})$. Further, we can apply the cut rule with (5.3) and remove all formulas which contain θ^*. Finally, we combine each $\varphi_i(G_{\varphi_i}(\vec{\mathfrak{x}}))$, $\varphi_i(H_{\varphi_i}(\vec{\mathfrak{x}}))$, $\varphi_i(M_{\varphi_i}(\vec{\mathfrak{x}}))$ and $\varphi_i(N_{\varphi_i}(\vec{\mathfrak{x}}))$ into $\varphi_i(F_{\varphi_i}(\vec{\mathfrak{x}}))$ by equation (4.1).

When θ is open, the proof proceeds nearly the same, except that there is no ∀–quantifier to invert in $\neg\,\theta$. We need a different search function $L_\theta(\vec{\mathfrak{x}}, y)$ for the substitution of b. The definition of $L_\theta(\vec{\mathfrak{x}}, y)$ is given as:

$$L_\theta(\vec{\mathfrak{x}}, y) = \begin{cases} \varepsilon & \text{if } \neg\,\theta(\varepsilon) \\ \text{shortest } z & \text{else if } (\exists z \sqsubseteq y)\left(\theta(z) \wedge \left(\neg\,\theta(S^0(z)) \vee \neg\,\theta(S^1(z))\right)\right) \\ y & \text{otherwise} \end{cases}.$$

□

Lemma 11. *Let $\phi(\vec{x})$ be a Σ^b_1-formula of the language $\mathbf{L}^{Sk}$, there is a s-Σ^b_1-formula $\phi^*(\vec{x})$ of the same language such that*

$$\mathbf{C^{QF}}(\Sigma^b_1) \vdash \neg W(\vec{x}), \phi(\vec{x}) \leftrightarrow \phi^*(\vec{x}).$$

Proof. This is a binary string version of Theorem 14 of 2.7 in [2]: The usual pairing function $\langle x, y\rangle$, the polynomial-time computable coding function $\beta(w, x)$, and all functions required to construct the bounding term $SqBd(x, y)$ are functions of $\mathbf{L}^p$. (Σ^b_1–$PIND$) is required in the proof of the Theorem 14 in [2]. We can get the relativized version $(\Sigma^b_1\text{–}PIND)^W$ in $\mathbf{C^{QF}}(\Sigma^b_1)$ by the argument for Lemma 6. Pushing all sharply bounded quantifiers into the scope of bounded existential quantifier using β and $SqBd$, replacing Σ^b_0-formulas with open ones and contracting any two bounded existential quantifiers into a single one by the pairing function gives us the desired s-Σ^b_1-formula. □

This equivalence requires a relativization to $\mathbf{W}$ in systems with comprehension, in contrast to the unconditional equivalence in systems with induction. For this reason, the proof of the next lemma is a bit harder. Nonetheless, the techniques used for proving Lemma 1.3.4 in [9] and Theorem 9 in [1] are helpful.

Lemma 12. *$\mathbf{C^{QF}}(\Sigma^b_1)$ is conservative over $\mathbf{C^{QF}}(s\text{-}\Sigma^b_1)$ for Π^0_2–formulas when relativized to $\mathbf{W}$.*

Proof. By inversions, we only have to establish $\mathbf{C^{QF}}(\Sigma^b_1)$ is conservative over $\mathbf{C^{QF}}(s\text{-}\Sigma^b_1)$ for sets of the form $\Gamma, \neg W(\vec{x})$ where Γ is purely existential and $\vec{x}$ is the list of all first-order parameters of the corresponding derivations.

We proceed by induction on the number n of CR–instances in a normal derivation $\mathbf{D}$ of the above set where the abstracts of these instances are not valid abstracts in derivations in $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma^b_1)$. The case $n = 0$ is trivial. Let $\mathbf{D}$ have $k + 1$ such CR–instances. By weakening we can assume that every set of $\mathbf{D}$ contains the set of formulas $\neg W(\vec{x})$. Consider one of the top-most applications of such a rule with premise Δ, $\neg W(\vec{x})$, $\varphi(V_\theta)$, where formula θ is neither a $s\text{-}\Sigma^b_1$-formula nor an open one. The basic idea is to replace θ with its $s\text{-}\Sigma^b_1$–equivalent θ^*. However, the equivalence is valid only when all first-order variables range over $\mathbf{W}$, including those eigenvariables.

Let $\mathbf{D}_1$ be the immediate sub-derivation of the above set and $\vec{\mathfrak{E}}$ be the eigenvariables which are quantified below $\mathbf{D}_1$. In particular, $\vec{e} = e_0, \dots, e_m$. By weakening with $\neg W(\vec{e})$ and using Lemma 11, we can obtain a $s\text{-}\Sigma^b_1$-formula θ^* and a derivation $\mathbf{D}^\Box_1$ in $\mathbf{C}^{\mathbf{QF}}(\Sigma^b_1)$ of Δ, $\neg W(\vec{x})$, $\neg W(\vec{e})$, $\varphi(V_{\theta^*})$. By removing $\mathbf{D}_1$ from $\mathbf{D}$, we obtain the sub-tree $\mathbf{D} \setminus \mathbf{D}_1$. We need to transform both $\mathbf{D}^\Box_1$ and $\mathbf{D} \setminus \mathbf{D}_1$ so they can be combined and get the same conclusion of $\Gamma, \neg W(\vec{x})$. The following procedures transform $\neg W(\vec{e})$ into simpler formulas. We denote by $\overrightarrow{e_{<j}}$ all variables whose quantifications are below that of e_j; in particular, $\overrightarrow{e_{<0}}$ is the empty vector of variables. Starting with $\mathbf{D}^\Box_1$, we perform a proof transformation for each variable e_j in descending order of j, with the result of each pass used in the next. In each pass, we obtain a formula λ^e_j; we denote by $\Lambda^e_{>j}$ the set $\{\lambda^e_i \mid e_i \in \vec{e} \text{ and } i > j\}$ which is of course empty when $j = m$. The transformation is given as follow:

1. If e_j is quantified in an unbounded $\forall$–instance, the unbounded quantifier must correspond to $\neg W(v)$, where $v \in \vec{x}$ or $v \in \overrightarrow{e_{<j}}$. We set $\lambda^e_j = \neg(e_j \preccurlyeq v)$. Using the cut rule with a $\mathbf{C}^{\mathbf{QF}}$-derivation of $\neg W(v)$, $\neg(e_j \preccurlyeq v)$, $W(e_j)$ (from Corollary 5), we obtain a derivation of
$$\Delta, \neg W(\vec{x}), \varphi(V_{\theta^*}), \neg W(\overrightarrow{e_{<j}}), \neg(e_j \preccurlyeq v), \Lambda^e_{>j}.$$

2. If e_j is quantified by a bounded $\forall$–instance, there must be a term $t_j(\vec{x}, \overrightarrow{e_{<j}})$ in $\mathbf{D}$ such that the $\forall$–e_j has the principal formula of the form $(\forall e_j \preccurlyeq t_j(\vec{x}, \overrightarrow{e_{<j}}))\psi(\vec{x}, \overrightarrow{e_{<j}}, e_j)$. We set $\lambda^e_j = \neg(e_j \preccurlyeq t_j(\vec{x}, \overrightarrow{e_{<j}}))$. Using a cut rule with a $\mathbf{C}^{\mathbf{QF}}$-derivation of the set $\neg W(\vec{x})$, $\neg W(\overrightarrow{e_{<j}})$, $\neg(e_j \preccurlyeq t_j(\vec{x}, \overrightarrow{e_{<j}}))$, $W(e_j)$ (again from Corollary 5), we obtain a derivation of
$$\Delta, \neg W(\vec{x}), \varphi(V_{\theta^*}), \neg W(\overrightarrow{e_{<j}}), \neg(e_j \preccurlyeq t_j(\vec{x}, \overrightarrow{e_{<j}})), \Lambda^e_{>j}.$$

3. If e_j is quantified by a sharply bounded $\forall$–instance, there must be a term $t_j(\vec{x}, \overrightarrow{e_{<j}})$ in $\mathbf{D}$ such that the $\forall$–e_j has the principal formula of the form $(\forall e_j \sqsubseteq t_j(\vec{x}, \overrightarrow{e_{<j}}))\psi(\vec{x}, \overrightarrow{e_{<j}}, e_j)$. We set $\lambda^e_j = \neg(e_j \sqsubseteq t_j(\vec{x}, \overrightarrow{e_{<j}}))$. From Corollary 5 and a $\mathbf{C}^{\mathbf{QF}}$-derivation of $\neg(e_j \sqsubseteq t_j(\vec{x}, \overrightarrow{e_{<j}}))$, $e_j \preccurlyeq t_j(\vec{x}, \overrightarrow{e_{<j}})$, we get a $\mathbf{C}^{\mathbf{QF}}$-derivation of the set $\neg W(\vec{x})$, $\neg W(\overrightarrow{e_{<j}})$, $\neg(e_j \sqsubseteq$

$t_j(\vec{x}, \overrightarrow{e_{<j}}))$, $W(e_j)$,. By an application of cut rule, we obtain a derivation of

$$\Delta, \neg W(\vec{x}), \varphi(V_{\theta^*}), \neg W(\vec{e}_{<j}), \neg(e_j \sqsubseteq t_j(\vec{x}, \overrightarrow{e_{<j}})), \Lambda^e_{>j}.$$

Denoting by Λ^e the set $\bigcup_{=0}^{m}\{\lambda^e_j\}$, the above transformation replaces all $\neg W(\vec{e})$ with Λ^e. We now apply a CR–rule with θ^* as abstract. The result is a $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma^b_1)$-derivation of the set Δ, $\neg W(\vec{x})$, $\exists X\varphi(X)$, Λ^e. We call the *normal* $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma^b_1)$-derivation of the same set $\mathbf{D}^*_1$.

Though Λ^e is simpler than $\neg W(\vec{e})$, the presence of $\vec{e}$ in these extra formulas makes the $\forall$–e_j inapplicable in $\mathbf{D} \setminus \mathbf{D}_1$. We need to transform this sub-tree to $(\mathbf{D} \setminus \mathbf{D}_1)^*$ so it is suitable for grafting back $\mathbf{D}^*_1$. Transformation procedures are as follows:

1. For every $\forall$–e_j instance, we add λ^e_j to all sets above such a rule as a side formula.

2. If the $\forall$–e_j is a bounded or sharply bounded instance, it is easy to eliminate this side formula while the sub-tree remains normal as its principal formula contains λ^e_j.

3. If the $\forall$–e_j is an unbounded instance, the minor formula must contain

   $$\neg\,\theta_j(e_j, \vec{x}, \overrightarrow{e_{<j}}) \vee \left(\theta_j(S^0(e_j), \vec{x}, \overrightarrow{e_{<j}}) \wedge \theta_j(S^1(e_j), \vec{x}, \overrightarrow{e_{<j}})\right).$$

   Let us denote by θ^*_j the Σ^b_1-formula $\theta_j(e_j, \vec{x}, \overrightarrow{e_{<j}}) \vee \neg(e_j \preccurlyeq v)$. With a few inferences, we can transform θ_j to θ^*_j in the above formula and eliminate λ_j, while the sub-tree remains normal.

4. The unbounded quantified formula with θ_j will eventually be a part of the minor formula for a CR–instance for $\neg W(v)$. Thus, the minor formula of the CR–instance also contains $\theta_j(\varepsilon, \vec{x}, \overrightarrow{e_{<j}})$ and $\neg\,\theta_j(v, \vec{x}, \overrightarrow{e_{<j}})$. Using similar inferences, we transform both to $\theta^*_j(\varepsilon, \vec{x}, \overrightarrow{e_{<j}})$ and $\neg\,\theta^*_j(v, \vec{x}, \overrightarrow{e_{<j}})$ respectively.

5. Using θ^*_j as a new abstract, we can get the same result of $\neg W(v)$.

Finally, we graft $\mathbf{D}^*_1$ back to $(\mathbf{D} \setminus \mathbf{D}_1)^*$ and obtain a normal $\mathbf{C}^{\mathbf{QF}}(\Sigma^b_1)$-derivation of the same set with only k CR–instances of which the abstracts are not valid in $\mathbf{C}^{\mathbf{QF}}(s\text{-}\Sigma^b_1)$. By induction, Lemma 12 is shown. □

6 Definable functions and the Polynomial Hierarchy

Definition 6. *Let $\mathbf{T}$ be a theory in $\mathbf{L} \supseteq \mathbf{L}^b$ such that $\mathbf{T}$ extends $\mathbf{C}^b$.*

1. *Φ-$DF^W(\mathbf{T})$ is the class of Φ-definable functionals in $\mathbf{T}$.*
2. *$Fn^W(\mathbf{L})$ is the collection of the restrictions of functionals of $\mathbf{L}$ to $\mathbf{W}$.*
3. *Similar to the definition of s-Σ_1^b, the class of formulas s-Σ_1^0 consists of all formulas of the form $(\exists y)\varphi(\vec{x}, y)$ where $\varphi(\vec{x}, y)$ is open.*

Lemma 13. *s-Σ_1^0-$DF^W(\mathbf{C^{QF}}(s\text{-}\Sigma_1^b)) \subseteq Fn^W(\mathbf{L^P})$*

Proof. A functional $F(\vec{\mathfrak{x}})$ that belongs to s-Σ_1^0-$DF^W(\mathbf{C^{QF}}(s\text{-}\Sigma_1^b))$, has a defining formula $(\exists z)\varphi_F(\vec{\mathfrak{x}}, y, z)$ where φ_F is open. The existence condition for $F(\vec{\mathfrak{x}})$ means:

$$\mathbf{C^{QF}}(s\text{-}\Sigma_1^b) \vdash \big((\forall\vec{\mathfrak{x}})(\exists y)(\exists z)\varphi_F(\vec{\mathfrak{x}}, y, z)\big)^W.$$

After inversions, we can obtain a derivation of the set

$$\neg W(\vec{x}), (\exists y)\Big(W(y) \wedge (\exists z)\Big(\varphi_F(\vec{\mathfrak{x}}, y, z) \wedge W(z)\Big)\Big).$$

Since both y and z belong to $\mathbf{W}$, the pairing function enables us to contract two existential quantifiers into one. With a few inferences, we obtain:

$$\mathbf{C^{QF}}(s\text{-}\Sigma_1^b) \vdash \neg W(\vec{x}), (\exists u)\varphi_F(\vec{\mathfrak{x}}, u|_1, u|_2)$$

where u is $< y, z >$ and its coordinates of are denoted by $u|_1$ and $u|_2$ respectively. By Lemma 10, there is a functional $F_\varphi(\vec{\mathfrak{x}}) \in Fn^W(\mathbf{L^P})$ such that $\mathbf{C^{QF}}(s\text{-}\Sigma_1^b)$ proves $\neg W(\vec{x})$, $\varphi_F(\vec{\mathfrak{x}}, F_\varphi(\vec{\mathfrak{x}})|_1, F_\varphi(\vec{\mathfrak{x}})|_2)$. Applying the $(\exists^1)$ inference, we obtain:

$$\neg W(\vec{x}), (\exists z)\varphi_F(\vec{\mathfrak{x}}, F_\varphi(\vec{\mathfrak{x}})|_1, z).$$

The uniqueness condition for $F(\vec{\mathfrak{x}})$ implies $F(\vec{\mathfrak{x}}) = F_\varphi(\vec{\mathfrak{x}})|_1$. $Fn^W(\mathbf{L^P})$ is closed under composition, hence $F_\varphi(\vec{\mathfrak{x}})|_1 \in Fn^W(\mathbf{L^P})$. □

Theorem 14. *Σ_1^b-$DF^W(\mathbf{C^{QF}}(\Sigma_1^b))$ is the class of polynomial-time computable functionals; thus, the Σ_1^b-definable functions of the theory $\mathbf{C^{QF}}(\Sigma_1^b)$ are exactly the polynomial-time computable functions.*

Proof. By Theorem 4, the class of polynomial-time computable functionals is $Fn^W(\mathbf{L^P})$. Extending the argument for Lemma 3 to functionals, we have $Fn^W(\mathbf{L^P}) \subseteq \Sigma_1^b$-$DF^W(\mathbf{C}^b(\Sigma_1^b))$. As a result of the previous lemmas, we have the other direction of the subset relation, depicted by the following table:

Claims	**Justifications**
Σ_1^b-$DF^W(\mathbf{C}^b(\Sigma_1^b))$	
$\subseteq \Sigma_1^b$-$DF^W(\mathbf{C^{QF}}(\Sigma_1^b))$	Corollary 9
$= s$-Σ_1^b-$DF^W(\mathbf{C^{QF}}(\Sigma_1^b))$	Lemma 11
$= s$-Σ_1^b-$DF^W(\mathbf{C^{QF}}(s\text{-}\Sigma_1^b))$	Lemma 12
$\subseteq s$-Σ_1^0-$DF^W(\mathbf{C^{QF}}(s\text{-}\Sigma_1^b))$	s-$\Sigma_1^b \subseteq s$-Σ_1^0
$\subseteq Fn^W(\mathbf{L^P})$	Lemma 13

Thus, all classes above are actually the class of polynomial-time computable functionals, including Σ_1^b-$DF^W(\mathbf{C}^{\mathbf{QF}}(\Sigma_1^b))$. □

Definition 7.

1. $\mathbf{L}^{p_1} \overset{df}{=} \mathbf{L}^p$, $BASIC^{p_1} \overset{df}{=} BASIC^p$, $\mathbf{C}_1^{\mathbf{QF}} \overset{df}{=} \mathbf{C}^{\mathbf{QF}}$

2. *For each purely first-order open formula* φ *and each term* t *of* $\mathbf{L}^{p_i}$*, there is a* Σ_1^b*-Skolem function* $f_{\varphi,t}$ *in* $\mathbf{L}^{p_{i+1}}$ *such that*

$$(\exists y \preccurlyeq t(\vec{x}))\varphi(\vec{x}, y) \to (\varphi(\vec{x}, f_{\varphi,t}(\vec{x})) \wedge (f_{\varphi,t} \preccurlyeq t(\vec{x})))$$

while $BASIC^{p_{i+1}}$ *contains* $BASIC^{p_i}$ *and two additional sets of formulas,* $\{\neg(y \preccurlyeq t(\vec{x})), \neg\varphi(\vec{x}, y), \varphi(\vec{x}, f_{\varphi,t}(\vec{x}))\}$ *and* $\{f_{\varphi,t}(\vec{x}) \preccurlyeq t(\vec{x})\}$*, for each* $f_{\varphi,t}$*.*

3. *For all* $i > 1$*, the language* $\mathbf{L}^{\mathbf{p}_i}$ *and the theory* $\mathbf{C}_i^{\mathbf{QF}}$ *are obtained from* $BASIC^{p_i}$ *in the same way as* $\mathbf{L}^{\mathbf{p}}$ *and* $\mathbf{C}^{\mathbf{QF}}$ *from* $BASIC^p$*, i.e. adding functionals which are defined by composition and limited recursion, adding* $\mathbf{\Sigma_0^b}$*–Skolem functionals and using open formulas for* CR*–rule.*

Theorem 15. Σ_i^b*-definable functions of* $\mathbf{C}^b(\Sigma_i^b)$ *are exactly* $\square_i^p$ *functions.*

Proof. It is clear that $f_{\varphi,t}$ of $\mathbf{L}^{p_{i+1}}$ belongs to $\square_{i+1}^p$. For any Σ_i^b-formula φ in $\mathbf{L}^b$, there is a Σ_1^b-formula φ^* in $\mathbf{L}^{p_i}$ such that $\mathbf{C}_i^{\mathbf{QF}} \vdash \varphi \leftrightarrow \varphi^*$. Thus, $\mathbf{C}_i^{\mathbf{QF}}(\Sigma_1^b)$ is a conservative extension of $\mathbf{C}^b(\Sigma_i^b)$. All our previous lemmas hold for $\mathbf{C}_i^{\mathbf{QF}}$ as all new functionals are bounded by terms of the previous theory and ultimately by terms in $\mathbf{L}^b$. Following the logic of Theorem 14, one obtains Σ_i^b-$DF^W(\mathbf{C}^b(\Sigma_i^b)) \subseteq Fn^W(\mathbf{L}^{\mathbf{p}_i})$. Restricting our discussion to functions, we get the converse of Lemma 3. □

Finally, we would like to mention that the uniformity of the proof-theoretic analysis of $\mathbf{C}^b(\Sigma_i^b)$ in this paper, $\Sigma_1^b(\mathfrak{P})$–$PIND$ in [9] and s-$\Sigma_1^b(\mathcal{P})$–NIA in [1] supports our belief that the method of Herbrand analysis will see many more applications in Proof Theory.

Bibliography

[1] Wilfried Buchholz and Wilfried Sieg. A note on polynomial time computable arithmetic. *Contemporary Mathematics*, 106:51–55, 1990.

[2] Samuel R. Buss. *Bounded Arithmetic.* Bibliopolis, 1986.

[3] Fernando J. I. Ferreira. *Polynomial Time Computable Arithmetic and Conservative Extensions.* PhD thesis, The Pennsylvania State University, 1988.

[4] Fernando J. I. Ferreira. Stockmeyer induction. In S.R. Buss and P.J. Scot, editors, *Feasible Mathematics, proceedings of the Mathematical Sciences Institute Workshop*, pages 161–180, Ithaca, New York, June 1989. Birkhäuser, 1990.

[5] Aleksandar Ignjatović. Delineating classes of computational complexity via second order theories with weak set existence principles (i). *The Journal of Symbolic Logic*, 60:103–121, 1995.

[6] Daniel Leviant. A foundational delineation of computational feasibility. *Logic in Computer Science, 1991. LICS '91., Proceedings of Sixth Annual IEEE Symposium on*, pages 2–11, 15-18 July 1991.

[7] Helmut Schwichtenberg. Proof theory: Some applications of cut-eliminiation. In J. Barwise, editor, *Handbook of Mathematical Logic*, chapter D.2, pages 867–895. North-Holland, 1977.

[8] Wilfried Sieg. Fragments of arithmetic. *Annals of Pure and Applied Logic*, 28:33–71, 1985.

[9] Wilfried Sieg. Herbrand analyses. *Archive for Mathematical Logic*, 30(5-6):409–441, 1991.

[10] Gaisi Takeuti. *Proof Theory.* North-Holland, second edition, 1987.