Pumping by Typing

Naoki Kobayashi
The University of Tokyo

***Abstract*—Higher-order recursion schemes (HORS), which are higher-order grammars for generating infinite trees, have recently been studied extensively in the context of model checking and its applications to higher-order program verification. We develop a pumping lemma for HORS by using a novel but simple intersection type system for reasoning about reductions of λ-terms. Our proof is arguably much simpler than the proof of Kartzow and Parys' pumping lemma for collapsible pushdown automata. As an application, we give an alternative proof of Kartzow and Parys' result about the strictness of the hierarchy of trees generated by HORS.**

I. Introduction

Higher-order grammars, where non-terminal symbols can take higher-order functions as parameters, were actively studied during 70–80's [9], [30], [27], and interest has recently been revived in the context of model checking [15], [22]. Model checking the tree structures generated by higher-order grammars (known as *higher-order recursion schemes* (HORS) [15], [22] in this context) is a natural generalization of finite-state and pushdown model checking, and has recently been applied to automated verification of higher-order programs [16], [19], [23].

In the present paper, we are interested in developing a pumping lemma for HORS. Pumping lemmas [1], [11] state properties about certain "repeated" structures of words or trees generated by a grammar, and constitute a fundamental tool for the study of formal languages; for example, pumping lemmas are used for showing that a certain language does not belong to a certain class of languages. Pumping lemmas may also be useful for the optimization of model checking algorithms for HORS [16], [17], [21], by detecting repeated structures and avoiding redundant inspection of them.

Pumping lemmas for regular or context-free languages are well-known, and can be easily understood. If there is a sufficiently long derivation sequence $S \longrightarrow^* z$ of a word z then it must contain repeated occurrences of a non-terminal symbol, like: $S \longrightarrow^* uAy \longrightarrow^* uvAwy \longrightarrow^* uvxwy = z$; thus, $uv^n xw^n y$ can be generated for any $n\,(\geq 0)$, by $S \longrightarrow^* uAy \longrightarrow^* uvAwy \longrightarrow^* \cdots \longrightarrow^* uv^n Aw^n y \longrightarrow^* uv^n xw^n y$. For indexed languages (which correspond to order-2 grammars), however, the proof of Hayashi's pumping lemma is quite involved [11]. For grammars of order-3 or more, pumping lemmas have not been studied until recently to our knowledge. Recently, Kartzow and Parys [13] obtained a pumping lemma for collapsible (higher-order) pushdown automata (CPDA), which are equivalent to HORS as tree generators, and used it to show that the hierarchy of trees generated by HORS is strict: there is a tree that can be generated by an order-$(n+1)$ HORS but cannot be generated by any order-n HORS. Their proof is, however, long and hard to understand at least for non-experts on CPDA. When viewed as a pumping lemma for HORS, it is also disappointing that one has to make a detour to CPDA; although CPDA and HORS are known to be equivalent (as tree generators), translations between them are rather complex [10], [7]. This is a sharp contrast with the situation for regular and context-free languages, for which pumping lemmas are obtained directly by reasoning about grammars, rather than pushdown automata.

Motivated by the observation above, we develop a pumping lemma for HORS by direct reasoning about grammars, without a detour to CPDA. It turned out that such a pumping lemma can be easily obtained by combining a novel but rather simple intersection type system with standard properties of the λ-calculus. The overall proof of the pumping lemma is (arguably) simpler than that of Kartzow and Parys' pumping lemma for CPDA. The obtained pumping lemma is actually not as strong as those for context-free and indexed languages [11], but it is strong enough to reprove Kartzow and Parys's result that the hierarchy of the classes of trees generated HORS is strict.

Section II reviews HORS, states our pumping lemma, and applies it to show the strictness of the hierarchy of the trees generated by HORS. Section III studies properties of the λ-calculus, and develops a new intersection type system for reasoning about reductions of λ-terms. By using those techniques, Section IV proves the pumping lemma. Section V discusses related work and Section VI concludes the paper.

II. HORS and Pumping Lemma

A. HORS

A higher-order recursion scheme (HORS) is essentially a term of the simply-typed λ-calculus with recursion and tree constructors for generating a (possibly infinite) tree. We assume a ranked alphabet Σ, which maps a finite set of symbols (ranged over by a) to their arities. An element of Σ of arity n is used as an n-ary tree constructor below. For the sake of simplicity, we assume that there is exactly one symbol of arity 0, and write $\mathtt{e}$ for it.[1]

The sets of *λ-terms* and *sorts*[2] are given by:

$$\begin{array}{rcl} t\ (\lambda\text{-terms}) & ::= & a \mid x \mid t_0 t_1 \mid \lambda x:\kappa.t \\ \kappa\ (\text{sorts}) & ::= & \mathtt{o} \mid \kappa_1 \to \kappa_2. \end{array}$$

[1] This does not lose generality, since any other symbol a of arity 0 can be expressed by $a'(\mathtt{e})$, where a' is a new symbol of arity 1.

[2] We use the term "sorts" instead of "types" to avoid confusion with intersection types introduced later.

We write $\mathbf{FV}(t)$ for the set of free variables in t. As usual, we identify λ-terms up to α-conversion. We call a term t_0 the head of a term t if t is of the form $t_0\,t_1\,\cdots\,t_m$ and t_0 is not an application (of the form $t_{00}t_{01}$). The sort $\mathtt{o}$ describes trees, and the sort $\kappa_1 \to \kappa_2$ describes functions from κ_1 to κ_2. The order of sort κ, written $\mathtt{ord}(\kappa)$, is defined by

$$\mathtt{ord}(\mathtt{o}) = 0 \quad \mathtt{ord}(\kappa_1 \to \kappa_2) = \max(\mathtt{ord}(\kappa_1)+1, \mathtt{ord}(\kappa_2))$$

We often omit sorts and just write $\lambda x.t$ for $\lambda x\!:\!\kappa.t$, but please keep in mind that every variable is implicitly sorted.

We use the following simple type system to assign a sort to a term.

$$\frac{\Sigma(a) = n}{\mathcal{K} \vdash_\Sigma a : \underbrace{\mathtt{o} \to \cdots \to \mathtt{o}}_{n} \to \mathtt{o}} \qquad \frac{}{\mathcal{K}, x : \kappa \vdash_\Sigma x : \kappa}$$

$$\frac{\begin{array}{c}\mathcal{K} \vdash_\Sigma t_0 : \kappa_1 \to \kappa \\ \mathcal{K} \vdash t_1 : \kappa_1\end{array}}{\mathcal{K} \vdash t_0 t_1 : \kappa} \qquad \frac{\mathcal{K}, x : \kappa_1 \vdash_\Sigma t : \kappa_2}{\mathcal{K} \vdash_\Sigma \lambda x : \kappa_1.t : \kappa_1 \to \kappa_2}$$

Here $\mathcal{K}$, called a sort assignment, maps variables to sorts.

For a term t such that $\mathcal{K} \vdash_\Sigma t : \kappa$, the order of t (with respect to $\mathcal{K}$), written $\mathtt{ord}_{\mathcal{K}}(t)$ (or simply $\mathtt{ord}(t)$), is the largest order of sorts occurring in $\mathcal{K}$, t, and κ. (Here, note that the sort κ of t is determined uniquely by $\mathcal{K}$ and t.) We write $\mathit{dom}(f)$ for the domain of a map f below.

Definition 2.1: A *higher-order recursion scheme* (HORS, for short) $\mathcal{G}$ is a quadruple $(\Sigma, \mathcal{N}, \mathcal{R}, S)$, where Σ is a ranked alphabet, $\mathcal{N}$ is a map from variables (called non-terminals) to their sorts, $\mathcal{R}$ is a map from non-terminals to λ-terms in β-normal form (where non-terminals are treated as variables), and $S (\in \mathit{dom}(\mathcal{N}))$ is a special non-terminal called a *start symbol*. We require that $\mathcal{N} \vdash_\Sigma \mathcal{R}(F) : \mathcal{N}(F)$ for each $F \in \mathit{dom}(\mathcal{N})$, and $\mathcal{N}(S) = \mathtt{o}$. The *order* of HORS $\mathcal{G}$, written $\mathtt{ord}(\mathcal{G})$, is the largest order of the sorts of non-terminals, i.e., $\max(\{\mathtt{ord}(\mathcal{N}(F)) \mid F \in \mathit{dom}(\mathcal{N})\})$.[3]

We often write $\Sigma_\mathcal{G}, \mathcal{N}_\mathcal{G}, \mathcal{R}_\mathcal{G}, S_\mathcal{G}$ for the four elements of HORS $\mathcal{G}$. Intuitively, a HORS $(\Sigma, \mathcal{N}, \mathcal{R}, S)$ with $\mathcal{R} = \{F_1 \mapsto t_1, \ldots, F_n \mapsto t_n\}$ can be understood as mutually recursive function definitions $F_1 = t_1, \ldots, F_n = t_n$ with $S \in \{F_1, \ldots, F_n\}$ being the "main" function.

We define a labeled transition relation $t \xrightarrow{\alpha}_\mathcal{G} t'$ for HORS (where α is ϵ or of the form (a, i)), following Carayol and Serre's labeled recursion schemes [7].

$$F\,s_1\,\cdots\,s_n \xrightarrow{\epsilon}_\mathcal{G} \mathcal{R}_\mathcal{G}(F)\,s_1\,\cdots\,s_n \qquad \text{(R-NT)}$$

$$(\lambda x.t)s_1\,\cdots\,s_n \xrightarrow{\epsilon}_\mathcal{G} ([s_1/x]t)\,s_2\,\cdots\,s_n \quad \text{(R-BETA)}$$

$$\frac{\Sigma(a) = n \qquad 1 \le i \le n}{a\,t_1\,\cdots\,t_n \xrightarrow{(a,i)}_\mathcal{G} t_i} \qquad \text{(R-CONST)}$$

Here $[s/x]t$ denotes the term obtained from t by replacing all the free occurrences of x with s. We write $t \overset{\alpha_1\cdots\alpha_n}{\Longrightarrow} t'$ if $t \xrightarrow{\alpha_1} t_1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_n} t'$ (where ϵ is treated as an empty sequence). For example, $\mathtt{a}((\lambda x.\mathtt{b}\,\mathtt{e})y) \overset{(\mathtt{a},1)(\mathtt{b},1)}{\Longrightarrow}_\mathcal{G} \mathtt{e}$. We sometimes omit the subscript $\mathcal{G}$ or drop the second component of a label; so, we may write $\mathtt{a}((\lambda x.\mathtt{b}\,\mathtt{e})y) \overset{(\mathtt{a},1)(\mathtt{b},1)}{\Longrightarrow} \mathtt{e}$ or $\mathtt{a}((\lambda x.\mathtt{b}\,\mathtt{e})y) \overset{\mathtt{ab}}{\Longrightarrow} \mathtt{e}$.

The labels of a reduction sequence correspond to a path of the tree generated by HORS [22]. A Σ-labeled (ranked) tree T is a partial map from $\{1, \ldots, m\}^*$ (where m is the largest arity of symbols in Σ) to $\mathit{dom}(\Sigma)$ such that $\mathit{dom}(T)$ is prefix-closed and $\forall \pi \in \mathit{dom}(T).\{i \mid \pi i \in \mathit{dom}(T)\} = \{i \mid 1 \le i \le \Sigma(T(\pi))\}$. We define the $(\Sigma \cup \{\bot \mapsto 0\})$-labeled tree generated by HORS, written $\mathbf{Tree}(\mathcal{G})$, by:

$$\begin{array}{l}\mathbf{Tree}(\mathcal{G}) = \\ \qquad \{i_1 \cdots i_{n-1} \mapsto a_n \mid S_\mathcal{G} \overset{(a_1,i_1)\cdots(a_{n-1},i_{n-1})(a_n,i_n)}{\Longrightarrow}_\mathcal{G} t\} \\ \qquad \cup\{i_1 \cdots i_n \mapsto \mathtt{e} \mid S_\mathcal{G} \overset{(a_1,i_1)\cdots(a_n,i_n)}{\Longrightarrow}_\mathcal{G} \mathtt{e}\} \\ \qquad \cup\{i_1 \cdots i_n \mapsto \bot \mid S_\mathcal{G} \overset{(a_1,i_1)\cdots(a_n,i_n)}{\Longrightarrow}_\mathcal{G} t \\ \qquad\qquad \text{but } t \ne \mathtt{e} \text{ and there is no } t' \text{ such that } t \overset{(a,i)}{\Longrightarrow} t' \ \}\end{array}$$

This coincides with the usual definition of the tree generated by $\mathcal{G}$ [22] and that of the Böhm tree [4] of the λ-term corresponding to $\mathcal{G}$ (where recursion is expressed by a fixedpoint combinator).

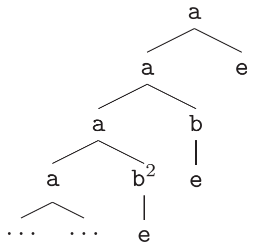
Example 2.1: Consider HORS $\mathcal{G}_1 = (\mathcal{N}, \Sigma, \mathcal{R}, S)$ where:

$$\begin{array}{l}\mathcal{N} = \{S : \mathtt{o}, F : \mathtt{o} \to \mathtt{o}\} \\ \Sigma = \{\mathtt{a} : \mathtt{o} \to \mathtt{o} \to \mathtt{o}, \mathtt{b} : \mathtt{o} \to \mathtt{o}, \mathtt{e} : \mathtt{o}\} \\ \mathcal{R} = \{S \mapsto F\,\mathtt{e}, \quad F \mapsto \lambda x.\mathtt{a}\,(F\,(\mathtt{b}\,x))\,x\}.\end{array}$$

The start symbol S has the reduction sequence:

$$S \overset{(\mathtt{a},1)^m(\mathtt{a},2)}{\Longrightarrow}_\mathcal{G} (\mathtt{b}^m\,\mathtt{e}) \overset{(\mathtt{b},1)^m}{\Longrightarrow}_\mathcal{G} \mathtt{e}$$

for each $m \ge 0$. Here, α^m stands for m repetitions of α. Thus $\mathbf{Tree}(\mathcal{G}_1)$ is:



Example 2.2: Consider HORS $\mathcal{G}_2 = (\mathcal{N}, \Sigma, \mathcal{R}, S)$ where:

$$\begin{array}{l}\mathcal{N} = \{S : \mathtt{o}, F : (\mathtt{o} \to \mathtt{o}) \to \mathtt{o}, \mathit{Two} : (\mathtt{o} \to \mathtt{o}) \to (\mathtt{o} \to \mathtt{o})\} \\ \Sigma = \{\mathtt{a} : \mathtt{o} \to \mathtt{o} \to \mathtt{o}, \mathtt{b} : \mathtt{o} \to \mathtt{o}, \mathtt{e} : \mathtt{o}\} \\ \mathcal{R} = \{S \mapsto F\,\mathtt{b}, \quad \mathit{Two} \mapsto \lambda f.\lambda x.f\,(f\,x), \\ \qquad F \mapsto \lambda f.\mathtt{a}\,(F\,(\mathit{Two}\,f))\,(f\,\mathtt{e})\}\end{array}$$

The non-terminal S has the following reduction sequences.

$$\begin{array}{l}S \xrightarrow{\epsilon}_\mathcal{G} F\,\mathtt{b} \overset{\epsilon}{\Longrightarrow}_\mathcal{G} \mathtt{a}\,(F\,(\mathit{Two}\,\mathtt{b}))\,(\mathtt{b}\,\mathtt{e}) \xrightarrow{(\mathtt{a},2)}_\mathcal{G} \mathtt{b}\,\mathtt{e} \overset{(\mathtt{b},1)}{\Longrightarrow}_\mathcal{G} \mathtt{e} \\ S \overset{(\mathtt{a},1)^m(\mathtt{a},2)}{\Longrightarrow}_\mathcal{G} (\mathit{Two}^m\,\mathtt{b}\,\mathtt{e}) \overset{(\mathtt{b},1)^{2^m}}{\Longrightarrow}_\mathcal{G} \mathtt{e} \text{ (for each } m \ge 1)\end{array}$$

Thus $\mathbf{Tree}(\mathcal{G}_2)$ consists of paths labeled by $\mathtt{a}^{m+1}\mathtt{b}^{2^m}\mathtt{e}$. □

Remark 2.1: A HORS can also be regarded as a generator of word or tree languages, by interpreting a special terminal symbol $\mathtt{br}$ of arity 2 as a non-deterministic choice. For

[3] By the assumption that $\mathcal{R}(F)$ is in β-normal form, it is equivalent to $\max(\{\mathtt{ord}_\mathcal{N}(\mathcal{R}(F)) \mid F \in \mathit{dom}(\mathcal{N})\})$.

example, the word language defined by $\mathcal{G}$ can be defined as: $\{a_1 \cdots a_n{\uparrow_{\mathtt{br}}} \mid S_{\mathcal{G}} \xRightarrow{(a_1,i_1)\cdots(a_n,i_n)}_{\mathcal{G}} \mathtt{e}\}$, where $w\uparrow_{\mathtt{br}}$ denotes the word obtained from w by removing $\mathtt{br}$. □

B. Pumping Lemma

We write $\widetilde{s}$ for a sequence $s_1 \cdots s_k$ of terms, and write $|w|$ for the length of a sequence w. We define $\mathbf{exp}_n(x)$ by $\mathbf{exp}_0(x) = x$ and $\mathbf{exp}_{n+1}(x) = 2^{\mathbf{exp}_n(x)}$.

Our pumping lemma for HORS is stated as follows.

Theorem 2.1 (Pumping):
Let $\mathcal{G}$ be an order-n HORS $(\Sigma, \mathcal{N}, \mathcal{R}, S)$ with $n \geq 1$. Then there exist constants c_1 and c_2 (that depend on $\mathcal{G}$) such that if $S \xRightarrow{w}_{\mathcal{R}} \mathtt{e}$ and $|w| > \mathbf{exp}_{n-1}(c_1)$, then there exist $F \in \mathit{dom}(\mathcal{N})$, $\{\widetilde{s_i} \mid i \geq 1\}$, $\{w_i \mid i \geq 1\}$, and $\{v_i \mid i \geq 1\}$ that satisfy the following conditions.

- $S \xRightarrow{w_1}_{\mathcal{G}} F\,\widetilde{s_1} \xRightarrow{w_2}_{\mathcal{G}} F\,\widetilde{s_2} \xRightarrow{w_3}_{\mathcal{G}} \cdots \xRightarrow{w_m}_{\mathcal{G}} F\,\widetilde{s_m} \xRightarrow{v_m}_{\mathcal{G}} \mathtt{e}$ for every $m \geq 1$;
- $|w_1 \cdots w_m v_m| \leq \mathbf{exp}_{n-1}((m+1)c_1^2)$; and
- $w_1 \cdots w_{m_1} v_{m_1} \neq w_1 \cdots w_{m_2} v_{m_2}$ for every m_1, m_2 such that $m_2 \geq m_1 + c_2$.

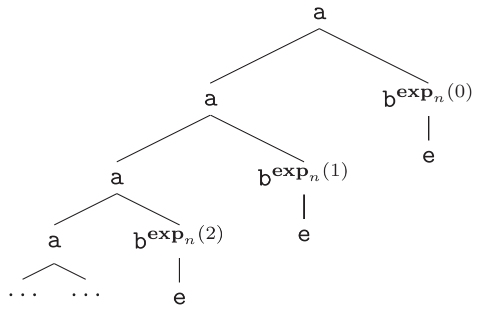
The theorem states that if $\mathcal{G}$ has a sufficiently long reduction sequence, then one can construct a series of reduction sequences in which the same non-terminal (F above) may occur arbitrarily many times in the head position, and the length of the m-th reduction sequence is bounded by $(n-1)$-fold exponential of $m+1$. Proving this theorem is the goal of Sections III and IV.

Example 2.3: Recall the order-2 HORS $\mathcal{G}_2$ from Example 2.2. Let $c_1 = c_2 = 1$. Then, the required conditions hold for $w_1 = \epsilon$, $w_i = (\mathtt{a}, 1)$ (for $i \geq 2$), and $v_i = (\mathtt{a}, 2)(\mathtt{b}, 1)^{2^{i-1}}$.

C. Application

As an application of the pumping lemma, we give an alternative proof of Kartzow and Parys' result that there is a tree that can be generated by an order-$(n+1)$ HORS, but cannot be generated by any order-n HORS [13].

Consider the following tree $\mathcal{T}_n$, which has the paths of the form $\mathtt{a}^{m+1}\mathtt{b}^{\mathbf{exp}_n(m)}\mathtt{e}$ for $m \geq 0$.


a
a
$\mathtt{b}^{\mathbf{exp}_n(0)}$
e
a
$\mathtt{b}^{\mathbf{exp}_n(1)}$
e
a
$\mathtt{b}^{\mathbf{exp}_n(2)}$
e
... ...

This is the same as the tree used by Kartzow and Parys [13].

The trees $\mathcal{T}_0$ and $\mathcal{T}_1$ are generated by $\mathcal{G}_1$ and $\mathcal{G}_2$ in Examples 2.1 and 2.2 respectively. For $n \geq 2$, $\mathcal{T}_n$ can be generated by the order-$(n+1)$ HORS $(\mathcal{N}, \Sigma, \mathcal{R}, S)$ where:

$$\begin{aligned}
&\mathcal{N} = \{S : \mathtt{o}, F : \kappa_n \to \mathtt{o}\} \cup \{\mathit{Two}_k : \kappa_k \to \kappa_k \mid 1 \leq k \leq n\} \\
&\qquad (\kappa_0 = \mathtt{o}, \kappa_k = \kappa_{k-1} \to \kappa_{k-1} \text{ for } 1 \leq k \leq n) \\
&\Sigma = \{\mathtt{a} : \mathtt{o} \to \mathtt{o} \to \mathtt{o}, \mathtt{b} : \mathtt{o} \to \mathtt{o}, \mathtt{e} : \mathtt{o}\} \\
&\mathcal{R} = \{S \mapsto F\ \mathit{Two}_{n-1}, \\
&\qquad F \mapsto \lambda f.\mathtt{a}\,(F\,(\mathit{Two}_n f))\,(f\ \mathit{Two}_{n-2} \cdots \mathit{Two}_1\,\mathtt{b}\,\mathtt{e})\} \\
&\qquad \cup \{\mathit{Two}_k \mapsto \lambda f : \kappa_k.\lambda x : \kappa_{k-1}.f(f(x)) \mid 1 \leq k \leq n\}
\end{aligned}$$

Note that we have:

$$S \xRightarrow{(\mathtt{a},1)^m(\mathtt{a},2)} \mathit{Two}_n^m\ \mathit{Two}_{n-1} \cdots \mathit{Two}_1\,\mathtt{b}\,\mathtt{e} \xRightarrow{(\mathtt{b},1)^{\mathbf{exp}_n(m)}} \mathtt{e}.$$

Theorem 2.2: There is no order-n HORS that generates $\mathcal{T}_n$.

Proof: If $n = 0$, then the result follows from the fact that an order-0 HORS can generate only a regular tree, but $\mathcal{T}_0$ is non-regular. For $n \geq 1$, the proof is by contradiction. Suppose that there is an order-n HORS $\mathcal{G}$ that generates $\mathcal{T}_n$. Since $S \xRightarrow{(\mathtt{a},1)^k(\mathtt{a},2)\mathtt{b}^{\mathbf{exp}_n(k)}}_{\mathcal{G}} \mathtt{e}$ for every $k \geq 0$, we can apply Pumping Lemma (Theorem 2.1) to get words $\{w_i \mid i \geq 1\}$ and $\{v_i \mid i \geq 1\}$ such that:

(i) $S \xRightarrow{w_1 \cdots w_m v_m}_{\mathcal{G}} \mathtt{e}$ for every $m \geq 1$;
(ii) $|w_1 \cdots w_m v_m| \leq \mathbf{exp}_{n-1}((m+1)c_1^2)$; and
(iii) $w_1 \cdots w_{m_1} v_{m_1} \neq w_1 \cdots w_{m_2} v_{m_2}$ for every m_1, m_2 such that $m_2 \geq m_1 + c_2$.

Let $u_m = w_1 \cdots w_m v_m$. Pick an arbitrary number $m' \geq 1$, and consider the words: $u_1, u_{c_2+1}, \ldots, u_{m'c_2+1}$. By condition (iii), all the words are different from each other. Since the $(m'+1)$-th shortest path of $\mathcal{T}_n$ (leading to $\mathtt{e}$) is labeled by $\mathtt{a}^{m'+1}\mathtt{b}^{\mathbf{exp}_n(m')}$, by condition (ii) above, we have:

$$\begin{aligned}
\mathbf{exp}_{n-1}((m'c_2+2)c_1^2) &\geq \max(|u_1|, |u_{c_2+1}|, \ldots, |u_{m'c_2+1}|) \\
&\geq |\mathtt{a}^{m'+1}\mathtt{b}^{\mathbf{exp}_n(m')}| = m' + 1 + \mathbf{exp}_n(m')
\end{aligned}$$

for every $m' \geq 1$. This does not hold, however, for a sufficiently large m' such that $(m'c_2+2)c_1^2 < 2^{m'}$, hence a contradiction. ■

Corollary 2.3:
The hierarchy of the trees generated by HORS is strict, i.e., $\{\mathbf{Tree}(\mathcal{G}) \mid \mathtt{ord}(\mathcal{G}) = n\} \subsetneq \{\mathbf{Tree}(\mathcal{G}) \mid \mathtt{ord}(\mathcal{G}) = n+1\}$ for all $n (\geq 0)$.

Remark 2.2: Theorem 2.1 is not strong enough to show that the *word language* (as defined in Remark 2.1) $\{\mathtt{a}^{m+1}\mathtt{b}^{\mathbf{exp}_n(m)} \mid m \geq 0\}$ cannot be generated by any order-n HORS. Note that in the third condition of the theorem, even if $w_1 \cdots w_{m_1} v_{m_1} \neq w_1 \cdots w_{m_2} v_{m_2}$ holds, it may be the case that $(w_1 \cdots w_{m_1} v_{m_1}){\uparrow_{\mathtt{br}}} = (w_1 \cdots w_{m_2} v_{m_2}){\uparrow_{\mathtt{br}}}$. □

III. Properties of the simply-typed λ-calculus

In this section we prepare properties of λ-terms required for proving Theorem 2.1. Before doing so, we explain our proof strategy for the theorem. The basic idea is the same as the argument for context-free languages: in a sufficiently long reduction sequence $S \xRightarrow{w}_{\mathcal{G}} \mathtt{e}$, there must be repeated occurrences of the same non-terminal in the head position:

$$S \xRightarrow{w_1}_{\mathcal{G}} F\,\widetilde{s} \xRightarrow{w_2}_{\mathcal{G}} F\,\widetilde{t} \xRightarrow{v_2}_{\mathcal{G}} \mathtt{e}$$

and the part $F\,\widetilde{s} \overset{w_2}{\Longrightarrow}_{\mathcal{G}} F\,\widetilde{t}$ can be repeated (or "pumped"). Obviously, not all such sequences can be pumped, however. For example, consider the rewriting rules

$$\{S \mapsto F(F\,\mathtt{e}), F \mapsto \lambda x.x\}.$$

Then

$$S \xrightarrow{\epsilon} F(F\,\mathtt{e}) \overset{\epsilon}{\Longrightarrow} F\,\mathtt{e} \xrightarrow{\epsilon} \mathtt{e},$$

but the part $F(F\,\mathtt{e}) \overset{\epsilon}{\Longrightarrow} F\,\mathtt{e}$ cannot be repeated. This is due to the fact that the third term $F\,\mathtt{e}$ comes from the argument position of the second term; thus, F may not be obtained again by unfolding F. This observation leads to the requirement that the second F in $F\,\widetilde{s} \overset{w_2}{\Longrightarrow}_{\mathcal{G}} F\,\widetilde{t}$ must have been obtained by unfolding the first F, not from the argument $\widetilde{s}$. This requirement is still insufficient. For example, consider

$$\{S \mapsto F(\lambda x.x), F \mapsto \lambda f.f(F(\lambda x.\mathtt{e}))\}.$$

We have:

$$S \xrightarrow{\epsilon} F(\lambda x.x) \overset{\epsilon}{\Longrightarrow} (\lambda x.x)(F(\lambda x.\mathtt{e})) \xrightarrow{\epsilon} F(\lambda x.\mathtt{e}) \overset{\epsilon}{\Longrightarrow} \mathtt{e},$$

but the part $F(\lambda x.x) \overset{\epsilon}{\Longrightarrow} F(\lambda x.\mathtt{e})$ cannot be repeated. This can be attributed to the fact that the reduction behaviors of the arguments $\lambda x.x$ and $\lambda x.\mathtt{e}$ are different: the former uses the argument x, while the latter ignores it. In Section III-B, we shall develop an intersection type system that captures this kind of behavior of λ-terms. By using it, we can ensure that if $F\,\widetilde{s} \overset{w_2}{\Longrightarrow}_{\mathcal{G}} F\,\widetilde{t}$ (where the second F must come from the first F) *and if* $\widetilde{s}$ *and* $\widetilde{t}$ *have the same intersection types*, then that part can be "pumped" (as stated in Theorem 2.1). In Section III-A, we give a bound on the length of words generated by a λ-term, which will be used to give the bound of $|w_1 \cdots w_m v_m|$ in Theorem 2.1.

We define the labeled reduction relations $\xrightarrow{\alpha}_\lambda$ (where α is either ϵ or (a,i) with $a \in \mathit{dom}(\Sigma)$ and $i \in \{1,\dots,\Sigma(a)\}$) and $\overset{w}{\Longrightarrow}_\lambda$ (where w ranges over a sequence of elements of the form (a,i)) on λ-terms as the restrictions of $\xrightarrow{\alpha}_{\mathcal{G}}$ and $\overset{w}{\Longrightarrow}_{\mathcal{G}}$ obtained by removing rule R-NT.

A. Bounding the Length of Reductions

This subsection gives the length of non-ϵ reductions by using some standard results on the size of β-normal forms [5], [26]. Since the technique is rather standard, we defer some of the proofs to the extended version [18].

We write $\longrightarrow_\beta$ for the standard β-reduction relation. For a binary relation $\mathcal{R}$, we write $\mathcal{R}^*$ for the reflexive and transitive closure of $\mathcal{R}$. We first define the length of non-ϵ reductions.

Definition 3.1: Let t be a λ-term. The measure $\mathbf{R}(t)$ is defined as $\max(\{|w| \mid t \overset{w}{\Longrightarrow}_\lambda u\})$, and ∞ if there is no bound on $|w|$.

The following lemma follows immediately from the strong normalization of the simply-typed λ-terms.

Lemma 3.1: If $\mathcal{K} \vdash t : \mathtt{o}$, then there is no infinite reduction sequence $t \xrightarrow{\alpha_1}_\lambda t_1 \xrightarrow{\alpha_2}_\lambda t_2 \xrightarrow{\alpha_3}_\lambda \cdots$.

Proof: Suppose that there is an infinite reduction sequence: $t \xrightarrow{\alpha_1}_\lambda t_1 \xrightarrow{\alpha_2}_\lambda t_2 \xrightarrow{\alpha_3}_\lambda \cdots$. Since the reduction sequence consisting of only applications of R-CONST must terminate, the infinite sequence must contain infinitely many applications of R-BETA. Since $t \xrightarrow{(a,i)}_\lambda t' \xrightarrow{\epsilon}_\lambda t''$ implies $t \longrightarrow_\beta t' \xrightarrow{(a,i)}_\lambda t''$ (where $\longrightarrow_\beta$ is the β-reduction relation), we obtain an infinite β-reduction sequence, which contradicts with the strong normalization of the simply-typed λ-terms. ■

The following lemma ensures that, in order to obtain a bound for t, it suffices to consider the bound for its β-normal form.

Lemma 3.2: If $t \longrightarrow_\beta^* u$, then $\mathbf{R}(t) \leq \mathbf{R}(u)$.[4]

Next, we bound the size of a term obtained by certain β-reductions. The *order* of a β-redex $(\lambda x\!:\!\kappa.t_1)t_2$ is $\mathtt{ord}(\kappa)+1$. We write $\mathtt{rord}(t)$ for the largest order of β-redexes in t. We write $t \longrightarrow_\beta^n t'$ if $t \longrightarrow_\beta t'$ is obtained by reducing an *innermost* order-n redex. We define the *size* and the *height* of a term t, written $|t|$ and $\mathbf{H}(t)$ respectively, by:

$$\begin{array}{l} |x| = |a| = 1 \quad |\lambda x.t| = |t| + 1 \quad |t_1 t_2| = |t_1| + |t_2| + 1 \\ \mathbf{H}(x) = \mathbf{H}(a) = 0 \qquad \mathbf{H}(\lambda x.t) = \mathbf{H}(t) + 1 \\ \mathbf{H}(t_1 t_2) = \max(\mathbf{H}(t_1), \mathbf{H}(t_2) + 1) \end{array}$$

The following is a standard result [26], [18].

Lemma 3.3: If $\mathtt{rord}(t) \leq n$ and $t(\longrightarrow_\beta^n)^* u$, then $|u| \leq 2^{|t|}$.

The following lemma bounds the height of the tree generated by a term.

Lemma 3.4:
If $\mathtt{rord}(t) \leq 1$ and $t(\longrightarrow_\beta^1)^* u$, then $\mathbf{H}(u) \leq |t|$ holds.

Proof: The proof proceeds by induction on t.

- Case $t = x$ or $t = a$: Trivial, as $u = t$.
- Case $t = \lambda x.t_1$: We have $t_1(\longrightarrow_\beta^1)^* u_1$ with $u = \lambda x.u_1$. By using the induction hypothesis, we obtain $\mathbf{H}(u) = \mathbf{H}(u_1) + 1 \leq |t_1| + 1 = |t|$ as required.
- Case $t = t_1 t_2$: We have either (i) $t_1(\longrightarrow_\beta^1)^* u_1$ and $t_2(\longrightarrow_\beta^1)^* u_2$ with $u = u_1 u_2$, or (ii) $t_1(\longrightarrow_\beta^1)^* \lambda x : \mathtt{o}.u_3$ and $t_2(\longrightarrow_\beta^1)^* u_2$ with $u = [u_2/x]u_3$, and u_2, u_3 are β-normal forms. By using the induction hypothesis we obtain $\mathbf{H}(u) = \max(\mathbf{H}(u_1), \mathbf{H}(u_2) + 1) \leq \max(|t_1|, |t_2| + 1) \leq |t|$ in case (i), and $\mathbf{H}(u) \leq \mathbf{H}(u_3) + \mathbf{H}(u_2) \leq |t_1| + |t_2| \leq |t|$ in case (ii), as required. ■

We are now ready to derive a bound for $\mathbf{R}(t)$.

Theorem 3.5: If $\mathcal{K} \vdash t : \mathtt{o}$ and $\mathtt{rord}(t) = n$, then $\mathbf{R}(t) \leq \mathbf{exp}_{n-1}(|t|)$.

Proof: By Lemma 3.3, we can reduce all the order-$k\,(>1)$ redexes of t in the decreasing order of k, and obtain a term s such that $\mathtt{rord}(s) \leq 1$ and $|s| \leq \mathbf{exp}_{n-1}(|t|)$. By Lemma 3.4, we get a β-normal form u of s such that $\mathbf{H}(u) \leq |s| \leq \mathbf{exp}_{n-1}(|t|)$. Since an application of the rule R-CONST strictly decreases the height of a term, we have $\mathbf{R}(u) \leq \mathbf{H}(u) \leq \mathbf{exp}_{n-1}(|t|)$. By Lemma 3.2, we obtain $\mathbf{R}(t) \leq \mathbf{exp}_{n-1}(|t|)$ as required. ■

[4] Actually $\mathbf{R}(t) = \mathbf{R}(u)$ holds, but we only need $\mathbf{R}(t) \leq \mathbf{R}(u)$ below.

B. Intersection Types for Reduction Properties

Next, we shall develop an intersection type system for reasoning about properties of reductions. Intersection types [2], [8], [28] have been used for characterizing normalization properties of the λ-calculus, but we use them here for reasoning about the shape of terms that may/must occur during reductions. For example, a typical question to be addressed by the type system is: given a λ-term $s\,t$, may t eventually occur in a head position, and if so, what kind of argument is given to it? To reason about such a property, we assume a set $\mathbf{Lab}$ of labels that is disjoint from the set $\mathbf{Var}$ of variables, and extend λ-terms with labeled terms.

$$t ::= \cdots \mid t^{\ell}$$

Here ℓ ranges over $\mathbf{Lab}$. The sort assignment system is accordingly extended, just ignoring labels.

$$\frac{\mathcal{K} \vdash_{\Sigma} t : \kappa}{\mathcal{K} \vdash_{\Sigma} t^{\ell} : \kappa}$$

The reduction relation is extended by the following rule, which allows any label to be removed during reductions.

$$C[t^{\ell}] \xrightarrow{\epsilon} C[t] \qquad \text{(R-LAB)}$$

Here C, called a *context*, is an expression obtained by replacing a free variable of a term with a hole $[\,]$, and $C[t]$ is the term obtained by replacing the hole with t.

The syntax of (intersection) types is given by:

$$\tau ::= \mathbf{r} \mid \bigwedge\{\tau_1, \dots, \tau_n\} \to \tau$$

Following van Bakel [28], we only allow intersection on the lefthand side of $\to$. In the type $\bigwedge\{\tau_1, \dots, \tau_n\} \to \tau$, n can be 0, in which case we write $\top \to \tau$. We also write $\tau_1 \wedge \cdots \wedge \tau_k \to \tau$ for $\bigwedge\{\tau_1, \dots, \tau_k\} \to \tau$, and write $\bigwedge_{i\in I} \tau_i \to \tau$ for $\bigwedge\{\tau_i \mid i \in I\} \to \tau$. When we write $\tau_1 \wedge \cdots \wedge \tau_k \to \tau$, the constructor $\wedge$ binds tighter than $\to$.

Intuitively, the type $\mathbf{r}$ describes terms of sort $\mathtt{o}$ that can be reduced to $\mathtt{e}$. For example, under the empty type environment, $\mathtt{a}\,\mathtt{e}\,x$ has type $\mathbf{r}$ (assuming that $\Sigma(\mathtt{a}) = 2$), but $\mathtt{a}\,x\,x$ does not. The type $\bigwedge\{\tau_1, \dots, \tau_n\} \to \tau$ describes functions that take an argument that has type τ_i for every $i \in \{1, \dots, n\}$ and return an element of type τ, by using the argument as values of types $\tau_1, \dots, \tau_n$. For example, $\lambda x.x$ has type $\mathbf{r} \to \mathbf{r}$ because, given a term t that is reducible to $\mathtt{e}$, $(\lambda x.x)t$ is reducible to $\mathtt{e}$ by using t. The term $\lambda x.\mathtt{e}$, however, has type $\top \to \mathbf{r}$ but not $\mathbf{r} \to \mathbf{r}$. Although $(\lambda x.\mathtt{e})t$ is reducible to $\mathtt{e}$, t is not used in the reduction. Thus, a closed term t having type $\mathbf{r} \to \mathbf{r}$ implies that $t\,s$ has a reduction sequence of the form $t\,s \xRightarrow{w_1}_{\lambda} s \xRightarrow{w_2}_{\lambda} \mathtt{e}$ for every term s of type $\mathbf{r}$.

More generally, a term t having a type $\tau \to \mathbf{r}$ means that for any term s of type τ, $t\,s$ can be reduced to a term of the form $s\,\widetilde{u}$ where s in the head position has type τ. For example, a term t having type $(\mathbf{r} \to \mathbf{r}) \to \mathbf{r}$ means that for any term s of type $\mathbf{r} \to \mathbf{r}$, t has a reduction sequence of the form $t\,s \xRightarrow{w_1}_{\lambda} s\,u \xRightarrow{w_2}_{\lambda} u$. In this manner, types can be used for reasoning about the shape of terms in reductions.

We define the relation $\tau :: \kappa$, which should be read "τ is a refinement of κ", by:

$$\frac{}{\mathbf{r} :: \mathtt{o}} \qquad \frac{\tau :: \kappa_2 \qquad \tau_i :: \kappa_1 \text{ for each } i \in I}{(\bigwedge_{i\in I} \tau_i \to \tau) :: \kappa_1 \to \kappa_2}$$

The relation $\tau :: \kappa$ is used to exclude ill-formed types like $\mathbf{r} \wedge (\mathbf{r} \to \mathbf{r}) \to \mathbf{r}$.

A *type judgment* is of the form $\Gamma \vdash_{\Sigma} t : \tau$, where Γ, called a type environment, is a set of bindings of the form $x : \tau$ or $\ell : \tau$ (where x is a variable and ℓ is a label). We often omit the subscript Σ below. Γ may have multiple bindings for the same variable or label. We write $\Gamma(v)$ (where v is a variable or a label) for the set $\{\tau \mid v : \tau \in \Gamma\}$. We write $dom(\Gamma)$ for the set $\{v \mid v : \tau \in \Gamma\}$. We write $\Gamma_{\mathbf{V}}$ for the set $\{x : \tau \in \Gamma \mid x \in \mathbf{Var}\}$ of the bindings on variables, and $\Gamma_{\mathcal{L}}$ for the set $\{\ell : \tau \in \Gamma \mid \ell \in \mathbf{Lab}\}$ of those on labels in Γ.

Intuitively, $\Gamma \vdash_{\Sigma} t : \tau$ means that if each variable x in t is bound to a term having all the types in $\Gamma(x)$, then t behaves like a term of type τ, and each term labeled by ℓ behaves like a term of types $\Gamma(\ell)$. For example, if $x : \tau_1, \ell : \tau_2 \vdash_{\Sigma} t : \mathbf{r}$, then for any term t_1 with $\emptyset \vdash_{\Sigma} t_1 : \tau_1$, there exist w, t_2, and $\widetilde{s}$ such that $[t_1/x]t \xRightarrow{w}_{\lambda} (t_2)^{\ell}\,\widetilde{s}$ and t_2 has type τ_2. (Such properties are formally stated later in Theorems 3.6 and 3.7 below.)

The type system consists of the following rules.

$$x : \tau \vdash_{\Sigma} x : \tau \qquad \text{(T-VAR)}$$

$$\emptyset \vdash_{\Sigma} \mathtt{e} : \mathbf{r} \qquad \text{(T-CONSTE)}$$

$$\frac{\begin{array}{c}\Sigma(a) = k \qquad \tau_j = \mathbf{r} \text{ for some } j \in \{1, \dots, k\} \\ \tau_i = \top \text{ for every } i \in \{1, \dots, k\} \setminus \{j\}\end{array}}{\emptyset \vdash_{\Sigma} a : \tau_1 \to \cdots \to \tau_k \to \mathbf{r}} \qquad \text{(T-CONST)}$$

$$\frac{\Gamma_0 \vdash_{\Sigma} t_0 : \bigwedge_{i\in I} \tau_i \to \tau \qquad \Gamma_i \vdash_{\Sigma} t_1 : \tau_i \text{ for every } i \in I}{\Gamma_0 \cup (\bigcup_{i\in I} \Gamma_i) \vdash_{\Sigma} t_0 t_1 : \tau}$$

(T-APP)

$$\frac{\begin{array}{c}\Gamma \cup \{x : \tau_i \mid i \in I\} \vdash_{\Sigma} t : \tau \\ \tau_i :: \kappa \text{ for every } i \in I \qquad x \notin dom(\Gamma)\end{array}}{\Gamma \vdash_{\Sigma} \lambda x : \kappa.t : \bigwedge_{i\in I} \tau_i \to \tau} \qquad \text{(T-ABS)}$$

$$\frac{\Gamma \vdash_{\Sigma} t : \tau}{\Gamma \cup \{\ell : \tau\} \vdash_{\Sigma} t^{\ell} : \tau} \qquad \text{(T-LAB)}$$

The typing rules above are fairly standard, except the following points.

1) The type of a constant is determined by its arity; if $\Sigma(a) = n > 0$, then a has n types $\mathbf{r} \to \top \to \cdots \to \top \to \mathbf{r}$, $\top \to \mathbf{r} \to \top \to \cdots \to \top \to \mathbf{r}, \dots, \top \to \cdots \to \top \to \mathbf{r} \to \mathbf{r}$. This reflects the fact that $a\,t_1 \cdots t_n$ can be reduced to any one of $t_1, \dots, t_n$.

2) Weakening of a type environment is disallowed (i.e., $\Gamma \vdash_{\Sigma} t : \tau$ does not necessarily imply $\Gamma, x : \tau' \vdash_{\Sigma} t : \tau$) but contraction is allowed (i.e., $\Gamma, x : \tau', x : \tau' \vdash_{\Sigma} t : \tau$ implies

$\Gamma, x : \tau' \vdash_\Sigma t : \tau$, since a type environment is a *set* of type bindings); therefore, $x : \tau \in \Gamma$ means that x is used as a value of type τ *at least* once in some reduction sequence of t. If weakening were allowed, we would obtain $\emptyset \vdash_\Sigma \lambda x.\mathtt{e} : \mathbf{r} \to \mathbf{r}$, but $(\lambda x.\mathtt{e})s$ cannot be reduced to s.

3) In rule T-App, it may be the case that $\tau_i = \tau_j$ even if $i \neq j$ for $i, j \in I$. Thus, $x : \mathbf{r}, y : \mathbf{r} \vdash (\lambda z.z)(\mathtt{a}\, x\, y) : \mathbf{r}$ is derivable from $x : \mathbf{r} \vdash \mathtt{a}\, x\, y : \mathbf{r}$ and $y : \mathbf{r} \vdash \mathtt{a}\, x\, y : \mathbf{r}$. Note that $x : \mathbf{r}, y : \mathbf{r} \vdash \mathtt{a}\, x\, y : \mathbf{r}$ is not derivable.

Remark 3.1: If $\Gamma \vdash t : \tau$, then $dom(\Gamma_{\mathbf{V}}) \subseteq \mathbf{FV}(t)$ (because weakening is disallowed), but the converse $dom(\Gamma_{\mathbf{V}}) \supseteq \mathbf{FV}(t)$ may not hold. For example, $\emptyset \vdash \mathtt{a}\, \mathtt{e}\, x : \mathbf{r}$ is obtained by assigning $\mathbf{r} \to \top \to \mathbf{r}$ to a constant $\mathtt{a}$ of arity 2. □

Example 3.1: Let t be $(\lambda f.f(f^\ell(x)))(\lambda z.\mathtt{a}\, z\, \mathtt{e})$. Then, we can obtain $\ell : \top \to \mathbf{r} \vdash (\lambda f.f(f^\ell(x)))(\lambda z.\mathtt{a}\, z\, \mathtt{e}) : \mathbf{r}$ from

$$\begin{array}{l} \ell : \top \to \mathbf{r} \vdash (\lambda f.f(f^\ell(x))) : (\top \to \mathbf{r}) \wedge (\mathbf{r} \to \mathbf{r}) \to \mathbf{r} \\ \emptyset \vdash \lambda z.\mathtt{a}\, z\, \mathtt{e} : \top \to \mathbf{r} \qquad \emptyset \vdash \lambda z.\mathtt{a}\, z\, \mathtt{e} : \mathbf{r} \to \mathbf{r}. \end{array}$$

Here, the three judgments can be derived as follows.

$$\dfrac{\dfrac{f : \mathbf{r} \to \mathbf{r} \vdash f : \mathbf{r} \to \mathbf{r} \qquad \dfrac{\dfrac{f : \top \to \mathbf{r} \vdash f : \top \to \mathbf{r}}{f : \top \to \mathbf{r}, \ell : \top \to \mathbf{r} \vdash f^\ell : \top \to \mathbf{r}}}{f : \top \to \mathbf{r}, \ell : \top \to \mathbf{r} \vdash f^\ell(x) : \mathbf{r}}}{f : \top \to \mathbf{r}, f : \mathbf{r} \to \mathbf{r}, \ell : \top \to \mathbf{r} \vdash f(f^\ell(x)) : \mathbf{r}}}{\ell : \top \to \mathbf{r} \vdash (\lambda f.f(f^\ell(x))) : (\top \to \mathbf{r}) \wedge (\mathbf{r} \to \mathbf{r}) \to \mathbf{r}}$$

$$\dfrac{\dfrac{\dfrac{\emptyset \vdash \mathtt{a} : \top \to \mathbf{r} \to \mathbf{r}}{\emptyset \vdash \mathtt{a}\, z : \mathbf{r} \to \mathbf{r}} \qquad \emptyset \vdash \mathtt{e} : \mathbf{r}}{\emptyset \vdash \mathtt{a}\, z\, \mathtt{e} : \mathbf{r}}}{\emptyset \vdash \lambda z.\mathtt{a}\, z\, \mathtt{e} : \top \to \mathbf{r}}$$

$$\dfrac{\dfrac{\dfrac{\emptyset \vdash \mathtt{a} : \mathbf{r} \to \top \to \mathbf{r} \quad z : \mathbf{r} \vdash z : \mathbf{r}}{z : \mathbf{r} \vdash \mathtt{a}\, z : \top \to \mathbf{r}}}{z : \mathbf{r} \vdash \mathtt{a}\, z\, \mathtt{e} : \mathbf{r}}}{\emptyset \vdash \lambda z.\mathtt{a}\, z\, \mathtt{e} : \mathbf{r} \to \mathbf{r}}$$

□

We use meta-variable σ for $\bigwedge\{\tau_1, \ldots, \tau_n\}$, and we write $\Gamma \vdash t : \bigwedge\{\tau_1, \ldots, \tau_n\}$ if $\Gamma_i \vdash t : \tau_i$ for every $i \in I$, with $\Gamma = \bigcup_{i \in \{1,\ldots,n\}} \Gamma_i$.

The type system ensures certain properties of reductions of λ-terms, as stated in the following theorems. They follow from standard properties of intersection type systems, like (a restricted form of) subject reduction (typing is preserved by reduction: see Lemma 3.9) and subject expansion (typing is preserved by the inverse of reduction: see Lemma 3.11).

Theorem 3.6 (soundness and completeness I): If $\mathcal{K} \vdash_\Sigma t : \mathtt{o}$, then the following are equivalent.

(i) $\Gamma \cup \{\ell : \sigma_1 \to \cdots \sigma_k \to \mathbf{r}\} \vdash t : \mathbf{r}$ for some Γ such that $\Gamma_{\mathbf{V}} = \emptyset$.

(ii) There exist $s_0, s_1, \ldots, s_k$, w, $\Gamma_0, \Gamma_1, \ldots, \Gamma_k$ such that $t \overset{w}{\Longrightarrow}_\lambda (s_0)^\ell s_1, \ldots, s_k$, $\Gamma_0 \vdash s_0 : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}$ and $\Gamma_i \vdash s_i : \sigma_i$ for every $i \in \{1, \ldots, k\}$
with $(\Gamma_0 \cup \cdots \cup \Gamma_k)_{\mathbf{V}} = \emptyset$.

Theorem 3.7 (soundness and completeness II): If $\mathcal{K} \vdash_\Sigma t : \mathtt{o}$, then the following are equivalent.

(i) $\Gamma \vdash t : \mathbf{r}$ for some Γ such that $\Gamma_{\mathbf{V}} = \emptyset$.

(ii) $t \overset{w}{\Longrightarrow}_\lambda \mathtt{e}$ for some w.

We prove Theorem 3.6 below, and defer the proof of Theorem 3.7 to the extended version [18].

We first state the substitution lemma: see [18] for a proof.

Lemma 3.8 (substitutions): Suppose that $\tau_1, \ldots, \tau_n$ are different from each other. If $\Gamma_0 \cup \{x : \tau_1, \ldots, x : \tau_n\} \vdash t : \tau$ and $\Gamma_i \vdash s : \tau_i$ for each $i \in \{1, \ldots, n\}$ with $x \notin dom(\Gamma_0)$, then $\Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_n \vdash [s/x]t : \tau$.

The following lemma states soundness of the type system, of which the direction from (i) to (ii) of Theorem 3.6 is an immediate corollary.

Lemma 3.9: If $\Gamma \cup \{\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}\} \vdash t : \mathbf{r}$ and $\Gamma_{\mathbf{V}} = \emptyset$, then either (i) $t = (s_0)^\ell s_1, \ldots, s_k$, $\Gamma_0 \vdash s_0 : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}$ and $\Gamma_i \vdash s_i : \sigma_i$ for each $i \in \{1, \ldots, k\}$ with $(\Gamma_0 \cup \cdots \cup \Gamma_k)_{\mathbf{V}} = \emptyset$; or (ii) $t \xrightarrow{\alpha}_\lambda t'$ and $\Gamma' \vdash t' : \mathbf{r}$ for some t' and Γ' such that $\Gamma'_{\mathbf{V}} = \emptyset$ and $\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r} \in \Gamma'$.

Proof: We show the lemma by case analysis on the shape of t. Since $\Gamma_{\mathbf{V}} = \emptyset$, the head term of t cannot be a variable.

- Case where $t \equiv (\lambda x.t_0)t_1 \cdots t_m$: We have:

$$\begin{array}{l} \Gamma_0 \cup \{x : \tau_1, \ldots, x : \tau_n\} \vdash t_0 : \tau \\ \Gamma_j \vdash t_1 : \tau_{p_j} \text{ for each } j \in J \qquad \{p_j \mid j \in J\} = \{1, \ldots, n\} \\ \Gamma \cup \{\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}\} = \Gamma_0 \cup (\bigcup_{j \in J} \Gamma_j) \\ \tau_1, \ldots, \tau_n \text{ are different from each other.} \end{array}$$

For each $i \in \{1, \ldots, n\}$, pick $j_i \in J$ so that $p_{j_i} = i$ and $\Gamma_0 \cup (\bigcup_{i \in \{1,\ldots,n\}} \Gamma_{j_i}) \ni \ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}$. By Lemma 3.8, condition (ii) holds for $t' = ([t_1/x]t_0)t_2 \cdots t_m$ and $\Gamma' = \Gamma_0 \cup (\bigcup_{i \in \{1,\ldots,n\}} \Gamma_{j_i})$ with $\alpha = \epsilon$.

- Case where $t \equiv a\, t_1 \cdots t_m$: We have $m > 0$ and

$$\begin{array}{l} \Gamma_j \vdash t_i : \mathbf{r} \text{ for } j \in \{1, \ldots, n\} \\ \Gamma_1 \cup \cdots \cup \Gamma_n = \Gamma \cup \{\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}\} \end{array}$$

for some $i \in \{1, \ldots, m\}$. Pick $j \in \{1, \ldots, n\}$ such that $\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r} \in \Gamma_j$. Then, condition (ii) holds for $t' = t_i$ and $\Gamma' = \Gamma_j$ with $\alpha = (a, i)$.

- Case where $t \equiv t_0^{\ell'}\, t_1 \cdots t_m$: We have:

$$\begin{array}{l} \Gamma_0 \vdash t_0 : \sigma'_1 \to \cdots \to \sigma'_m \to \mathbf{r} \\ \Gamma_i \vdash t_i : \sigma'_i \text{ for each } i \in \{1, \ldots, m\} \\ \Gamma \cup \{\ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}\} = \\ \quad \Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_m \cup \{\ell' : \sigma'_1 \to \cdots \to \sigma'_m \to \mathbf{r}\} \end{array}$$

If $\ell' : \sigma'_1 \to \cdots \to \sigma'_m \to \mathbf{r} = \ell : \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}$, we have condition (i). Otherwise, condition (ii) holds for $t' = t_0\, t_1 \cdots t_m$ and $\Gamma' = \Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_m$ with $\alpha = \epsilon$. ■

The following lemmas state that typing is preserved by the inverse of substitutions and reductions.

Lemma 3.10 (inverse substitution): If $\Gamma \vdash [t_1/x]t_0 : \tau$, then there exist a finite set I, types $\tau_i (i \in I)$ and type environments $\Gamma_i (i \in \{0\} \cup I)$ that satisfy the following conditions.

$$\begin{array}{l} \Gamma_0 \cup \{x : \tau_i \mid i \in I\} \vdash t_0 : \tau \\ \Gamma_i \vdash t_1 : \tau_i \text{ for each } i \in I \qquad \bigcup_{i \in \{0\} \cup I} \Gamma_i = \Gamma. \end{array}$$

Lemma 3.11 (subject expansion): If $\Gamma' \vdash t' : \mathbf{r}$ and $t \xrightarrow{\alpha}_\lambda t'$, then $\Gamma \vdash t : \mathbf{r}$ for some Γ such that $\Gamma_{\mathbf{V}} = \Gamma'_{\mathbf{V}}$ and $\Gamma_{\mathcal{L}} \supseteq \Gamma'_{\mathcal{L}}$.

Proof: The proof follows by case analysis on the rule used for deriving $t \xrightarrow{\alpha}_\lambda t'$.

- Case R-BETA: In this case, we have:

$$\begin{array}{ll} t = (\lambda x.t_0)t_1t_2\cdots t_m & t' = ([t_1/x]t_0)t_2\cdots t_m \\ \Gamma'_1 \vdash [t_1/x]t_0 : \sigma_2 \to \cdots \to \sigma_m \to \mathbf{r} & \\ \Gamma'_i \vdash t_i : \sigma_i \text{ for } i \in \{2,\ldots,m\} & \Gamma' = \Gamma'_1 \cup \cdots \Gamma'_m. \end{array}$$

By applying Lemma 3.10 to $\Gamma'_1 \vdash [t_1/x]t_0 : \tau$, we obtain:

$$\begin{array}{l} \Gamma_0 \cup \{x:\tau_1,\ldots,x:\tau_n\} \vdash t_0 : \sigma_2 \to \cdots \to \sigma_m \to \mathbf{r} \\ \Gamma_i \vdash t_1 : \tau_i \text{ for each } i \in \{1,\ldots,n\} \\ \Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_n = \Gamma'_1 \end{array}$$

By using T-ABS and T-APP, we obtain

$$\Gamma'_1 \vdash (\lambda x.t_0)t_1 : \sigma_2 \to \cdots \to \sigma_m \to \mathbf{r}.$$

Thus, the required result holds for $\Gamma = \Gamma'$.

- Case R-CONST: In this case, $t = a\,t_1 \cdots t_n$ and $t' = t_i$ for some $i \in \{1,\ldots,n\}$. By the assumption, we have $\Gamma' \vdash t_i : \mathbf{r}$. Let $\Gamma = \Gamma'$. By using T-CONST and T-APP, we obtain $\Gamma \vdash t : \mathbf{r}$ as required.
- Case R-LAB: In this case, $t = C[t_0^\ell]$ and $t' = C[t_0]$. Let $\{\Gamma_1 \vdash t_0 : \tau_1, \ldots, \Gamma_n \vdash t_0 : \tau_n\}$ be the set of judgments for t_0 occurring in the derivation for $\Gamma' \vdash t' : \tau$. Then, the required result holds for $\Gamma = \Gamma' \cup \{\ell : \tau_1, \ldots, \ell : \tau_n\}$. (More formally, this follows by induction on the structure of C.)

■

We are now ready to prove Theorem 3.6.

Proof of Theorem 3.6:

- (i)⇒(ii): Suppose that (i) holds but (ii) does not hold. Then, by Lemma 3.9, there must be an infinite reduction sequence $t \xrightarrow{\alpha_1}_\lambda t_1 \xrightarrow{\alpha_2}_\lambda t_2 \xrightarrow{\alpha_3}_\lambda \cdots$. Let t' be the term obtained from t by removing all the labels. Then from the sequence above, we have an infinite reduction sequence

$$t' \xrightarrow{\alpha'_1}_\lambda t'_1 \xrightarrow{\alpha'_2}_\lambda t'_2 \xrightarrow{\alpha'_3}_\lambda \cdots$$

that does not use R-LAB. (Note that consecutive applications of R-LAB must be finite in the reduction sequence of t.) However, this cannot be the case by Lemma 3.1.

- (ii)⇒(i): If (ii) holds, then we have

$$\ell : \sigma_1 \to \cdots \sigma_k \to \mathbf{r} \vdash (s_0)^\ell s_1, \ldots, s_k : \mathbf{r}.$$

Condition (i) follows by repeated applications of Lemma 3.11.

■

IV. Proof of Pumping Lemma (Theorem 2.1)

This section proves Theorem 2.1 by using the techniques developed in Section III. We fix $\mathcal{G} = (\Sigma, \mathcal{N}, \mathcal{R}, S)$ below. We first give the constants c_1 and c_2 in the theorem. We define a few constants determined by $\mathcal{G}$.

- $B_{\mathcal{G}}$: the largest size of the body of a rewriting rule, i.e., $\max(\{|\mathcal{R}(F)| \mid F \in dom(\mathcal{N})\})$.
- $T_{\mathcal{G}}$: the number of type bindings of the form $F : \tau$ that conform to $\mathcal{N}$, i.e., $|\{F : \tau \mid F \in dom(\mathcal{N}), \tau :: \mathcal{N}(F)\}|$. Note that it is finite.

The constants c_1 and c_2 are given by: $c_1 = B_{\mathcal{G}}^{T_{\mathcal{G}}}$ and $c_2 = T_{\mathcal{G}} + 1$.

As explained at the beginning of Section III, the key observation for the proof of the theorem is that for a sufficiently long reduction sequence $S \overset{w}{\Longrightarrow} \mathtt{e}$, there exists a sub-sequence of the form $F\,\widetilde{s} \overset{w'}{\Longrightarrow} F\,\widetilde{t}$ such that (i) the second F comes from the first F (instead of $\widetilde{s}$), and (ii) the types of the two F are the same, so that the part $F\,\widetilde{s} \overset{w'}{\Longrightarrow} F\,\widetilde{t}$ can be pumped. In order to talk about condition (i), we first extend the reduction relation $\xrightarrow{\alpha}_{\mathcal{G}}$.

$$\begin{array}{l} (F^{\ell'}\,s_1\cdots s_n, \prec) \xrightarrow{\epsilon}_{\mathcal{G}} \\ \quad (([F_1^\ell/F_1,\ldots,F_k^\ell/F_k]\mathcal{R}(F))s_1\cdots s_n, \prec \cup\{(\ell',\ell)\}) \\ \hfill (\ell \text{ fresh}) \end{array}$$

$$(t, \prec) \xrightarrow{\alpha}_{\mathcal{G}} (t', \prec) \quad \text{if } t \xrightarrow{\alpha}_\lambda t'$$

Here, we assume that $dom(\mathcal{N}) = \{F_1,\ldots,F_k\}$. The relation $\overset{w}{\Longrightarrow}_{\mathcal{G}}$ is extended accordingly. The label (ℓ, ℓ' above) for a non-terminal is used to express how a non-terminal has been introduced; when $(S^{\ell_0}, \emptyset) \overset{w}{\Longrightarrow}_{\mathcal{G}} (t, \prec)$, the relation $\ell' \prec \ell$ intuitively means that the non-terminals labeled by ℓ have been introduced by expanding a non-terminal labeled by ℓ'. If $(S^{\ell_0}, \emptyset) \overset{w}{\Longrightarrow}_{\mathcal{G}} (t, \prec)$ and ℓ occurs in $\prec$, then there is exactly one sequence of the form $\ell_0 \prec \ell_1 \prec \cdots \prec \ell_m = \ell$. We write $\mathtt{depth}_\prec(\ell)$ for m.

Example 4.1: Recall $\mathcal{G}_1$ from Example 2.1. S is reduced as follows by the extended reduction.

$$\begin{array}{l} (S^{\ell_0}, \emptyset) \xrightarrow{\epsilon}_{\mathcal{G}_1} (F^{\ell_1}\,\mathtt{e}, \{(\ell_0,\ell_1)\}) \\ \quad \overset{\epsilon}{\Longrightarrow}_{\mathcal{G}_1} (\mathtt{a}\,(F^{\ell_2}\,(\mathtt{b}\,\mathtt{e}))\,\mathtt{e}, \{(\ell_0,\ell_1),(\ell_1,\ell_2)\}) \\ \quad \xrightarrow{(\mathtt{a},1)}_{\mathcal{G}} (F^{\ell_2}\,(\mathtt{b}\,\mathtt{e}), \{(\ell_0,\ell_1),(\ell_1,\ell_2)\}) \xrightarrow{(\mathtt{a},1)}_{\mathcal{G}} \cdots \end{array}$$

□

In order to use the results of Section III, we need to "approximate" each non-terminal by a finite unfolding of it and remove recursion. We define λ-terms ${F_i}^{\langle m \rangle}$ by:

$${F_i}^{\langle 0 \rangle} = F_i \qquad {F_i}^{\langle m+1 \rangle} = [{F_1}^{\langle m \rangle}/F_1,\ldots,{F_k}^{\langle m \rangle}/F_k]\mathcal{R}(F_i)$$

Intuitively, ${F_i}^{\langle m \rangle}$ is an "approximation" of F_i, obtained by unfolding each non-terminal m times. We also write $\theta^{\langle m \rangle}$ for the substitution $[{F_1}^{\langle m \rangle}/F_1,\ldots,{F_n}^{\langle m \rangle}/F_n]$. Note that $\theta^{\langle m \rangle}F_i = {F_i}^{\langle m \rangle}$.

Let us write $t \preceq t'$ if t' is obtained by unfolding some non-terminals F_i in t with ${F_i}^{\langle m \rangle}$. We repeatedly use the following properties: (i) if $t \overset{w}{\Longrightarrow}_{\mathcal{G}} s$, then $\theta^{\langle m \rangle}t \overset{w}{\Longrightarrow}_\lambda s'$ for some m and s' such that $s \preceq s'$; and (ii) if $t \preceq t' \overset{w}{\Longrightarrow}_\lambda s'$, then $t \overset{w}{\Longrightarrow}_{\mathcal{G}} s \preceq s'$ for some s.

Example 4.2: Recall $\mathcal{G}_1$ in Example 2.1.

$$S^{\langle 2 \rangle} = F^{\langle 1 \rangle}\,\mathtt{e} = (\lambda x.\mathtt{a}\,(F^{\langle 0 \rangle}\,(\mathtt{b}\,x))\,x)\mathtt{e} = (\lambda x.\mathtt{a}\,(F\,(\mathtt{b}\,x))\,x)\mathtt{e}.$$

The reduction from S to $F\,(\mathtt{b}\,\mathtt{e})$ in Example 4.1 can be simulated by: $S^{\langle 2 \rangle} \xrightarrow{\epsilon}_\lambda \mathtt{a}\,(F\,(\mathtt{b}\,\mathtt{e}))\,\mathtt{e} \xrightarrow{(\mathtt{a},1)}_\lambda F\,(\mathtt{b}\,\mathtt{e})\;\cdot$ □

The following lemma states that a sufficiently long reduction sequence must contain a certain number of unfoldings.

Lemma 4.1: Suppose $(S^{\ell_0}, \emptyset) \overset{w}{\Longrightarrow}_{\mathcal{G}} (t, \prec)$. If $|w| > \mathbf{exp}_{\mathtt{ord}(\mathcal{G})-1}(B_{\mathcal{G}}^{m-1})$, then there exist $\ell_0, \ell_1, \ldots, \ell_m$ such that $\ell_0 \prec \ell_1 \prec \cdots \prec \ell_m$.

Proof: The proof is by contradiction. Suppose that there is no increasing chain $\ell_0 \prec \ell_1 \prec \cdots \prec \ell_m$ of length m. Then, the reduction $S \overset{w}{\Longrightarrow}_{\mathcal{G}} t$ can be simulated by $S^{\langle m-1 \rangle} \overset{w}{\Longrightarrow}_{\lambda} t'$. By Theorem 3.5, $|w| \leq \mathbf{exp}_{\mathtt{ord}(\mathcal{G})-1}(|S^{\langle m-1 \rangle}|) \leq \mathbf{exp}_{\mathtt{ord}(\mathcal{G})-1}(B_{\mathcal{G}}^{m-1})$, hence a contradiction. ■

The following lemma guarantees that a constant number of unfoldings is actually sufficient to simulate the reduction properties captured by the type system in the previous section.

Lemma 4.2: If $\emptyset \vdash \theta^{\langle m \rangle} t : \tau$ for some m, then $\emptyset \vdash \theta^{\langle m' \rangle} t : \tau$ for every $m' \geq T_{\mathcal{G}}$.

Proof: By Lemmas 3.8 and 3.10, it suffices to show that $\emptyset \vdash F^{\langle m \rangle} : \tau$ implies $\emptyset \vdash F^{\langle m' \rangle} : \tau$ for every $m' \geq T_{\mathcal{G}}$, non-terminal F, and type τ. Let $\Gamma^{\langle m \rangle}$ be:

$$\{F : \tau \mid \emptyset \vdash F^{\langle m \rangle} : \tau \text{ and } \tau :: \mathcal{N}(F)\}.$$

We define $\mathcal{F}$ by:

$$\mathcal{F}(\Gamma) = \{F_i : \tau \mid \Gamma' \vdash \mathcal{R}(F_i) : \tau \text{ for some } \Gamma' \subseteq \Gamma\}.$$

By the definition of $F^{\langle m \rangle}$ and Lemmas 3.8 and 3.10, we have:

$$\begin{array}{l} \emptyset \vdash F^{\langle m \rangle} : \tau \\ \Leftrightarrow \{F_i : \tau_{i,j} \mid i \in \{1, \ldots, k\}, j \in I_i\} \vdash \mathcal{R}(F) : \tau \text{ and} \\ \qquad \emptyset \vdash F_i{}^{\langle m-1 \rangle} : \tau_{i,j} \text{ for each } i, j \\ \Leftrightarrow F : \tau \in \mathcal{F}(\Gamma^{\langle m-1 \rangle}) \end{array}$$

Thus, we have $\Gamma^{\langle m \rangle} = \mathcal{F}(\Gamma^{\langle m-1 \rangle})$. Since $\mathcal{F}$ is monotonic, $\Gamma^{\langle m \rangle}$ forms a monotonically-increasing sequence:

$$\Gamma^{\langle 0 \rangle}(= \emptyset) \subseteq \Gamma^{\langle 1 \rangle} \subseteq \cdots \subseteq \Gamma^{\langle T_{\mathcal{G}} \rangle} \subseteq \cdots.$$

Since $|\Gamma^{\langle m \rangle}| \leq T_{\mathcal{G}}$, it must be the case that $\Gamma^{\langle k \rangle} = \Gamma^{\langle k+1 \rangle}$ for some $k \leq T_{\mathcal{G}}$. By the condition $\Gamma^{\langle m \rangle} = \mathcal{F}(\Gamma^{\langle m-1 \rangle})$, we have $\Gamma^{\langle m \rangle} = \mathcal{F}^{m-k}(\Gamma^{\langle k \rangle}) = \mathcal{F}^{m-k}(\Gamma^{\langle k+1 \rangle}) = \Gamma^{\langle m+1 \rangle}$ for every $m \geq T_{\mathcal{G}}$. Thus, we have $\Gamma^{\langle m \rangle} \subseteq \Gamma^{\langle T_{\mathcal{G}} \rangle}$ for every $m \geq 0$, which completes the proof. ■

We are now ready to prove the main theorem. Since the proof uses heavy notations, while reading the proof, the reader may wish to consult Example 4.3, which provides examples of the constructions.

Proof of Theorem 2.1: Let c_1 and c_2 be as given at the beginning of this section. Assume that $S \overset{w}{\Longrightarrow}_{\mathcal{G}} \mathtt{e}$ for some w such that $|w| > \mathbf{exp}_{\mathtt{ord}(\mathcal{G})-1}(c_1) = \mathbf{exp}_{\mathtt{ord}(\mathcal{G})-1}(B_{\mathcal{G}}^{T_{\mathcal{G}}})$. By the assumption, we have an extended reduction sequence $(S^{\ell_0}, \emptyset) \overset{w}{\Longrightarrow}_{\mathcal{G}} (\mathtt{e}, \prec)$ for some ℓ and $\prec$. By Lemma 4.1, there is at least one increasing sequence:

$$\ell_0 \prec \ell_1 \prec \cdots \prec \ell_{T_{\mathcal{G}}+1}.$$

Choose such an increasing sequence with $\ell_{T_{\mathcal{G}}+1}$ introduced in the earliest step of the reduction sequence (so that there are no $\ell'_1, \ldots, \ell'_{T_{\mathcal{G}}+1}$ such that $\ell_0 \prec \ell'_1 \prec \cdots \prec \ell'_{T_{\mathcal{G}}+1}$ and $\ell'_{T_{\mathcal{G}}+1}$ is introduced earlier than $\ell_{T_{\mathcal{G}}+1}$).

By definition, the reduction sequence $(S^{\ell_0}, \emptyset) \overset{w}{\Longrightarrow}_{\mathcal{G}} (\mathtt{e}, \prec)$ is of the form:

$$\begin{array}{r} (F_{j_0}^{\ell_0}, \emptyset) \overset{u_1}{\Longrightarrow}_{\mathcal{G}} (F_{j_1}^{\ell_1} \widetilde{s_1}, \prec_1) \overset{u_2}{\Longrightarrow}_{\mathcal{G}} \cdots \overset{u_{T_{\mathcal{G}}}}{\Longrightarrow}_{\mathcal{G}} \\ (F_{j_{T_{\mathcal{G}}}}^{\ell_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}}, \prec_{T_{\mathcal{G}}}) \overset{u_{T_{\mathcal{G}}+1}}{\Longrightarrow}_{\mathcal{G}} (\mathtt{e}, \prec) \qquad (*) \end{array}$$

where $S = F_{j_0}$. By replacing F_{j_0} in the reduction sequence with $F_{j_0}{}^{\langle 2T_{\mathcal{G}} \rangle}$ and appropriately labeling subterms, we obtain the reduction sequence:

$$t_0^{\ell_0} \overset{u_1}{\Longrightarrow}_{\lambda} (t_1)^{\ell_1} \widetilde{s'_1} \overset{u_2}{\Longrightarrow}_{\lambda} \cdots \overset{u_{T_{\mathcal{G}}}}{\Longrightarrow}_{\lambda} (t_{T_{\mathcal{G}}})^{\ell_{T_{\mathcal{G}}}} \widetilde{s'_{T_{\mathcal{G}}}} \overset{u'}{\Longrightarrow}_{\lambda} \mathtt{e} \qquad (**)$$

where:

- $\widetilde{s'_i}$ is the sequence of terms obtained from $\widetilde{s_i}$ by replacing each non-terminal F_j^r with $F_j{}^{\langle 2T_{\mathcal{G}} - \mathtt{depth}_{\prec}(r) \rangle}$; and
- t_i is the term obtained from $F_{j_i}{}^{\langle 2T_{\mathcal{G}} - i \rangle}$, by labeling each subterm $F_{j_m}{}^{\langle 2T_{\mathcal{G}} - m \rangle}$ (where $i < m \leq T_{\mathcal{G}}$) with ℓ_m.

Note that the reduction $\overset{u_i}{\Longrightarrow}_{\mathcal{G}}$ has been replaced by $\overset{u_i}{\Longrightarrow}_{\lambda}$. Except the last step, the replacement is justified by the fact that each non-terminal is unfolded at most $T_{\mathcal{G}}$ times in the original reduction sequence from $S(= F_{j_0})$ to $F_{j_{T_{\mathcal{G}}}}^{\ell_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}}$. As for the last step $(t_{T_{\mathcal{G}}})^{\ell_{T_{\mathcal{G}}}} \widetilde{s'_{T_{\mathcal{G}}}} \overset{u'}{\Longrightarrow}_{\lambda} \mathtt{e}$, from $F_{j_{T_{\mathcal{G}}}}^{\ell_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}} \overset{u_{T_{\mathcal{G}}+1}}{\Longrightarrow}_{\mathcal{G}} \mathtt{e}$, we obtain $\theta^{\langle m \rangle}(F_{j_{T_{\mathcal{G}}}}^{\ell_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}}) \overset{u_{T_{\mathcal{G}}+1}}{\Longrightarrow}_{\lambda} \mathtt{e}$ for some m. By Theorem 3.7, we have $\emptyset \vdash \theta^{\langle m \rangle}(F_{j_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}}) : \mathbf{r}$. By Lemma 4.2, we obtain $\emptyset \vdash (t_{T_{\mathcal{G}}}) \widetilde{s'_{T_{\mathcal{G}}}} : \mathbf{r}$ (where all the labels are removed in the two typings above); note that $(t_{T_{\mathcal{G}}}) \widetilde{s'_{T_{\mathcal{G}}}}$ has been obtained by replacing each non-terminal F in $F_{j_{T_{\mathcal{G}}}} \widetilde{s_{T_{\mathcal{G}}}}$ with $F^{\langle k \rangle}$ for some $k \geq T_{\mathcal{G}}$. Thus, by using Theorem 3.7 again, we obtain $(t_{T_{\mathcal{G}}})^{\ell_{T_{\mathcal{G}}}} \widetilde{s'_{T_{\mathcal{G}}}} \overset{u'}{\Longrightarrow}_{\lambda} \mathtt{e}$ for some u' as required.

Now, by applying Lemma 3.11 repeatedly to the reduction sequence (**) (in the backward direction), we obtain:

$$\Gamma_{T_{\mathcal{G}}}(= \{\ell_{T_{\mathcal{G}}} : \tau_{T_{\mathcal{G}}}\}), \ldots, \Gamma_1, \Gamma_0$$

such that:

$$\begin{array}{l} \Gamma_i \vdash (t_i)^{\ell_i} \widetilde{s'_i} : \mathbf{r} \text{ for each } i \in \{0, \ldots, T_{\mathcal{G}}\} \\ (\Gamma_i)_{\mathbf{V}} = \emptyset \text{ for each } i \in \{0, \ldots, T_{\mathcal{G}}\} \\ \ell_j : \tau_j \in \Gamma_i \text{ for every } i, j \text{ such that } 0 \leq i \leq j \leq T_{\mathcal{G}}, \\ \quad \text{where } \tau_i \text{ is the type of } t_i^{\ell_i} \text{ used to derive } \Gamma_i \vdash (t_i)^{\ell_i} \widetilde{s'_i} : \mathbf{r} \end{array}$$

By the pigeonhole principle, there must be duplicated occurrences of the same type binding in:

$$S(= F_{j_0}) : \tau_0, F_{j_1} : \tau_1, \ldots, F_{j_{T_{\mathcal{G}}}} : \tau_{T_{\mathcal{G}}}.$$

That is, there exist i_1 and i_2 ($\in \{0, \ldots, T_{\mathcal{G}}\}$) such that $j_{i_1} = j_{i_2}$ and $\tau_{i_1} = \tau_{i_2}$. Let $F = F_{j_{i_1}}$ and $\tau = \tau_{i_1} = \sigma_1 \to \cdots \to \sigma_k \to \mathbf{r}$. Then, we have: $\Gamma, \ell_{i_2} : \tau \vdash t_{i_1} : \tau$ for some Γ such that $\Gamma_{\mathbf{V}} = \emptyset$. By Theorem 3.6, for all $s_1, \ldots, s_k$ such that $\emptyset \vdash s_i : \sigma_i$ for each $i \in \{1, \ldots, k\}$, there exist $s'_1, \ldots, s'_k, w'$ such that

$$t_{i_1}\, s_1 \cdots s_k \overset{w'}{\Longrightarrow}_{\lambda} t_{i_2}^{\ell_{i_2}}\, s'_1 \cdots s'_k$$

and $\emptyset \vdash s'_i : \sigma_i$ for each $i \in \{1, \ldots, k\}$. By removing labels, we obtain

$$F^{\langle 2T_{\mathcal{G}} - i_1 \rangle}\, s_1 \cdots s_k \overset{w'}{\Longrightarrow}_{\lambda} F^{\langle 2T_{\mathcal{G}} - i_2 \rangle}\, s'_1 \cdots s'_k.$$

Therefore, we have $\widetilde{s'_1}, \ldots, \widetilde{s'_m}$ such that:

$$\begin{array}{l}
S^{\langle 2T_{\mathcal{G}}\rangle} \xRightarrow{w_1}_{\lambda} F^{\langle 2T_{\mathcal{G}}-i_1\rangle}\, \widetilde{s'_1} \\
F^{\langle 2T_{\mathcal{G}}-i_1\rangle}\, \widetilde{s'_1} \xRightarrow{w_2}_{\lambda} F^{\langle 2T_{\mathcal{G}}-i_2\rangle}\, \widetilde{s'_2} \\
\cdots \\
F^{\langle 2T_{\mathcal{G}}-i_1\rangle}\, \widetilde{s'_{m-1}} \xRightarrow{w_m}_{\lambda} F^{\langle 2T_{\mathcal{G}}-i_2\rangle}\, \widetilde{s'_m} \\
\emptyset \vdash F^{\langle 2T_{\mathcal{G}}-i_2\rangle}\, \widetilde{s'_m} : \mathbf{r} \text{ (for every } m \geq 1)
\end{array}$$

From the last condition and Theorem 3.7, we also have $F^{\langle 2T_{\mathcal{G}}-i_2\rangle}\, \widetilde{s'_m} \xRightarrow{v_m}_{\lambda} \mathtt{e}$ for some v_m. Thus, we can construct the corresponding reduction sequence of $\mathcal{G}$:

$$S \xRightarrow{w_1}_{\mathcal{G}} F\, \widetilde{s_1} \xRightarrow{w_2}_{\mathcal{G}} F\, \widetilde{s_2} \xRightarrow{w_3}_{\mathcal{G}} \cdots \xRightarrow{w_m}_{\mathcal{G}} F\, \widetilde{s_m} \xRightarrow{v_m}_{\mathcal{G}} \mathtt{e}.$$

By Lemma 4.2, we can assume that the depth of unfoldings of F in the last step $F\, \widetilde{s_m} \xRightarrow{v_m}_{\mathcal{G}} \mathtt{e}$ is at most $T_{\mathcal{G}}$. Thus, if $m_2 \geq m_1 + c_2 = m_1 + T_{\mathcal{G}} + 1$, then $F\, \widetilde{s_{m_1}} \xRightarrow{v_{m_1}}_{\mathcal{G}} \mathtt{e}$ and $F\, \widetilde{s_{m_1}} \xRightarrow{w_{m_1+1}}_{\mathcal{G}} \cdots \xRightarrow{w_{m_2}}_{\mathcal{G}} F\, \widetilde{s_{m_2}} \xRightarrow{v_{m_2}}_{\mathcal{G}} \mathtt{e}$ must be different reduction sequences; thus, we have $w_1 \cdots w_{m_1} v_{m_1} \neq w_1 \cdots w_{m_2} v_{m_2}$.

Finally, to bound $|w_1 \cdots w_m v_m|$, we construct terms $t'_i (0 \leq i \leq m)$ by:

- $t'_m = F^{\langle 2T_{\mathcal{G}}-i_2\rangle}$; and
- $t'_i (0 \leq i \leq m-1)$ is the term obtained from $F^{\langle 2T_{\mathcal{G}}-i_1\rangle}$ by replacing the (single) occurrence of $F^{\langle 2T_{\mathcal{G}}-i_2\rangle}$ corresponding to the head of $F^{\langle 2T_{\mathcal{G}}-i_2\rangle}\, \widetilde{s'_{i+1}}$, with t'_{i+1}.

Then, we have:

$$t'_0 \xRightarrow{w_1}_{\lambda} t'_1\, \widetilde{s''_1} \xRightarrow{w_2}_{\lambda} t'_2\, \widetilde{s''_2} \xRightarrow{w_3}_{\lambda} \cdots \xRightarrow{w_m}_{\lambda} t'_m\, \widetilde{s''_m} \xRightarrow{v_m}_{\lambda} \mathtt{e}$$

for some $\widetilde{s''_1}, \ldots, \widetilde{s''_m}$. The size $|t'_0|$ is estimated by:

$$\begin{array}{rcl}
|t'_0| & \leq & |F^{\langle 2T_{\mathcal{G}}-i_1\rangle}| + |t'_1| \\
& \leq & |F^{\langle 2T_{\mathcal{G}}-i_1\rangle}| + |F^{\langle 2T_{\mathcal{G}}-i_1\rangle}| + |t'_2| \leq \cdots \\
& \leq & m|F^{\langle 2T_{\mathcal{G}}-i_1\rangle}| + |t'_m| = (m+1)|F^{\langle 2T_{\mathcal{G}}-i_1\rangle}| \\
& \leq & (m+1)B_{\mathcal{G}}^{2T_{\mathcal{G}}} = (m+1)c_1^2
\end{array}$$

Therefore, by Theorem 3.5, we have

$$|w_1 \cdots w_m v_m| \leq \mathbf{exp}_{n-1}((m+1)c_1^2)$$

as required. ■

Example 4.3: We demonstrate the constructions in the proof above for $\mathcal{G}_2$ in Example 2.2. For the sake of simplicity, we treat $T_{\mathcal{G}_2}$ as if it were 2. The non-terminal *Two* is abbreviated to T below. Consider $w = (\mathtt{a}, 1)(\mathtt{a}, 2)(\mathtt{b}, 1)^2$, and the following reduction sequence that generates it:

$$\begin{array}{l}
(S^{\ell_0}, \emptyset) \xrightarrow{\epsilon}_{\mathcal{G}_2} (F^{\ell_1}\, \mathtt{b}, \prec_1) \xRightarrow{\epsilon}_{\mathcal{G}_2} \\
(\mathtt{a}\, (F^{\ell_2}\, (T^{\ell_2}\, \mathtt{b}))\, (\mathtt{b}\, \mathtt{e}), \prec_2) \xrightarrow{(\mathtt{a},1)}_{\mathcal{G}_2} (F^{\ell_2}\, (T^{\ell_2}\, \mathtt{b}), \prec_2) \xRightarrow{\epsilon}_{\mathcal{G}_2} \\
(\mathtt{a}\, (F^{\ell_3}\, (T^{\ell_3} (T^{\ell_2}\, \mathtt{b})))\, (T^{\ell_2}\, \mathtt{b}\, \mathtt{e}), \prec_3) \xrightarrow{(\mathtt{a},2)}_{\mathcal{G}_2} \\
(T^{\ell_2}\, \mathtt{b}\, \mathtt{e}, \prec_3) \xRightarrow{(\mathtt{b},1)^2}_{\mathcal{G}_2} (\mathtt{e}, \prec_4).
\end{array}$$

The above sequence is simulated by the following reduction sequence:

$$(t_0^{\ell_0}, \emptyset) \xRightarrow{\epsilon}_{\lambda} (t_1^{\ell_1}\, \mathtt{b}, \prec_1) \xRightarrow{(\mathtt{a},1)}_{\lambda} (t_2^{\ell_2}\, (T^{\langle 2\rangle}\, \mathtt{b}), \prec_2) \xRightarrow{u'}_{\lambda} (\mathtt{e}, \prec_4).$$

Here, t_0, t_1, and t_2 are:

$$t_2 = F^{\langle 2\rangle} \quad t_1 = \lambda f.\mathtt{a}\, (t_2^{\ell_2}\, (T^{\langle 2\rangle}\, f))\, (f\, \mathtt{e}) \quad t_0 = t_1^{\ell_1} \mathtt{b}$$

Note that except labels, t_i is identical to ${F_{j_i}}^{\langle 4-i\rangle}$ where F_{j_i} is S or F. In particular, t_0 is the same as $S^{\langle 4\rangle}$. The label u' for the last step can be either $(\mathtt{a}, 2)(\mathtt{b}, 1)^2$ or $(\mathtt{a}, 1)(\mathtt{a}, 2)(\mathtt{b}, 1)^4$; whichever is fine for the construction below.

By applying Lemma 3.11 repeatedly to the sequence above in the backward direction, we obtain:

$$\begin{array}{ll}
\ell_2 : \tau \vdash t_2^{\ell_2}\, (T^{\langle 2\rangle}\, \mathtt{b}) : \mathbf{r} & \quad \ell_1 : \tau, \ell_2 : \tau \vdash t_1^{\ell_1}\, \mathtt{b} : \mathbf{r} \\
\ell_0 : \mathbf{r}, \ell_1 : \tau, \ell_2 : \tau \vdash t_0^{\ell_0} : \mathbf{r} &
\end{array}$$

for $\tau = (\mathbf{r} \to \mathbf{r}) \to \mathbf{r}$. By Theorem 3.6 and $\ell_1 : \tau, \ell_2 : \tau \vdash t_1^{\ell_1}\, \mathtt{b} : \mathbf{r}$, for every s such that $\emptyset \vdash s : \mathbf{r} \to \mathbf{r}$, there exists s' such that $t_1\, s \xRightarrow{w'}_{\lambda} t_2^{\ell_2}\, s'$ and $\emptyset \vdash s' : \mathbf{r} \to \mathbf{r}$. By removing the labels, we obtain:

$$F^{\langle 3\rangle}\, s \xRightarrow{w'}_{\lambda} F^{\langle 2\rangle}\, s'.$$

Thus, we have $s'_1 (= \mathtt{b}), s'_2 (= T^{\langle 2\rangle}\, \mathtt{b}), s'_3, \ldots, s'_m$ and $w_1 (= \epsilon), w_2 (= (\mathtt{a}, 1)), w_3, \ldots, w_m$ such that:

$$\begin{array}{l}
S^{\langle 4\rangle} \xRightarrow{w_1}_{\lambda} F^{\langle 3\rangle}\, s'_1 \xRightarrow{w_2}_{\lambda} F^{\langle 2\rangle}\, s'_2 \\
F^{\langle 3\rangle}\, s'_2 \xRightarrow{w_3}_{\lambda} F^{\langle 2\rangle}\, s'_3 \quad \cdots \quad F^{\langle 3\rangle}\, s'_{m-1} \xRightarrow{w_m}_{\lambda} F^{\langle 2\rangle}\, s'_m \\
\emptyset \vdash F^{\langle 2\rangle}\, s'_m : \mathbf{r}
\end{array}$$

From the last condition and Theorem 3.7, we also have $F^{\langle 2\rangle}\, s'_m \xRightarrow{v_m}_{\lambda} \mathtt{e}$ for some v_m. Thus, we can construct the corresponding reduction sequence:

$$S \xRightarrow{w_1}_{\mathcal{G}_2} F\, s_1 \xRightarrow{w_2}_{\mathcal{G}_2} F\, s_2 \xRightarrow{w_3}_{\mathcal{G}_2} \cdots \xRightarrow{w_m}_{\mathcal{G}_2} F\, s_m \xRightarrow{v_m}_{\mathcal{G}_2} \mathtt{e}.$$

To bound $|w_1 \cdots w_m v_m|$, let $t'_i (0 \leq i \leq m)$ be:

$$\begin{array}{l}
t'_m = F^{\langle 2\rangle} \qquad t'_0 = t'_1\, \mathtt{b} \\
t'_i = \lambda f.\mathtt{a}\, (t'_{i+1}\, (T^{\langle 2\rangle}\, f))\, (f\, \mathtt{e}) \text{ (for each } i \in \{1, \ldots, m-1\})
\end{array}$$

Then we have

$$t'_0 \xRightarrow{w_1}_{\lambda} t'_1\, s''_1 \xRightarrow{w_2}_{\lambda} t'_2\, s''_2 \xRightarrow{w_3}_{\lambda} \cdots \xRightarrow{w_m}_{\lambda} t'_m\, s''_m \xRightarrow{v_m}_{\lambda} \mathtt{e}$$

and the size $|t'_0|$ is bounded by $m \times |\lambda f.\mathtt{a}\, (F\, (T^{\langle 2\rangle}\, f))\, (f\, \mathtt{e})| + |t'_m| < (m+1)B_{\mathcal{G}_2}^3$. Thus, by Theorem 3.5, we have a required bound for $|w_1 \cdots w_m v_m|$. □

V. Related Work

As mentioned in Section I, higher-order grammars have been actively studied [9], [15], [20], [22], [27], [30] but there are not so many results on pumping lemmas. Hayashi [11] proved a pumping lemma for indexed languages (which correspond to order-2 HORS for word languages). Our pumping lemma (Theorem 2.1) is weaker than Hayashi's lemma for the order-2 case, in that ours does not state the strict increase of the length $|w_1 \cdots w_m v_m \uparrow_{\mathtt{br}}|$. Because of this weakness, Theorem 2.1 cannot be used to separate the classes of *word languages* generated by HORS (recall Remark 2.1). Strengthening Theorem 2.1 is left for future work.

Kartzow and Parys [12], [13], [25], [24] have recently shown pumping lemmas for higher-order pushdown automata, and proved various properties of (collapsible) higher-order pushdown automata.[5] The present paper is closest to [13],

[5] In prior to their work, Blumensath [6] presented a pumping lemma for higher-order pushdown automata, but his proof contained a flaw [25].

and provides an alternative proof of the main result of [13]. As mentioned in Section I, their reasoning is for collapsible higher-order pushdown automata (CPDA) rather than for grammars, and complex probably partly due to the complexity of CPDA. They also use the notion of *types* for capturing certain properties of configurations of CPDA, but the correspondence between their notion of types and ours is unclear and worthy of further investigation.

Damm [9] showed that the classes of word languages generated by higher-order grammars form an infinite hierarchy.[6] He used a complexity measure of languages called the *rational index* instead of a pumping lemma.[7] Damm's technique can actually be used to show that the classes of word languages generated by (unsafe) HORS form an infinite hierarchy (but it is not strong enough to show that the hierarchy is *strict* in the sense that there is a word language that is generated by an order-$(n+1)$ HORS but not by any order-n HORS).

Our intersection type system is non-standard in the sense that weakening is disallowed but contraction (or, the idempotency of intersection types) is allowed, while previous standard intersection type systems [3], [29], [14] either allow or disallow both. The need for disallowing weakening is clear from the example at the beginning of Section III. The idempotency of intersection types is required to ensure that the number of possible intersection types for each non-terminal is finite.

VI. Conclusion

We have shown a pumping lemma for higher-order recursion schemes, by using a novel intersection type system for the λ-calculus to reason about reductions. Our proof is arguably much simpler than Kartzow and Parys's proof for the pumping lemma for collapsible pushdown automata.

Acknowledgment: This work is partially supported by Kakenhi 21220001 and the Mitsubishi Foundation. We would like to thank Christopher Broadbent, Kazutaka Matsuda and anonymous referees for useful comments.

References

[1] Y. Bar-Hillel, M. A. Perles, and E. Shamir. On formal properties of simple phrase structure grammars. *Z. Phonetik Sprachwiss. und Kommunikat.*, (14):143–172, 1961.

[2] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *J. Symb. Log.*, 48(4):931–940, 1983.

[3] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *J. Symb. Log.*, 48(4):931–940, 1983.

[4] H. P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*. North Holland, 1985.

[5] A. Beckmann. Exact bounds for lengths of reductions in typed lambda-calculus. *J. Symb. Log.*, 66(3):1277–1285, 2001.

[6] A. Blumensath. On the structure of graphs in the Caucal hierarchy. *Theor. Comput. Sci.*, 400(1-3):19–45, 2008.

[7] A. Carayol and O. Serre. Collapsible pushdown automata and labeled recursion schemes: Equivalence, safety and effective selection. In *Proceedings of LICS 2012*, pages 165–174. IEEE, 2012.

[8] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Functional characters of solvable terms. *Mathematical Logic Quarterly*, 27(2-6):45–58, 1981.

[9] W. Damm. The IO- and OI-hierarchies. *Theor. Comput. Sci.*, 20:95–207, 1982.

[10] M. Hague, A. Murawski, C.-H. L. Ong, and O. Serre. Collapsible pushdown automata and recursion schemes. In *Proceedings of LICS 2008*, pages 452–461. IEEE Computer Society, 2008.

[11] T. Hayashi. On derivation trees of indexed grammars –an extension of the uvwxy-theorem–. *Publ. RIMS, Kyoto Univ.*, pages 61–92, 1973.

[12] A. Kartzow. A pumping lemma for collapsible pushdown graphs of level 2. In *Proceedings of CSL 2011*, volume 12 of *LIPIcs*, pages 322–336, 2011.

[13] A. Kartzow and P. Parys. Strictness of the collapsible pushdown hierarchy. In *Proceedings of MFCS 2012*, volume 7464 of *LNCS*, pages 566–577. Springer, 2012.

[14] A. J. Kfoury and J. B. Wells. Principality and type inference for intersection types using expansion variables. *Theor. Comput. Sci.*, 311(1-3):1–70, 2004.

[15] T. Knapik, D. Niwinski, and P. Urzyczyn. Higher-order pushdown trees are easy. In *FoSSaCS 2002*, volume 2303 of *LNCS*, pages 205–222. Springer-Verlag, 2002.

[16] N. Kobayashi. Types and higher-order recursion schemes for verification of higher-order programs. In *Proc. of POPL*, pages 416–428, 2009.

[17] N. Kobayashi. A practical linear time algorithm for trivial automata model checking of higher-order recursion schemes. In *Proceedings of FoSSaCS 2011*, volume 6604 of *LNCS*, pages 260–274. Springer-Verlag, 2011.

[18] N. Kobayashi. Pumping by typing. An extended version, available from the author's web page, 2013.

[19] N. Kobayashi, R. Sato, and H. Unno. Predicate abstraction and CEGAR for higher-order model checking. In *Proc. of PLDI*, pages 222–233, 2011.

[20] A. N. Maslov. The hierarchy of indexed languages of an arbitrary level. *Soviet Math. Dokl.*, 15(4):1170–1174, 1974.

[21] R. P. Neatherway, S. J. Ramsay, and C.-H. L. Ong. A traversal-based algorithm for higher-order model checking. In *ACM SIGPLAN International Conference on Functional Programming (ICFP '12)*, pages 353–364, 2012.

[22] C.-H. L. Ong. On model-checking trees generated by higher-order recursion schemes. In *LICS 2006*, pages 81–90. IEEE Computer Society Press, 2006.

[23] C.-H. L. Ong and S. Ramsay. Verifying higher-order programs with pattern-matching algebraic data types. In *Proc. of POPL*, pages 587–598, 2011.

[24] P. Parys. On the significance of the collapse operation. In *Proceedings of LICS 2012*, pages 521–530, 2012.

[25] P. Parys. A pumping lemma for pushdown graphs of any level. In *Proceedings of STACS 2012*, volume 14 of *LIPIcs*, pages 54–65. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[26] K. Terui. Semantic evaluation, intersection types and complexity of simply typed lambda calculus. In *23rd International Conference on Rewriting Techniques and Applications (RTA'12)*, volume 15 of *LIPIcs*, pages 323–338, 2012.

[27] R. Turner. An infinite hierarchy of term languages - an approach to mathematical complexity. In *Proceedings of ICALP*, pages 593–608, 1972.

[28] S. van Bakel. Complete restrictions of the intersection type discipline. *Theor. Comput. Sci.*, 102(1):135–163, 1992.

[29] S. van Bakel. Intersection type assignment systems. *Theor. Comput. Sci.*, 151(2):385–435, 1995.

[30] M. Wand. An algebraic formulation of the Chomsky hierarchy. In *Category Theory Applied to Computation and Control*, volume 25 of *LNCS*, pages 209–213. Springer-Verlag, 1974.

[6] He considered two hierarchies of word languages, IO_n and OI_n, and showed $IO_n \subsetneq IO_{n+1}$ and $OI_n \subsetneq OI_{2n+1}$. OI_n corresponds to the word languages generated by order-n *safe* HORS [15].

[7] He writes: "We note, that an approach to the hierarchy question ... would require the proof of (some sort of) pumping lemmata ..., which seem to be extremely hard to prove." [9].