1 Introduction

An input for a Turing Machine (TM) can be considered as a sequence of numeric values in a \( n \)-adic number system where \( n \) is the size of alphabet. A TM computation for decision problem can then be transformed as a evaluation of a numeric function whose range is \( \{0, 1\} \). With some basic functions, Buss [1] built a hierarchy of bounded arithmetic theories \( S^i_2 \) which characterizes the polynomial hierarchy (PH) [1]. We will discuss \( S^1_2 \) briefly (up to chapter 3 of [1]), with focus on \( S^1_2 \), and complete the discussion with Herbrand analyse of \( S^i_2 \) [3] as a simpler alternative to witness theorem (chapter 5 in [1]).

2 Limited Iteration and Polynomial Hierarchy

Definition 1.

- A function \( f(\vec{x}) \) is defined by limited recursion from \( g(\vec{x}) \) and \( h(\vec{x}, y, z) \) with time bound \( p \) and space bound \( q \) iff the followings hold

\[
\begin{align*}
\tau(\vec{x}, 0) &= g(\vec{x}, 0) \\
\tau(\vec{x}, y') &= h(\vec{x}, y, \tau(\vec{x}, y)) \\
f(\vec{x}) &= \tau(\vec{x}, p(|\vec{x}|))
\end{align*}
\]

and

\[(\forall n \leq p(|\vec{x}|)) [\tau(\vec{x}, n) \leq q(|\vec{x}|)]\]

where \( p \) and \( q \) are polynomials with non-negative integer coefficients.

- The collection \( \mathcal{B} \) of numeric functions contains the following functions:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The constant zero function</td>
</tr>
<tr>
<td>( 2 \cdot x )</td>
<td>The left-shift function</td>
</tr>
<tr>
<td>( \lfloor \frac{x}{2} \rfloor )</td>
<td>The right-shift function</td>
</tr>
</tbody>
</table>
| \( \text{leq}(x, y) \) | \( \begin{cases} 
1 & \text{if } x \leq y \\
0 & \text{otherwise} 
\end{cases} \) |
| \( \text{Choice}(x, y, z) \) | \( \begin{cases} 
y & \text{if } x > 0 \\
z & \text{otherwise} 
\end{cases} \) |
• For a collection of functions $\mathcal{F}$, $\text{Cl}(\mathcal{F})$ is closure of $\mathcal{F}$ under composition and limited iteration.

• $\mathcal{P} \overset{df}{=} \text{Cl} (\mathcal{B})$.

• $\mathcal{P}_1$ is the collection of functions which are computable by a polynomial-time TM.

**Theorem 1.** $\mathcal{P} = \mathcal{P}_1$

**Proof.** This is the Theorem 2 in [1]. We will only give a brief sketch of the proof.

**Case 1:** $\mathcal{P} \subseteq \mathcal{P}_1$

By induction on complexity of definition of $f \in \mathcal{P}$. All functions in $\mathcal{B}$ are $p$–time computable. If an oracle function is $p$–time computable by a TM, adding the oracle to the TM will not change its ability for $p$-time computation. Composition is just an oracle consultation. It is not difficult to see that limited iteration captures the idea of polynomial-time computation of a TM with oracles for previously-defined functions. Hence we have this side of inclusion.

**Case 2:** $\mathcal{P}_1 \subseteq \mathcal{P}$

By encoding the instant description (ID) of a TM. This is very similar to the class proof of SAT being a NP-complete problem [2]. The coding scheme used in this section of [1] is different from the one about $S^2_1$ in the later chapter of the same book. Nonetheless, all functions necessary for encoding/decoding the ID’s of a TM are definable from $\mathcal{B}$ using composition and limited iteration.

$\square$

## 3 Bounded Arithmetic

**Definition 2.**

• The language of bounded arithmetic, $\mathcal{L}_{BA}$, is given as

$$\mathcal{L}_{BA} = \{0, x', +, \cdot, |x|, \lfloor \frac{x}{2} \rfloor, \#, =, \leq \}$$

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meanings</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>the zero constant (function)</td>
</tr>
<tr>
<td>$x'$</td>
<td>successor function</td>
</tr>
<tr>
<td>$x + y$</td>
<td>addition</td>
</tr>
<tr>
<td>$x \cdot y$</td>
<td>multiplication</td>
</tr>
<tr>
<td>$</td>
<td>x</td>
</tr>
<tr>
<td>$\lfloor \frac{x}{2} \rfloor$</td>
<td>largest integer smaller than or equal to $x/2$</td>
</tr>
<tr>
<td>$x # y$</td>
<td>$x # y = 2^{</td>
</tr>
</tbody>
</table>

• $\mathcal{B} \overset{df}{=} \{0, x', +, \cdot, |x|, \lfloor \frac{x}{2} \rfloor, \#$, i.e. the collection of all functions in $\mathcal{L}_{BA}$.}
• **BASIC** is the set of open formulas that defines the functions in $\mathfrak{B}$. Details can be found in chapter 2 of [1].

• $\Phi$–*PIND* axioms are of the form $(\varphi(0) \land \forall x[\varphi(\left\lceil \frac{n}{2} \right\rceil) \rightarrow \varphi(x)]) \rightarrow \forall x\varphi(x)$ where $\varphi \in \Phi$.

$\Phi$–*PIND* can be incorporated into a Tait-style calculus as an inference rule as follows:

$$
\frac{
(\Phi$–*PIND*) $\quad \Gamma, \varphi(0) \quad \Gamma, \neg\varphi(\left\lceil \frac{n}{2} \right\rceil), \varphi(x) 
}{\Gamma, \varphi(t)}
$$

where $x$ does not occur free in $\Gamma$ and $t$ is any term in the language.

**Definition 3.**

• Sharply bounded quantifiers are of the form $\exists x \leq |t|$ or $\forall x \leq |t|$ where $t$ is a term in the language.

• $QF(\mathcal{L})$ is the set of all open formulas in the language $\mathcal{L}$. They are quantifier-free.

• $\Sigma^b_{i+1}(\mathcal{L})$ is the closure of $QF(\mathcal{L})$ under connectives and sharply bounded quantifications.

• $\Delta^b_{i}(\mathcal{L}) \overset{df}{=} \Sigma^b_{i}(\mathcal{L}) \overset{df}{=} \Pi^b_{i}(\mathcal{L})$

• $\Sigma^b_{i+1}(\mathcal{L})$ is the closure of $\Pi^b_{i}(\mathcal{L})$ under $\land$, $\lor$, sharply bounded quantifications and bounded existential quantification, i.e. $\exists x \leq t$.

• $\Pi^b_{i+1}(\mathcal{L})$ is the closure of $\Sigma^b_{i}(\mathcal{L})$ under $\land$, $\lor$, sharply bounded quantifications and bounded universal quantification, i.e. $\forall x \leq t$.

• With respect to a theory $T$ in $\mathcal{L}$, a formula $\delta \in \Delta^b_{i}(\mathcal{L})$ iff $T \vdash (\delta \leftrightarrow \varphi) \land (\delta \leftrightarrow \psi)$ where $\varphi \in \Sigma^b_{i}(\mathcal{L})$ and $\psi \in \Pi^b_{i}(\mathcal{L})$.

• The theory $S^b_2 \overset{df}{=} \text{BASIC} + \Sigma^b_1$–*PIND* in [1]. Using notation in [3], $S^b_2 \overset{df}{=} \Sigma^b_1(\mathfrak{B})$–*PIND*.

**4 Definable Functions and Conservative Extension**

**Definition 4.** For a theory $T$ and a class of formula $\Phi$,

• A function $f(\vec{x})$ is $\Phi$–definable in $T$ iff

$$
T \vdash \forall \vec{x} \exists y \varphi(\vec{x}, y) \\
T \vdash \forall \vec{x}yz[(\varphi(\vec{x}, y) \land \varphi(\vec{x}, z)) \rightarrow (y = z)] \\
\mathbb{N} \models \forall \vec{x} \varphi(\vec{x}, f(\vec{x}))
$$

where $\varphi \in \Phi$. 

3
The collection of all $\Phi$-definable functions in $T$ is denoted by $\Phi$–DF($T$).

Definition 5. Let $T_1$ be a theory in a language $L_1$ and $T_2$ in $L_2$ where $L_1 \subseteq L_2$. If $T_2 \vdash \varphi$ implies $T_1 \vdash \varphi$ for every formula $\varphi$ in $L_1$, we say $T_2$ is a conservative extension of $T_1$.

Theorem 2. For a theory $T$ with $\Sigma_{i+1}^b$–PIND, extending $T$ with symbols for $\Sigma_{i+1}^b$–definable functions and $\Delta_i^b$–definable predicates yields a conservative extension.

Proof. Let the original language be $L$ and $\varphi_f \in \Sigma_{i+1}^b(L)$ be the defining formula of a function $f$ in the extended language. We have $T \vdash_e \exists! y \varphi_f(y)$. For a derivation of $\Gamma$ in the extended theory where $\Gamma$ is in $L$, we can replace $\psi(f)$ with $\psi(y) \land \varphi_f(y)$ and combine a suitable subderivation of $d$ to get a valid derivation of $\Gamma$ in $T$. Note that if $\psi(f)$ is used in $\Sigma_{i+1}^b$–PIND in the extended system, $\psi(y) \land \varphi_f(y) \in \Sigma_{i+1}^b(L)$ and $\Sigma_{i+1}^b$–PIND is applicable to the formula. Similar arguments apply for predicates, by replacing the predicates with the defining formulas. To ensure sure $\Sigma_{i+1}^b$–PIND is applicable to the replaced formula, predicate should be $\Delta_i^b$–definable, since the predicates may be within the scope of negation. □

By theorem (2), we can use symbols for functions and predicates which are $\Sigma_i^b$–definable functions and $\Delta_i^b$–definable respectively in $S_2^1$ to simplify our discussion. In [1], it is called “bootstrapping”. The objective is to define functions which are similar to Gödel’s coding/decoding functions for a sequence of numbers in $S_2^1$. We won’t go into details about all these definitions. Instead, we illustrate the meanings of these functions by example.

To encode a number, we turn it into binary representation, insert a “1” before every binary digit, e.g. the number 10 is 1010 in binary and is coded as 11101110, i.e. 238. Similarly, the number 3 is coded as 1111, i.e. 15. The number 0 is coded as 10, i.e 2. To code the sequence of numbers, we first code the individual numbers and add “00’ as the comma. Thus the code representing the sequence $<0,10,3>$ is 1001111011100011111111, i.e. 146319. The functions used for decoding are also definable in $S_2^1$. Using the sequence $<0,10,3>$ as an example, we have

\[
\begin{align*}
\text{len}(146319) &= 3 \\
\beta(146319, 1) &= 0 \\
\beta(146319, 2) &= 10 \\
\beta(146319, 3) &= 3
\end{align*}
\]

Not every number is a valid sequence but every (finite) sequence can be coded into a unique number. Suppose the largest number in a sequence is $a$ and there are total $b$ numbers in it. The code $w$ for the sequence is bound by

\[
S_{QBd}(a, b) = (2 \cdot b + 1)\#(4 \cdot (2 \cdot a + 1)^2)
\]

Theorem 3. $\Psi \subseteq \Sigma_1^b$–DF($S_2^1$)
Proof. By induction on complexity of definition of $f \in \mathcal{P}$. It is clear that the functions in $\mathcal{B}$ is $\Sigma^b_1$–definable in $S^1_2$. Induction step for the composition case relies on the existence of the code $w$ which encodes the sequence of all component functions in the arguments. It is left to the reader to work out the details.

For limited iteration, it is enough to find a number $w$ that encodes the sequence of results of each iteration. The $SqBd$ function gives us the bound of $w$ so we can express the defining formula with a $\Sigma^b_1$–formula. Suppose $f \in \mathcal{P}$ is defined from $g$ and $h$ by limited iteration, the following formula is equivalent to $f(\vec{x}) = y$ and is provable in $S^1_2$

$$\exists!y(\exists w \leq SqBd(2^{q(|\vec{x}|)}, 2^{p(|\vec{x}|)}))$$

$$\left[ \begin{array}{l}
\beta(1, w) = g(\vec{x}) \\
\land \ (\forall i < len(w) - 1) [\beta(i'', w) = h(\vec{x}, i, \beta(i', w))] \\
\land \ y = \beta(len(w), w)
\end{array} \right]$$

and we can use subformula of the above as the defining formula for $f(\vec{x})$. Strictly speaking, the defining formula in this form is not a $\Sigma^b_1$–formula. Looking up the definition of $len$, we can replace $(\forall i < len(w) - 1)$ with its equivalence in term of $|w|$, and get a equivalent sharply bounded quantification. Hence there is a defining $\Sigma^b_1$–formula for $f(\vec{x})$. \qed

Note: The proofs about definable functions are really sketchy. They are presented in such a way in order to illustrate the idea without going too deep into details. Readers are strongly encouraged to refer to [1].

5 Herbrand Analysis for $S^1_2$

We will show the other way of inclusion of theorem (3) by technique used in [3], as an simpler alternative to chapter 5 of [1].

Theorem 4. (Herbrand Theorem) For a theory $T$ which is specified by open formulas as axioms, if $T \vdash \exists y \varphi(y)$, we have a finite number of term $t_1, \ldots, t_k$ s.t.

$T \vdash \varphi(t_1), \ldots, \varphi(t_k)$

Proof. By induction on the length of a derivation of $\Gamma, \exists y \varphi(y)$ where $\Gamma$ contains only open formula. \qed

Definition 6.

- $\mathcal{L}_w$ is the language obtained from $\mathcal{L}_{BA}$ by adding symbol for each function in $\mathcal{P}$.
- $\Phi(\mathcal{P})$–$PIND$ is the theory obtained from theory which defines all functions in $\mathcal{P}$ with $(\Phi$–$PIND)$ inference rule.\footnote{It should be noticed that all axioms of $\Phi(\mathcal{P})$–$PIND$ are open formulas.}
Theorem 5. $\mathfrak{P}$ is closed under definition by (finite) cases where the conditions are open formulas.

Proof. Firstly, we show that the following functions for each atomic formula and connectives are in $\mathfrak{P}$:

$$
\begin{align*}
  f_\wedge(x,y) &= \text{choice}(\text{leq}(x,y),\text{leq}(y,x),0) \\
  f_\leq(x,y) &= \text{leq}(x,y) \\
  f_\neg(x) &= \text{choice}(x,0,1) \\
  f_\land(x,y) &= \text{choice}(x,\text{leq}(1,y),0) \\
  f_\lor(x,y) &= \text{choice}(x,1,\text{leq}(1,y))
\end{align*}
$$

Then, we have the characteristic function $f_\varphi \in \mathfrak{P}$ for every open formula $\varphi$ in $\mathcal{L}_\mathfrak{P}$ by induction on the complexity of $\varphi$.

If a function $f$ is defined by case s.t.

$$f(\vec{x}) = \begin{cases} 
  t_1 & \text{if } \varphi_1(\vec{x}) \\
  t_2 & \text{else if } \varphi_2(\vec{x}) \\
  \vdots 
\end{cases},$$

it is easy to give the same definition by choice function:

$$\text{choice}(f_{\varphi_1}(\vec{x}),t_1,\text{choice}(f_{\varphi_2}(\vec{x}),t_2,\text{choice}(\ldots)))$$

Corollary 6. $QF-DF(QF(\mathfrak{P})-PIND) \subseteq \mathfrak{P}$

Proof. Immediate consequence of theorems (4) and (5).

Theorem 7. (Term Extraction) If $QF(\mathfrak{P})-PIND \vdash_d \Gamma, \exists y \varphi(\vec{x},y)$ where $\Gamma$ contain no universal quantification, $\varphi$ is open and $\vec{x}$ is the list of all parameters of the derivation $d$, then we have $QF(\mathfrak{P})-PIND \vdash \Gamma, \varphi(\vec{x},f(\vec{x}))$ where $f \in \mathfrak{P}$.

Proof. By modifying the proof for theorem (4) for purely existential formulas and using Corollary (6). It should be noted that the argument does not work if $\Gamma$ contains universal quantified formulas since $\forall$-inversions introduce extra variables not in $\vec{x}$.

Theorem 8. For an open formula $\varphi$ and a term $t$ in $\mathcal{L}_\mathfrak{P}$, there is a function $f_\varphi$ in $\mathfrak{P}$ s.t.

$$\Sigma^b_1(\mathfrak{P})-PIND \vdash (\exists y \leq |t|)\varphi(\vec{x},y) \leftrightarrow \varphi(\vec{x},f_\varphi(\vec{x}))$$

Proof. By construction of a thorough search function, begin with 0 up to $|t|$. If no such witness is found, the search function returns $|t| + 1$. Such function is $\Sigma^b_1$-definable in $\Sigma^b_1(\mathfrak{P})-PIND$, using the sequence coding technique.

2In [3], it is called purely existential.
Corollary 9. For any formula $\varphi \in \Sigma_0^b(\mathfrak{P})$ there is a formula $\varphi^* \in QF(\mathfrak{P})$ such that
$$\Sigma_1^b(\mathfrak{P}) - \text{PIND} \vdash \varphi \iff \varphi^*.$$ 

Theorem 10. ($\Sigma_1^b$-replacement)

For a $\Sigma_1^b$-formula $\varphi$ and a term $s_1$ in $\mathfrak{L}_{BA}$, there is a $\Sigma_1^b$-formula $\psi$ and a term $s_2$ s.t.
$$S_1^b \vdash (\forall x \leq |t|)(\exists y \leq s_1)\varphi(x, y) \iff (\exists y \leq s_2)(\forall x \leq |t|)\psi(x, y).$$

Proof. This is Theorem 14 in [1]. The basic idea is to code the sequence of $y$’s for each $x$ into $w$.

$$S_1^b \vdash (\forall x \leq |t|)(\exists y \leq s_1)\varphi(x, y) \iff (\exists w \leq SqBd(s_1, t))(\forall x \leq |t|)[\varphi(x, \beta(x', w)) \land \beta(x', w) \leq s_1].$$

The above theorem enables us to “push” the sharply bounded quantification into the scope of bounded quantifier.

Definition 7. The collection of strict $\Sigma_1^b$-formulas in a language $\mathfrak{L}$, denoted by $s-\Sigma_1^b(\mathfrak{L})$, is the smallest set of formulas which begin with exactly one bounded existential quantifier, i.e. $\exists x \leq t$, followed by an open formula.

Corollary 11. For any formula $\varphi \in \Sigma_1^b(\mathfrak{P})$, there is a formula $\varphi^* \in s-\Sigma_1^b(\mathfrak{P})$ s.t.
$$\Sigma_1^b(\mathfrak{P}) - \text{PIND} \vdash \varphi \iff \varphi^*.$$ 

Proof. Pushing any sharply bounded quantifier inside the scope of a bounded quantifier by $\Sigma_1^b$-replacement (theorem (10)), combining any two bounded existential quantifiers into one by pairing function (definable in $S_2^b$) and replacing the sharply-bounded formula with its equivalence by corollary (9) gives the desired $s-\Sigma_1^b$ formula.

Theorem 12. $\Sigma_1^b-\text{DF}(\Sigma_1^b(\mathfrak{P}) - \text{PIND}) = QF-\text{DF}(QF(\mathfrak{P}) - \text{PIND})$

Proof. The $\forall$–inversion holds as none of the axioms is specified using $\forall$. Together with corollary (11), it is suffice to show that $s-\Sigma_1^b(\mathfrak{P}) - \text{PIND}$ is conservative over $QF(\mathfrak{P}) - \text{PIND}$ for $s-\Sigma_1^b$–formula. We examine a normal derivation of $\Gamma$ which contains no universal quantification. Obvious lines in the following derivations are skipped and we may optionally illustrate the skipped part/inference by $\vdash$ and double lines.

Consider the first application of $(s-\Sigma_1^b-\text{PIND})$ with p.f. $\exists y[y \leq t(s) \land \varphi(s, y)]$ where $\varphi$ is an open formula. Let $\psi(x, y) \overset{df}{=} y \leq t(x) \land \varphi(x, y)$. The instance is as follows:

$$\Delta, \exists y[\psi(0, y)] \quad \Delta, \neg \exists y[\psi(\lfloor t \rfloor, y)], \exists y[\psi(b, y)]$$

$$\Delta, \exists y[\psi(s, y)]$$
Because $\Gamma$ is purely existential and the derivation is normal, $\Delta$ must also be purely existential. Inverting the $\forall$-quantifier in $\neg \exists y[\psi([\frac{b}{2}], y)]$, we obtain an eigenvariable $c$ which does not occur in $\Delta$. By Theorem (7), there are $f_0, f_1 \in \mathcal{P}$ s.t.

$$QF(\mathcal{P}) - PIND \vdash \Delta, \psi(0, f_0(0))$$
$$QF(\mathcal{P}) - PIND \vdash \Delta, \neg \psi([\frac{b}{2}], c), \psi(b, f_1(b, c))$$

Now we define

$$f(0) = f_0(0)$$
$$f(x') = f_1(x', f([\frac{b}{2}]))$$

Function defined in this manner belongs to $\mathcal{P}$ and will be shown later in the proof. We choose this form to facilitate our argument for conservativeness.

A formal way to perform substitution in derivation is by using equality axioms and cuts. Substituting $f([\frac{b}{2}])$ for $c$, $f$ for $f_1$ when $b \neq 0$ (i.e. $b = x'$), we have the following derivation in $QF(\mathcal{P}) - PIND$:

$$\vdots$$
$$\neg \psi(0, f(0)), \psi(0, f(0))$$
$$\quad b \neq 0, \neg \psi([\frac{b}{2}], f([\frac{b}{2}])), \psi(b, f(b))$$
$$\Delta, \neg \psi([\frac{b}{2}], c), \psi(b, f_1(b, c))$$
$$\quad b = 0, \neg \psi([\frac{b}{2}], f([\frac{b}{2}])), \psi(b, f(b))$$
$$\quad (C)$$
$$\neg \psi([\frac{b}{2}], f([\frac{b}{2}])), \psi(b, f(b))$$

With this, we can derive the same conclusion of $s-\Sigma^b_1 - PIND$ as follows:

$$\vdots$$
$$\quad \Delta, \psi(0, f(0))$$
$$\Delta, \neg \psi([\frac{b}{2}], f([\frac{b}{2}])), \psi(b, f(b))^{(QF - PIND)}$$
$$\Delta, \neg \psi(s, f(s))$$
$$\Delta, \exists y[y \leq t(s) \land \varphi(s, y)]^{(3)}$$

We need to show $f \in \mathcal{P}$ to complete the proof. Let’s consider the function $frontbits(x, y)$ which gives the number with binary representation identical to the leading $y$ bits of $x$. We skip the limited iteration definition but it is not difficult to see $frontbits(x, y) \in \mathcal{P}$. Then $f_2(b, c, d) \equiv f_1(frontbits(b, c), d)$ belongs to $\mathcal{P}$ as $\mathcal{P}$ is closed under composition. The function $f$ can be defined by limited iteration from $f_0$ and $f_2$ as follows:

$$\tau(b, 0) = f_0(0)$$
$$\tau(b, c') = f_2(b, c, \tau(b, c))$$
$$f(b) = \tau(b, |b|)$$

where $p(|b|) = |b|$ and $q(|b|) = t(b)$. Thus, $f \in \mathcal{P}$.

Hence, we can reduce the number of $s-\Sigma^b_1 - PIND$ application by one. By induction, the conservative property is shown.
6 Generalization: $S_2^i$ and $\Sigma_i^P$

Definition 8.

- A function $f$ is a predicate if its range is $\{0, 1\}$.
- For a collection of functions $\mathcal{F}$, $PRED(\mathcal{F})$ is the collection of predicates in $\mathcal{F}$.
- For two functions $f$ and $g$, we define
  $$d_f(\bar{x}) = \begin{cases} 
1 & \text{if } (\exists y \leq f(\bar{x})) [g(\bar{x}, y) > 0] \\
0 & \text{otherwise}
\end{cases}$$
  $$d_f(\bar{x}) = \begin{cases} 
1 & \text{if } (\forall y \leq f(\bar{x})) [g(\bar{x}, y) > 0] \\
0 & \text{otherwise}
\end{cases}$$

- For a collection of functions $\mathcal{F}$ which is closed under composition:
  $$PB\exists(\mathcal{F}) = \{\exists s.t. f(\bar{x}) \in \mathcal{F} \mid R(\bar{x}, y) \in PRED(\mathcal{F})\}$$
  $$PB\forall(\mathcal{F}) = \{\forall s.t. f(\bar{x}) \in \mathcal{F} \mid R(\bar{x}, y) \in PRED(\mathcal{F})\}$$

where $p$ is a suitable polynomial.

Definition 9. With these, we can define the hierarchy of predicates which corresponds to polynomial hierarchy:

$$\Delta_i^p \overset{df}{=} PRED(\Delta_i^p).$$
$$\Sigma_i^p \overset{df}{=} PB\exists(\Delta_i^p).$$
$$\Pi_i^p \overset{df}{=} PB\forall(\Delta_i^p).$$
$$D_{i+1}^p \overset{df}{=} Cl(\Sigma_i^p).$$
$$PH \overset{df}{=} \bigcup_{k \in \mathbb{N}} \Sigma_k^p.$$  

$\Delta_i^p$, $\Sigma_i^p$ and $\Pi_i^p$ are essentially the computational complexity classes $P$, $NP$ and $co-NP$ respectively.

Theorem 13. $\forall_i^p \subseteq \Sigma_i^{p-DF}(S_2^i)$

Proof. The base case is Theorem (3). Here we will show the case for $i + 1$. Modify the proof for theorem (3), we only need to include the case when a function $f \in D_{i+1}^p$ is defined from $g \in \Sigma_i^{p-DF}(S_2^i)$ by $PB\exists$ s.t.

$$f(\bar{x}) = \begin{cases} 
1 & \text{if } (\exists y \leq 2^{p(|\bar{x}|)}) [g(\bar{x}, y) > 0] \\
0 & \text{otherwise}
\end{cases}$$
and the defining formula for $g$ is $\varphi_g(\vec{x}, y, z) \in \Sigma^b_1$ s.t. $(g(\vec{x}, y) = z) \iff \varphi_g(\vec{x}, y, z)$. Consider the following formula $\varphi_f \equiv$

$\left( (u = 1) \land (\exists y \leq 2^{p(\norm{\vec{x}})}) [(z > 0) \land \varphi_g(\vec{x}, y, z)] \right) \lor \left( (u = 0) \land (\forall y \leq 2^{p(\norm{\vec{x}})}) [(z \neq 0) \lor \lnot \varphi_g(\vec{x}, y, z)] \right)$

It is obvious that $(f(\vec{x}) = u) \iff \varphi_f(\vec{x}, u)$ and $\varphi_f \in \Sigma^b_{i+1}$.

**Theorem 14.** $\Sigma^b_i$-$DF(S^2_i) \subseteq \Pi^p_i$

**Proof.** By extending the language $\Sigma_{BA}$ with symbols for functions in $\Pi^p_i$. It can be shown that for every formula in $\Sigma_i^b(\Sigma_{BA})$, there is an equivalence in $\Pi_i^b$-formula in the extended language, provable in the extended theory as a form of Skolemization or operator theory in [3]. The form of defining formulas for functions in $\Pi^p_i$ is carefully stated to preserve $\land$, $\lor$, and $\forall$-inversions as well as Theorem (7), i.e. we will avoid any use of quantification or logical connectives. For example, to define a $\Sigma^b_1$-Skolem function $f$ for an open formula $\varphi$ s.t.

$(\exists y \leq t(\vec{x})) \varphi(\vec{x}, y) \iff \varphi(\vec{x}, f(\vec{x}))$,

the axioms are given as two sets of formula

$\{\lnot y \leq t(\vec{x}), \lnot \varphi(\vec{x}, y), \varphi(\vec{x}, f(\vec{x}))\}, \{f(\vec{x}) \leq t(\vec{x})\}$.

$\Sigma^b_{i+1}$-Skolem functions are defined with open formulas which consist of $\Pi^p_i$ function symbols, instead of using $\Sigma^b_i$-formulas directly. With these measures, the argument follows the proof of Theorem (12).

**References**

