Pumping Lemma for Higher-order Languages

Kazuyuki Asada[1] and Naoki Kobayashi[1]

1 **The University of Tokyo**

Abstract

We study a pumping lemma for the word/tree languages generated by higher-order grammars. Pumping lemmas are known up to order-2 word languages (i.e., for regular/context-free/indexed languages), and have been used to show that a given language does not belong to the classes of regular/context-free/indexed languages. We prove a pumping lemma for word/tree languages of arbitrary orders, modulo a conjecture that a higher-order version of Kruskal's tree theorem holds. We also show that the conjecture indeed holds for the order-2 case, which yields a pumping lemma for order-2 tree languages and order-3 word languages.

1998 ACM Subject Classification F.4.3 Formal Languages

Keywords and phrases pumping lemma, higher-order grammars, Kruskal's tree theorem

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

We study a pumping lemma for higher-order languages, i.e., the languages generated by higher-order word/tree grammars where non-terminals can take higher-order functions as parameters. The classes of higher-order languages [26, 18, 4, 5, 6] form an infinite hierarchy, where the classes of order-0, order-1, and order-2 languages are those of regular, context-free and indexed languages. Higher-order grammars and languages have been extensively studied by Damm [4] and Engelfriet [5, 6] and recently re-investigated in the context of model checking and program verification [9, 20, 15, 24, 11, 16, 12, 23].

Pumping lemmas [2, 7] are known up to order-2 word languages, and have been used to show that a given language does not belong to the classes of regular/context-free/indexed languages. To our knowledge, however, little is known about languages of order-3 or higher. Pumping lemmas [21, 12] are also known for higher-order *deterministic* grammars (as generators of infinite *trees*, rather than tree languages), but they cannot be applied to non-deterministic grammars.

In the present paper, we state and prove a pumping lemma for unsafe[1] languages of arbitrary orders modulo an assumption that a "higher-order version" of Kruskal's tree theorem [17, 19] holds. Let $\preceq$ be the homeomorphic embedding on finite ranked trees[2], and $\prec$ be the strict version of $\preceq$. The statement of our pumping lemma[3] is that for any order-n infinite tree language L, there exist a constant c and a strictly increasing infinite sequence of trees $T_0 \prec T_1 \prec T_2 \prec \cdots$ in L such that $|T_i| \leq \mathbf{exp}_n(ci)$ for every $i \geq 0$, where $\mathbf{exp}_0(x) = x$ and $\mathbf{exp}_{n+1}(x) = 2^{\mathbf{exp}_n(x)}$. Due to the correspondence between word/tree languages [4, 1],

[1] See, e.g., [16] for the distinction between safe vs unsafe languages; the class of unsafe languages subsumes that of safe languages.
[2] I.e., $T_1 \preceq T_2$ if there exists an injective map from the nodes of T_1 to those of T_2 that preserves the labels of nodes and the ancestor/descendant-relation of nodes; see Section 2 for the precise definition.
[3] This should perhaps be called a pumping "conjecture" since it relies on the conjecture of the higher-order Kruskal's tree theorem.

Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–68
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

it also implies that for any order-n infinite word language L (where $n \geq 1$), there exist a constant c and a strictly increasing infinite sequence of words $w_0 \prec w_1 \prec w_2 \prec \cdots$ in L such that $|w_i| \leq \mathbf{exp}_{n-1}(ci)$ for every $i \geq 0$, where $\prec$ is the subsequence relation. The pumping lemma can be used, for example, to show (modulo the conjecture) that the order-$(n+1)$ language $\{a^{\mathbf{exp}_n(k)} \mid k \geq 0\}$ does not belong to the class of order-n word languages, for $n > 0$. Thus the lemma would also provide an alternative proof of the strictness of the hierarchy of the classes of higher-order languages.[4]

We now informally explain the assumption of "higher-order Kruskal's tree theorem" (see Section 2 for details). Kruskal's tree theorem [17, 19] states that the homeomorphic embedding $\preceq$ is a well-quasi order, i.e., that for any infinite sequence of trees $T_0, T_1, T_2, \ldots$, there exist $i < j$ such that $T_i \preceq T_j$. The homeomorphic embedding $\preceq$ can be naturally lifted (e.g. via the logical relation) to a family of relations $(\preceq_\kappa)_\kappa$ on higher-order tree functions of type κ. Our conjecture of "higher-order Kruskal's theorem" states that, for every simple type κ, $\preceq_\kappa$ is also a well-quasi order on the functions expressed by the simply-typed λ-terms. We prove that the conjecture indeed holds up to order-2 functions, if we take $\preceq_\kappa$ as the logical relation induced from the homeomorphic embedding $\preceq$. Thus, our pumping "lemma" is indeed true for order-2 tree languages and order-3 word languages. To our knowledge, the pumping lemma for those languages is novel. The conjecture remains open for order-3 or higher, which should be of independent interest.

Our proof of the pumping lemma (modulo the conjecture) uses the recent work of Parys [23] on an intersection type system for deciding the infiniteness of the language generated by a given higher-order grammar, and our previous work on the relationship between higher-order word/tree languages [1].

The rest of this paper is organized as follows. Section 2 prepares several definitions and states our pumping lemma and the conjecture more formally. Section 3 derives some corollaries of Parys' result [23]. Section 4 prepares a simplified and specialized version of our previous result [1]. Using the results in Sections 3 and 4, we prove our pumping lemma (modulo the conjecture) in Section 5. Section 6 proves the conjecture on higher-order Kruskal's tree theorem for the order-2 case, by which we obtain the (unconditional) pumping lemma for order-2 tree languages and order-3 word languages. Section 7 discusses related work and Section 8 concludes.

2 Preliminaries

We first give basic definitions needed for explaining our main theorem. We then state the main theorem and provide an overview of its proof.

2.1 λ-terms and Higher-order Grammars

This section gives basic definitions for terms and higher-order grammars.

▶ **Definition 1** (types and terms)**.** The set of *simple types*, ranged over by κ, is given by: $\kappa ::= \mathtt{o} \mid \kappa_1 \to \kappa_2$. The order of a simple type κ, written $\mathtt{order}(\kappa)$ is defined by $\mathtt{order}(\mathtt{o}) = 0$ and $\mathtt{order}(\kappa_1 \to \kappa_2) = \max(\mathtt{order}(\kappa_1) + 1, \mathtt{order}(\kappa_2))$. The type $\mathtt{o}$ describes trees, and

[4] The strictness of the hierarchy of higher-order *safe* languages has been shown by Engelfriet [5] using a complexity argument, and Kartzow [8] observed that essentially the same argument is applicable to obtain the strictness of the hierarchy of unsafe languages as well. Their argument cannot be used for showing that a particular language does not belong to the class of order-n languages.

$\kappa_1 \to \kappa_2$ describes functions from κ_1 to κ_2. The set of $\lambda^{\to,+}$-*terms* (or *terms*), ranged over by s, t, u, v, is defined by:

$$t ::= x \mid a\, t_1 \, \cdots \, t_k \mid t_1\, t_2 \mid \lambda x : \kappa.t \mid t_1 + t_2$$

Here, x ranges over variables, and a over constants (which represent tree constructors). Variables are also called *non-terminals*, ranged over by x, y, z, f, g, A, B; and constants are also called *terminals*. A ranked alphabet Σ is a map from a finite set of terminals to natural numbers called *arities*; we implicitly assume a ranked alphabet whose domain contains all terminals discussed, unless explicitly described. $+$ is non-deterministic choice. As seen below, our simple type system forces that a terminal must be fully applied; this does not restrict the expressive power, as $\lambda x_1, \dots, x_k.a\, x_1 \cdots x_k$ is available. We often omit the type κ of $\lambda x : \kappa.t$. A term is called an *applicative term* if it does not contain λ-abstractions nor $+$, and called a $\lambda^{\to}$-*term* if it does not contain $+$. As usual, we identify terms up to the α-equivalence, and implicitly apply α-conversions.

A (simple) type environment $\mathcal{K}$ is a sequence of type bindings of the form $x : \kappa$ such that if $\mathcal{K}$ contains $x : \kappa$ and $x' : \kappa'$ in different positions then $x \neq x'$. In type environments, non-terminals are also treated as variables. A term t has type κ under $\mathcal{K}$ if $\mathcal{K} \vdash_{\mathtt{ST}} t : \kappa$ is derivable from the following typing rules.

$$\frac{}{\mathcal{K},\, x : \kappa,\, \mathcal{K}' \vdash_{\mathtt{ST}} x : \kappa} \qquad \frac{\Sigma(a) = k \qquad \mathcal{K} \vdash_{\mathtt{ST}} t_i : \mathtt{o} \quad (\text{for each } i \in \{1, \dots, k\})}{\mathcal{K} \vdash_{\mathtt{ST}} a\, t_1 \cdots t_k : \mathtt{o}}$$

$$\frac{\mathcal{K} \vdash_{\mathtt{ST}} t_1 : \kappa_2 \to \kappa \qquad \mathcal{K} \vdash_{\mathtt{ST}} t_2 : \kappa_2}{\mathcal{K} \vdash_{\mathtt{ST}} t_1\, t_2 : \kappa} \quad \frac{\mathcal{K},\, x : \kappa_1 \vdash_{\mathtt{ST}} t : \kappa_2}{\mathcal{K} \vdash_{\mathtt{ST}} \lambda x : \kappa_1.t : \kappa_1 \to \kappa_2} \quad \frac{\mathcal{K} \vdash_{\mathtt{ST}} t_1 : \mathtt{o} \qquad \mathcal{K} \vdash_{\mathtt{ST}} t_2 : \mathtt{o}}{\mathcal{K} \vdash_{\mathtt{ST}} t_1 + t_2 : \mathtt{o}}$$

We consider below only well-typed terms. Note that given $\mathcal{K}$ and t, there exists at most one type κ such that $\mathcal{K} \vdash_{\mathtt{ST}} t : \kappa$. We call κ the type of t (with respect to $\mathcal{K}$). We often omit "with respect to $\mathcal{K}$" if $\mathcal{K}$ is clear from context. The (internal) *order* of t, written $\mathtt{order}_{\mathcal{K}}(t)$, is the largest order of the types of subterms of t, and the *external order* of t, written $\mathtt{eorder}_{\mathcal{K}}(t)$, is the order of the type of t (both with respect to $\mathcal{K}$). We often omit $\mathcal{K}$ when it is clear from context. For example, for $t = (\lambda x : \mathtt{o}.x)\mathtt{e}$, $\mathtt{order}_{\emptyset}(t) = 1$ and $\mathtt{eorder}_{\emptyset}(t) = 0$.

We call a term t *ground* (with respect to $\mathcal{K}$) if $\mathcal{K} \vdash_{\mathtt{ST}} t : \mathtt{o}$. We call t a (finite, Σ-ranked) *tree* if t is a closed ground applicative term consisting of only terminals. We write $\mathbf{Tree}_{\Sigma}$ for the set of Σ-ranked trees, and use the meta-variable π for trees.

The set of *contexts*, ranged over by C, D, G, H, is defined by $C ::= [\,] \mid C\,t \mid t\,C \mid \lambda x.C$. We write $C[t]$ for the term obtained from C by replacing $[\,]$ with t. Note that the replacement may capture variables; e.g., $(\lambda x.[\,])[x]$ is $\lambda x.x$. We call C a $(\mathcal{K}', \kappa')$-$(\mathcal{K}, \kappa)$-*context* if $\mathcal{K} \vdash_{\mathtt{ST}} C : \kappa$ is derived by using axiom $\mathcal{K}' \vdash_{\mathtt{ST}} [\,] : \kappa'$. We also call a $(\emptyset, \kappa')$-$(\emptyset, \kappa)$-context a κ'-κ-*context*. The (internal) *order* of a $(\mathcal{K}', \kappa')$-$(\mathcal{K}, \kappa)$-context, is the largest order of the types occurring in the derivation of $\mathcal{K} \vdash_{\mathtt{ST}} C : \kappa$. A context is called a $\lambda^{\to}$-*context* if it does not contain $+$.

We define the *size* $|t|$ of a term t by: $|x| := 1$, $|a\, t_1 \, \cdots, t_k| := 1 + |t_1| + \cdots + |t_k|$, $|s\, t| := |s| + |t| + 1$, $|\lambda x.t| := |t| + 1$, and $|s + t| := |s| + |t| + 1$. The size $|C|$ of a context C is defined similarly, with $|[\,]| := 0$.

▶ **Definition 2** (reduction and language)**.** The set of *(call-by-name) evaluation contexts* is defined by:

$$E ::= [\,]\, t_1 \cdots t_k \mid a\, \pi_1 \cdots \pi_i\, E\, t_1 \cdots t_k$$

and the *call-by-name reduction* for (possibly open) ground terms is defined by:

$$E[(\lambda x.t)t'] \longrightarrow E[[t'/x]t] \qquad\qquad E[t_1 + t_2] \longrightarrow E[t_i] \quad (i = 1, 2)$$

where $[t'/x]t$ is the usual capture-avoiding substitution. We write $\longrightarrow^*$ for the reflexive transitive closure of $\longrightarrow$. A *call-by-name normal form* is a ground term t such that $t \not\longrightarrow t'$ for any t'. For a closed ground term t, we define the *tree language* $\mathcal{L}(t)$ *generated by* t by $\mathcal{L}(t) := \{\pi \mid t \longrightarrow^* \pi\}$. For a closed ground $\lambda^{\to}$-term t, $\mathcal{L}(t)$ is a singleton set $\{\pi\}$; we write $\mathcal{T}(t)$ for such π and call it *the tree of* t.

Note that $t \longrightarrow^* t'$ implies $[s/x]t \longrightarrow^* [s/x]t'$, and that the set of call-by-name normal forms equals the set of trees and ground terms of the form $E[x]$.

For $x : \kappa \vdash_{\mathtt{ST}} t : \mathtt{o}$ where t does not contain the non-deterministic choice, t is called *linear* (with respect to x) if x occurs exactly once in the call-by-name normal form of t. A pair of contexts $[\,] : \kappa \vdash_{\mathtt{ST}} C : \mathtt{o}$ and $[\,] : \kappa \vdash_{\mathtt{ST}} D : \kappa$ is called *linear* if $x : \kappa \vdash_{\mathtt{ST}} C[D^i[x]] : \mathtt{o}$ is linear for any $i \geq 0$ where x is a fresh variable that is not captured by the context applications.

▶ **Definition 3** (higher-order grammar)**.** A *higher-order grammar* (or *grammar* for short) is a quadruple $(\Sigma, \mathcal{N}, \mathcal{R}, S)$, where (i) Σ is a ranked alphabet; (ii) $\mathcal{N}$ is a map from a finite set of non-terminals to their types; (iii) $\mathcal{R}$ is a finite set of *rewriting rules* of the form $A \to \lambda x_1.\cdots\lambda x_\ell.t$, where $\mathcal{N}(A) = \kappa_1 \to \cdots \to \kappa_\ell \to \mathtt{o}$, t is an applicative term, and $\mathcal{N}, x_1 : \kappa_1, \ldots, x_\ell : \kappa_\ell \vdash_{\mathtt{ST}} t : \mathtt{o}$ holds; (iv) S is a non-terminal called the *start symbol*, and $\mathcal{N}(S) = \mathtt{o}$. The *order* of a grammar $\mathcal{G}$ is the largest order of the types of non-terminals. We sometimes write $\Sigma_\mathcal{G}, \mathcal{N}_\mathcal{G}, \mathcal{R}_\mathcal{G}, S_\mathcal{G}$ for the four components of $\mathcal{G}$. We often write $A\,x_1\,\cdots\,x_k \to t$ for the rule $A \to \lambda x_1.\cdots\lambda x_k.t$.

For a grammar $\mathcal{G} = (\Sigma, \mathcal{N}, \mathcal{R}, S)$, the rewriting relation $\longrightarrow_\mathcal{G}$ is defined by:

$$\frac{(A \to \lambda x_1.\cdots\lambda x_k.t) \in \mathcal{R}}{A\,t_1\,\cdots\,t_k \longrightarrow_\mathcal{G} [t_1/x_1, \ldots, t_k/x_k]t} \qquad \frac{t_i \longrightarrow_\mathcal{G} t_i' \qquad i \in \{1, \ldots, k\} \qquad \Sigma(a) = k}{a\,t_1\,\cdots\,t_k \longrightarrow_\mathcal{G} a\,t_1\,\cdots\,t_{i-1}\,t_i'\,t_{i+1}\,\cdots\,t_k}$$

We write $\longrightarrow_\mathcal{G}^*$ for the reflexive transitive closure of $\longrightarrow_\mathcal{G}$. The *tree language generated by* $\mathcal{G}$, written $\mathcal{L}(\mathcal{G})$, is the set $\{\pi \mid S \longrightarrow_\mathcal{G}^* \pi\}$.

▶ Remark. An order-n grammar can also be represented as a ground closed order-n $\lambda^{\to,+}$-term extended with the Y-combinator such that $Y_\kappa x.t \longrightarrow [Y_\kappa x.t/x]t$. Conversely, any ground closed order-n $\lambda^{\to,+}$-term (extended with Y) can be represented as an equivalent order-n grammar. We shall make use of this correspondence in Appendix.

The grammars defined above may also be viewed as generators of word languages.

▶ **Definition 4** (word alphabet / br-alphabet)**.** We call a ranked alphabet Σ a *word alphabet* if it has a special nullary terminal $\mathtt{e}$ and all the other terminals have arity 1; also we call a grammar $\mathcal{G}$ a *word grammar* if its alphabet is a word alphabet. For a tree $\pi = a_1(\cdots(a_n\,\mathtt{e})\cdots)$ of a word grammar, we define $\mathbf{word}(\pi) = a_1 \cdots a_n$. The *word language* generated by a word grammar $\mathcal{G}$, written $\mathcal{L}_\mathtt{w}(\mathcal{G})$, is $\{\mathbf{word}(\pi) \mid \pi \in \mathcal{L}(\mathcal{G})\}$.

The frontier word of a tree π, written $\mathbf{leaves}(\pi)$, is the sequence of symbols in the leaves of π. It is defined inductively by: $\mathbf{leaves}(a) = a$ when $\Sigma(a) = 0$, and $\mathbf{leaves}(a\,\pi_1\,\cdots\,\pi_k) = \mathbf{leaves}(\pi_1)\cdots\mathbf{leaves}(\pi_k)$ when $\Sigma(a) = k > 0$. The *frontier language* generated by $\mathcal{G}$, written $\mathcal{L}_\mathtt{leaf}(\mathcal{G})$, is the set: $\{\mathbf{leaves}(\pi) \mid S \longrightarrow_\mathcal{G}^* \pi\}$. A br-*alphabet* is a ranked alphabet such that it has a special binary constant br and a special nullary constant e and the other constants are nullary. We consider e as the empty word ε: for a grammar with a br-alphabet, we also define $\mathcal{L}^{\varepsilon}_{\texttt{leaf}}(\mathcal{G}) := (\mathcal{L}_{\texttt{leaf}}(\mathcal{G}) \setminus \{\texttt{e}\}) \cup \{\varepsilon \mid \texttt{e} \in \mathcal{L}_{\texttt{leaf}}(\mathcal{G})\}$. We call a tree π an e-*free* br-*tree* if it is a tree of some br-alphabet but does not contain e.

We note that the classes of order-0, order-1, and order-2 word languages coincide with those of regular, context-free, and indexed languages, respectively [26].

2.2 Homeomorphic Embedding and Kruskal's Tree Theorem

In our main theorem, we use the notion of homeomorphic embedding for trees.

▶ **Definition 5** (homeomorphic embedding)**.** Let Σ be an arbitrary ranked alphabet. The *homeomorphic embedding* order $\preceq$ between Σ-ranked trees[5] is inductively defined by the following rules:

$$\frac{\pi_i \preceq \pi'_i \quad (\text{for all } i \leq k)}{a\,\pi_1 \cdots \pi_k \preceq a\,\pi'_1 \cdots \pi'_k}(k = \Sigma(a)) \qquad \frac{\pi \preceq \pi_i}{\pi \preceq a\,\pi_1 \cdots \pi_k}(k = \Sigma(a) > 0,\ i \in \{1, \dots, k\})$$

For example, $\texttt{br}\,\texttt{a}\,\texttt{b} \preceq \texttt{br}\,(\texttt{br}\,\texttt{a}\,\texttt{c})\,\texttt{b}$. We extend $\preceq$ to words: for $w = a_1 \cdots a_n$ and $w' = a'_1 \cdots a'_{n'}$, we define $w \preceq w'$ if $a_1(\cdots(a_n(\texttt{e}))) \preceq a'_1(\cdots(a'_{n'}(\texttt{e})))$, where a_i and a'_i are regarded as unary constants and e is a nullary constant (this order on words is nothing but the (scattered) subsequence relation). We write $\pi \prec \pi'$ if $\pi \preceq \pi'$ and $\pi' \not\preceq \pi$.

Next we explain a basic property on $\preceq$, Kruskal's tree theorem. A *quasi-order* (also called a *pre-order*) is a reflexive and transitive relation. A *well quasi-order* on a set S is a quasi-order $\leq$ on S such that for any infinite sequence $(s_i)_i$ of elements in S there exist $j < k$ such that $s_j \leq s_k$.

▶ **Proposition 6** (Kruskal's tree theorem [17])**.** *For any (finite) ranked alphabet Σ, the homeomorphic embedding $\preceq$ on Σ-ranked trees is a well quasi-order.*

2.3 Conjecture and Pumping Lemma for Higher-order Grammars

As explained in Section 1, our pumping lemma makes use of a conjecture on "higher-order" Kruskal's tree theorem, which is stated below.

▶ **Conjecture 7.** *There exists a family $(\preceq_\kappa)_\kappa$ of relations indexed by simple types such that*

- *$\preceq_\kappa$ is a well quasi-order on the set of closed $\lambda^{\to}$-terms of type κ modulo $\beta\eta$-equivalence; i.e., for an infinite sequence $t_1, t_2, \dots$ of closed $\lambda^{\to}$-terms of type κ, there exist $i < j$ such that $t_i \preceq_\kappa t_j$.*
- *$\preceq_\mathsf{o}$ is a conservative extension of $\preceq$, i.e., $t \preceq_\mathsf{o} t'$ if and only if $\mathcal{T}(t) \preceq \mathcal{T}(t')$.*
- *$(\preceq_\kappa)_\kappa$ is closed under applications, i.e., if $t \preceq_{\kappa_1 \to \kappa_2} t'$ and $s \preceq_{\kappa_1} s'$ then $t\,s \preceq_{\kappa_2} t'\,s'$.*

A candidate of $(\preceq_\kappa)_\kappa$ would be the logical relation induced from $\preceq$. Indeed, if we choose the logical relation as $(\preceq_\kappa)_\kappa$, the above conjecture holds up to order-2 (see Theorem 18 in Section 6).

Actually, for our pumping lemma, the following, slightly weaker property called the *periodicity* is sufficient.

▶ **Conjecture 8** (Periodicity)**.** *There exists a family $(\preceq_\kappa)_\kappa$ indexed by simple types such that*

[5] In the usual definition, a quasi order on labels (tree constructors) is assumed. Here we fix the quasi-order on labels to the identity relation.

- $\preceq_\kappa$ *is a quasi-order on the set of closed* $\lambda^{\to}$*-terms of type* κ *modulo* $\beta\eta$*-equivalence.*
- *for any* $\vdash_{\mathtt{ST}} t : \kappa \to \kappa$ *and* $\vdash_{\mathtt{ST}} s : \kappa$*, there exist* $i, j > 0$ *such that*

$$t^i\, s \preceq_\kappa t^{i+j}\, s \preceq_\kappa t^{i+2j}\, s \preceq_\kappa \cdots .$$

- $\preceq_{\mathtt{o}}$ *is a conservative extension of* $\preceq$*.*
- $(\preceq_\kappa)_\kappa$ *is closed under applications.*

Note that Conjecture 7 implies Conjecture 8, since if the former holds, for the infinite sequence $(t^i\, s)_i$, there exist $i < i + j$ such that $t^i\, s \preceq_\kappa t^{i+j}\, s$, and then by the monotonicity of $u \mapsto t^j\, u$, we have $t^{i+kj}\, s \preceq_\kappa t^{i+(k+1)j}\, s$ for any $k \geq 0$.

We can now state our pumping lemma.

▶ **Theorem 9** (pumping lemma)**.** *Assume that Conjecture 8 holds. Then, for any order-*n *tree grammar* $\mathcal{G}$ *such that* $\mathcal{L}(\mathcal{G})$ *is infinite, there exist an infinite sequence of trees* $\pi_0, \pi_1, \pi_2, \ldots \in \mathcal{L}(\mathcal{G})$*, and constants* c*,* d *such that: (i)* $\pi_0 \prec \pi_1 \prec \pi_2 \prec \cdots$*, and (ii)* $|\pi_i| \leq \mathbf{exp}_n(ci + d)$ *for each* $i \geq 0$*. Furthermore, we can drop the assumption on Conjecture 8 when* $\mathcal{G}$ *is of order up to 2.*

By the correspondence between order-n tree grammars and order-$(n+1)$ grammars [4, 1], we also have:

▶ **Corollary 10** (pumping lemma for word languages)**.** *Assume that Conjecture 8 holds. Then, for any order-*n *word grammar* $\mathcal{G}$ *(where* $n \geq 1$*) such that* $\mathcal{L}_{\mathtt{w}}(\mathcal{G})$ *is infinite, there exist an infinite sequence of words* $w_0, w_1, w_2, \ldots \in \mathcal{L}_{\mathtt{w}}(\mathcal{G})$*, and constants* c*,* d *such that: (i)* $w_0 \prec w_1 \prec w_2 \prec \cdots$*, and (ii)* $|w_i| \leq \mathbf{exp}_{n-1}(ci + d)$ *for each* $i \geq 0$*. Furthermore, we can drop the assumption on Conjecture 8 when* $\mathcal{G}$ *is of order up to 3.*

We sketch the overall structure of the proof of Theorem 9 below. Let $\mathcal{G}$ be an order-n tree grammar. By using the recent type system of Parys [23], if $\mathcal{L}(\mathcal{G})$ is infinite, we can construct order-n linear $\lambda^{\to}$-contexts C, D and an order-n $\lambda^{\to}$-term t such that $\{\mathcal{T}(C[D^i[t]]) \mid i \geq 0\}$ $(\subseteq \mathcal{L}(\mathcal{G}))$ is infinite. It then suffices to show that there exist constants p and q such that $\mathcal{T}(C[D^p[t]]) \prec \mathcal{T}(C[D^{p+q}[t]]) \prec \mathcal{T}(C[D^{p+2q}[t]]) \prec \cdots$. The bound $\mathcal{T}(C[D^{p+iq}]) \leq \mathbf{exp}_n(c + id)$ would then follow immediately from the standard result on an upper-bound on the size of β-normal forms. Actually, assuming Conjecture 8, we can easily deduce $\mathcal{T}(C[D^p[t]]) \preceq \mathcal{T}(C[D^{p+q}[t]]) \preceq \mathcal{T}(C[D^{p+2q}[t]]) \preceq \cdots$. Thus, the main remaining difficulty is to show that the "strict" inequality holds periodically. To this end, we prove it by induction on the order, by making use of three ingredients: an extension of the result of Parys' type system (again) [23], an extension of our previous work on a translation from word languages to tree languages [1], and Conjecture 8. In Sections 3 and 4, we derive corollaries from the results of Parys' and our previous work respectively. We then provide the proof of Theorem 9 (except the statement "Furthermore, ...") in Section 5. We then, in Section 6, discharge the assumption on Conjecture 8 for order up to 2, by proving Conjecture 7 for order up to 2.

3 Corollaries of Parys' Results

Parys [23] developed an intersection type system with judgments of the form $\Gamma \vdash s : \tau \triangleright c$, where s is a term of a simply-typed, infinitary λ-calculus (that corresponds to the λY-calculus) extended with choice, and c is a natural number. He proved that for any order-n closed ground term s, (i) $\emptyset \vdash s : \tau \triangleright c$ implies that s can be reduced to a tree π such that $c \leq |\pi|$, and (ii) if s can be reduced to a tree π, then $\emptyset \vdash s : \tau \triangleright c$ holds for some c such that $|\pi| \leq \mathbf{exp}_n(c)$.

Let $\mathcal{G}$ be an order-n tree grammar and S be its start symbol. By Parys' result,[6] if $\mathcal{L}(\mathcal{G})$ is infinite, there exists a derivation for $\emptyset \vdash S : \mathtt{o} \triangleright c_1 + c_2 + c_3$ in which $\Theta \vdash A : \gamma \triangleright c_1 + c_2$ is derived from $\Theta \vdash A : \gamma \triangleright c_1$ for some non-terminal A. Thus, by "pumping" the derivation of $\Theta \vdash A : \gamma \triangleright c_1 + c_2$ from $\Theta \vdash A : \gamma \triangleright c_1$, we obtain a derivation for $\emptyset \vdash S : \mathtt{o} \triangleright c_1 + kc_2 + c_3$ for any $k \geq 0$. From the derivation, we obtain a $\lambda^{\to}$-term t and $\lambda^{\to}$-contexts C, D of at most order-n, such that $C[D^k[t]]$ generates a tree π_k such that $c_1 + kc_2 + c_3 \leq |\pi_k|$. By further refining the argument above (see Appendix A for details), we can also ensure that the pair (C, D) is linear. Thus, we obtain the following lemma.

▶ **Lemma 11.** *Given an order-n tree grammar $\mathcal{G}$ such that $\mathcal{L}(\mathcal{G})$ is infinite, there exist order-n linear $\lambda^{\to}$-contexts C, D, and an order-n $\lambda^{\to}$-term t such that*

1. $\{\mathcal{T}(C[D^k[t]]) \mid k \geq 1\} \subseteq \mathcal{L}(\mathcal{G})$
2. $\{\mathcal{T}(C[D^{\ell_k}[t]]) \mid k \geq 1\}$ *is infinite for any strictly increasing sequence* $(\ell_k)_k$.

By slightly modifying Parys' type system, we can also reason about the length of a particular path of a tree. Let us annotate each constructor a as $a^{\langle i \rangle}$, where $0 \leq i \leq \Sigma(a)$. We call i a *direction*. We define $|\pi|_{\mathrm{p}}$ by:

$$|a^{\langle 0 \rangle}\, \pi_1 \,\cdots\, \pi_k|_{\mathrm{p}} = 1 \qquad |a^{\langle i \rangle}\, \pi_1 \,\cdots\, \pi_k|_{\mathrm{p}} = |\pi_i|_{\mathrm{p}} + 1 \quad (1 \leq i \leq k).$$

We define **rmdir** as the function that removes all the direction annotations.

▶ **Lemma 12.** *For any order-n linear $\lambda^{\to}$-contexts C, D and any order-n $\lambda^{\to}$-term t such that $\{\mathcal{T}(C[D^k[t]]) \mid k \geq 1\}$ is infinite, there exist direction-annotated order-n linear $\lambda^{\to}$-contexts G, H, a direction-annotated order-n $\lambda^{\to}$-term u, and $p, q > 0$ such that*

1. $\mathbf{rmdir}(\mathcal{T}(G[H^k[u]])) = \mathcal{T}(C[D^{pk+q}[t]])$ *for any* $k \geq 1$
2. $\{|\mathcal{T}(G[H^{\ell_k}[u]])|_{\mathrm{p}} \mid k \geq 1\}$ *is infinite for any strictly increasing sequence* $(\ell_k)_k$.

4 Word to Frontier Transformation

We have an "order-decreasing" transformation [1] that transforms an order-$(n+1)$ word grammar $\mathcal{G}$ to an order-n tree grammar $\mathcal{G}'$ (with a `br`-alphabet) such that $\mathcal{L}_{\mathtt{w}}(\mathcal{G}) = \mathcal{L}^{\varepsilon}_{\mathtt{leaf}}(\mathcal{G}')$. We use this as a method for induction on order; this method was originally suggested by Damm [4] for safe languages.

The transformation in the present paper has been modified from the original one in [1]. On the one hand, the current transformation is a specialized version in that we apply the transformation only to $\lambda^{\to}$-terms instead of terms of (non-deterministic) grammars. On the other hand, the current transformation has been strengthened in that the transformation preserves linearity. Due to the preservation of linearity, a *single*-hole context is transformed to a *single*-hole context, and the uniqueness of an occurrence of $[\,]$ will be utilized for the calculation of the size of "pumped trees" in Lemma 16.

The definition of the current transformation is given just by translating the transformation rules in [1] by following the idea of the embedding of $\lambda^{\to}$-terms into grammars. For the detailed definition, see Appendix B. By using this transformation, we have:

▶ **Lemma 13.** *Given order-n $\lambda^{\to}$-contexts C, D, and an order-n $\lambda^{\to}$-term t such that*

[6] See Section 6 of [23]. Parys considered a λ-calculus with infinite regular terms, but the result can be easily adapted to terms of grammars.

- *the constants in* C, D, t *are in a word alphabet,*
- $\{\mathcal{T}(C[D^{\ell_i}[t]]) \mid i \geq 0\}$ *is infinite for any strictly increasing sequence* $(\ell_i)_i$, *and*
- C *and* D *are linear,*

there exist order-$(n-1)$ $\lambda^{\to}$*-contexts* G, H, *order-*$(n-1)$ $\lambda^{\to}$*-term* u, *and some constant numbers* c, $d \geq 1$ *such that*

- *the constants in* G, H, u *are in a* `br`*-alphabet*
- *for* $i \geq 0$, $\mathcal{T}(G[H^i[u]])$ *is either an* `e`*-free* `br`*-tree or* `e`, *and*

$$\mathbf{word}(\mathcal{T}(C[D^{ci+d}[t]])) = \begin{cases} \varepsilon & (\mathcal{T}(G[H^i[u]]) = \mathtt{e}) \\ \boldsymbol{leaves}(\mathcal{T}(G[H^i[u]])) & (\mathcal{T}(G[H^i[u]]) \neq \mathtt{e}) \end{cases}$$

- G *and* H *are linear.*

Proof. The preservation of meaning (the second condition) follows as a corollary of a theorem in [1]. Also, the preservation of linearity (the third condition) can be proved in a manner similar to the proof of the preservation of meaning in [1], using a kind of subject-reduction. See Appendix B for the detail. ☐

5 Proof of the Main Theorem

We first prepare some lemmas.

▶ **Lemma 14.** *For* `e`*-free* `br`*-trees* π *and* π', *if* $\pi \prec \pi'$ *then* $\boldsymbol{leaves}(\pi) \prec \boldsymbol{leaves}(\pi')$.

Proof. We can show that $\pi \preceq \pi'$ implies $\mathbf{leaves}(\pi) \preceq \mathbf{leaves}(\pi')$ and then the statement, both by straightforward induction on the derivation of $\pi \preceq \pi'$. ☐

▶ Remark. The above lemma does not necessarily hold for an arbitrary ranked alphabet, especially that with a unary constant; e.g., `a e` $\prec$ `a (a e)` but their leaves are both `e`. Also, it does not hold if a tree contains `e` and if we regard `e` as ε in the leaves word; e.g., for `br a b` $\prec$ `br (br a e) b`, their leaves are `ab` $\prec$ `aeb`, but if we regard `e` as ε then `ab` $\not\prec$ `ab`.

▶ **Lemma 15.** *For direction-annotated trees* π *and* π', *if* $\pi \prec \pi'$ *then* $\mathbf{rmdir}(\pi) \prec \mathbf{rmdir}(\pi')$.

Proof. We can show that $\pi \preceq \pi'$ implies $\mathbf{rmdir}(\pi) \preceq \mathbf{rmdir}(\pi')$ and then the statement, both by straightforward induction on the derivation of $\pi \preceq \pi'$. ☐

Now, we prove the following lemma (Lemma 16) by the induction on order. Theorem 9 (except the last statement) will then follow as an immediate corollary of Lemmas 11 and 16.

▶ **Lemma 16.** *Assume that the statement of Conjecture 8 is true. For any order-*n *linear* $\lambda^{\to}$*-contexts* C, D *and any order-*n $\lambda^{\to}$*-term* t *such that* $\{\mathcal{T}(C[D^i[t]]) \mid i \geq 1\}$ *is infinite, there exist* c, d, j, $k \geq 1$ *such that*

- $\mathcal{T}(C[D^j[t]]) \prec \mathcal{T}(C[D^{j+k}[t]]) \prec \mathcal{T}(C[D^{j+2k}[t]]) \prec \cdots$
- $|\mathcal{T}(C[D^{j+ik}[t]])| \leq \mathbf{exp}_n(ci+d) \qquad (i = 0, 1, \ldots)$

Proof. The proof proceeds by induction on n. The case $n = 0$ is clear, and we discuss the case $n > 0$ below. By Lemma 12, from C, D, and t, we obtain direction-annotated order-n linear $\lambda^{\to}$-contexts G, H, a direction-annotated order-n $\lambda^{\to}$-term u, and j_0, $k_0 > 0$ such that

$$\mathbf{rmdir}(\mathcal{T}(G[H^i[u]])) = \mathcal{T}(C[D^{j_0+ik_0}[t]]) \text{ for any } i \geq 1 \tag{1}$$

$$\{|\mathcal{T}(G[H^{\ell_i}[u]])|_{\mathrm{p}} \mid i \geq 1\} \text{ is infinite for any strictly increasing sequence } (\ell_i)_i. \tag{2}$$

Next we transform G, H, and u by choosing a path according to directions, i.e., we define G_{p}, H_{p}, and u_{p} as the contexts/term obtained from G, H, and u by replacing each $a^{\langle i\rangle}$ with: (i) $\lambda x_1 \dots x_\ell . a_i x_i$ if $i > 0$ or (ii) $\lambda x_1 \dots x_\ell . \mathtt{e}$ if $i = 0$, where $\ell = \Sigma(a)$ and a_i is a fresh unary constant. For any $i \geq 0$,

$$|\mathcal{T}(G[H^i[u]])|_{\mathrm{p}} = |\mathbf{word}(\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^i[u_{\mathrm{p}}]]))| + 1. \tag{3}$$

We also define a function **path** on trees annotated with directions, by the following induction: $\mathbf{path}(a^{\langle i\rangle}\,\pi_1 \cdots \pi_\ell) = a_i\,\mathbf{path}(\pi_i)$ if $i > 0$ and $\mathbf{path}(a^{\langle 0\rangle}\,\pi_1 \cdots \pi_\ell) = \mathtt{e}$. Then for any $i \geq 0$,

$$\mathbf{path}(\mathcal{T}(G[H^i[u]])) = \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^i[u_{\mathrm{p}}]]). \tag{4}$$

By (2) and (3), $\{\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{\ell_i}[u_{\mathrm{p}}]]) \mid i \geq 0\}$ is infinite for any strictly increasing sequence $(\ell_i)_i$. Also, the transformation from G, H to G_{p}, H_{p} preserves the linearity, because: let N be the normal form of $G[H^i[x]]$ where x is fresh, and N_{p} be the term obtained by applying this transformation to N; then $G_{\mathrm{p}}[{H_{\mathrm{p}}}^i[x]] \longrightarrow^* N_{\mathrm{p}}$, and by the infiniteness of $\{\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^i[u_{\mathrm{p}}]]) \mid i \geq 0\}$, N_{p} must contain x, which implies N_{p} is a linear normal form.

Now we decrease the order by using the transformation in Section 4. By Lemma 13 to G_{p}, H_{p}, and u_{p}, there exist order-$(n-1)$ linear $\lambda^{\to}$-contexts G_l, H_l, an order-$(n-1)$ $\lambda^{\to}$-term u_l, and some constant numbers c', $d' \geq 1$ such that, for any $i \geq 0$, $\mathcal{T}(G_l[{H_l}^i[u_l]])$ is either an $\mathtt{e}$-free $\mathtt{br}$-tree or $\mathtt{e}$, and

$$\mathbf{word}(\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{c'i+d'}[u_{\mathrm{p}}]])) = \begin{cases} \varepsilon & (\mathcal{T}(G_l[{H_l}^i[u_l]]) = \mathtt{e}) \\ \mathbf{leaves}(\mathcal{T}(G_l[{H_l}^i[u_l]])) & (\mathcal{T}(G_l[{H_l}^i[u_l]]) \neq \mathtt{e}). \end{cases} \tag{5}$$

By (2), (3), and (5), $\{\mathcal{T}(G_l[{H_l}^i[u_l]]) \mid i \geq 1\}$ is also infinite.

By the induction hypothesis, there exist j_1 and k_1 such that

$$\mathcal{T}(G_l[{H_l}^{j_1}[u_l]]) \prec \mathcal{T}(G_l[{H_l}^{j_1+k_1}[u_l]]) \prec \mathcal{T}(G_l[{H_l}^{j_1+2k_1}[u_l]]) \prec \cdots.$$

Hence by Lemma 14, we have

$$\mathbf{leaves}(\mathcal{T}(G_l[{H_l}^{j_1}[u_l]])) \prec \mathbf{leaves}(\mathcal{T}(G_l[{H_l}^{j_1+k_1}[u_l]])) \prec \mathbf{leaves}(\mathcal{T}(G_l[{H_l}^{j_1+2k_1}[u_l]])) \prec \cdots.$$

Then by (5), we have

$$\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{c'j_1+d'}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{c'(j_1+k_1)+d'}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{c'(j_1+2k_1)+d'}[u_{\mathrm{p}}]]) \prec \cdots.$$

Let $j'_1 = c'j_1 + d'$ and $k'_1 = c'k_1$; then

$$\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j'_1}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j'_1+k'_1}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j'_1+2k'_1}[u_{\mathrm{p}}]]) \prec \cdots. \tag{6}$$

Now, by Conjecture 8, there exist $j_2 \geq 0$ and $k_2 > 0$ such that

$$H^{j_2}[u] \preceq_\kappa H^{j_2+k_2}[u] \preceq_\kappa H^{j_2+2k_2}[u] \preceq_\kappa \cdots. \tag{7}$$

Let j_3 be the least j_3 such that $j_3 = j'_1 + i_3\,k'_1 = j_2 + m_0$ for some i_3 and m_0, and k_3 be the least common multiple of k'_1 and k_2, whence $k_3 = m_1 k'_1 = m_2 k_2$ for some m_1 and m_2. Then since the mapping $s \mapsto \mathcal{T}(G[H^{m_0}[s]])$ is monotonic, from (7) we have:

$$\mathcal{T}(G[H^{j_3}[u]]) \preceq \mathcal{T}(G[H^{j_3+k_2}[u]]) \preceq \mathcal{T}(G[H^{j_3+2k_2}[u]]) \preceq \cdots.$$

Since $j_3 + ik_3 = j_3 + (im_2)k_2$, we have

$$\mathcal{T}(G[H^{j_3}[u]]) \preceq \mathcal{T}(G[H^{j_3+k_3}[u]]) \preceq \mathcal{T}(G[H^{j_3+2k_3}[u]]) \preceq \cdots. \tag{8}$$

Also, since $j_3 + ik_3 = j'_1 + (i_3 + im_1)k'_1$, from (6) we have

$$\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j_3}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j_3+k_3}[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^{j_3+2k_3}[u_{\mathrm{p}}]]) \prec \cdots . \tag{9}$$

Thus, from (4), (8), and (9) we obtain

$$\mathcal{T}(G[H^{j_3}[u]]) \prec \mathcal{T}(G[H^{j_3+k_3}[u]]) \prec \mathcal{T}(G[H^{j_3+2k_3}[u]]) \prec \cdots . \tag{10}$$

By applying **rmdir** to this sequence, and by (1) and Lemma 15, we have

$$\mathcal{T}(C[D^{j_0+j_3k_0}[t]]) \prec \mathcal{T}(C[D^{j_0+(j_3+k_3)k_0}[t]]) \prec \mathcal{T}(C[D^{j_0+(j_3+2k_3)k_0}[t]]) \prec \cdots . \tag{11}$$

We define $j = j_0 + k_0 j_3$ and $k = k_0 k_3$; then we obtain

$$\mathcal{T}(C[D^{j}[t]]) \prec \mathcal{T}(C[D^{j+k}[t]]) \prec \mathcal{T}(C[D^{j+2k}[t]]) \prec \cdots .$$

Finally, we show that $|\mathcal{T}(C[D^{j+ik}[t]])| \leq \mathbf{exp}_n(ci+d)$ for some c and d. Since C and D are single-hole contexts, $|C[D^{j+ik}[t]]| = |C|+(j+ik)|D|+|t|$. Let $c = k|D|$ and $d = |C|+j|D|+|t|$; then $|C[D^{j+ik}[t]]| = ci + d$. It is well-known that, for an order-n $\lambda^{\rightarrow}$-term s, we have $|\mathcal{T}(s)| \leq \mathbf{exp}_n(|s|)$ (see, e.g., [25, Lemma 3]). Thus, we have $|\mathcal{T}(C[D^{j+ik}[t]])| \leq \mathbf{exp}_n(ci+d)$. ☐

The step obtaining (10) (the steps using Lemma 14 and obtaining (11), resp.) indicates why we need to require $\mathcal{T}(C[D^{j+ik}[t]]) \prec \mathcal{T}(C[D^{j+i'k}[t]])$ for any $i < i'$ rather than $|\mathcal{T}(C[D^{j+ik}[t]])| < |\mathcal{T}(C[D^{j+i'k}[t]])|$ ($\mathcal{T}(C[D^{j+ik}[t]]) \neq \mathcal{T}(C[D^{j+i'k}[t]])$, resp.) to make the induction work.

6 Second-order Kruskal's theorem

In this section, we prove Conjecture 7 (hence also Conjecture 8) up to order-2. First, we extend the homeomorphic embedding $\preceq$ on trees to a family of relations $\preceq_\kappa$ by using logical relation: (i) $t_1 \preceq_{\mathsf{o}} t_2$ if $\emptyset \vdash_{\mathsf{ST}} t_1 : \mathsf{o}$, $\emptyset \vdash_{\mathsf{ST}} t_2 : \mathsf{o}$, and $\mathcal{T}(t_1) \preceq \mathcal{T}(t_2)$. (ii) $t_1 \preceq_{\kappa_1\to\kappa_2} t_2$ if $\emptyset \vdash_{\mathsf{ST}} t_1 : \kappa_1 \to \kappa_2$, $\emptyset \vdash_{\mathsf{ST}} t_2 : \kappa_1 \to \kappa_2$, and $t_1 s_1 \preceq_{\kappa_2} t_2 s_2$ holds for every s_1, s_2 such that $s_1 \preceq_{\kappa_1} s_2$. We often omit the subscript κ and just write $\preceq$ for $\preceq_\kappa$. We also write $x_1 : \kappa_1, \ldots, x_k : \kappa_k \models t \preceq_\kappa t'$ if $[s_1/x_1, \ldots, s_k/x_k]t \preceq_\kappa [s'_1/x_1, \ldots, s'_k/x_k]t'$ for every $s_1, \ldots, s_k, s'_1, \ldots, s'_k$ such that $s_i \preceq_{\kappa_i} s'_i$.

The relation $\preceq_\kappa$ is well-defined for $\beta\eta$-equivalence classes, and by the abstraction lemma of logical relation, it turns out that the relation $\preceq_\kappa$ is a pre-order for any κ (see Appendix C for these). Note that the relation is also preserved by applications by the definition of the logical relation. It remains to show that $\preceq_\kappa$ is a well quasi-order for κ of order up to 2.

For ℓ-ary terminal a and $k \geq \ell$, we write $\mathbf{CTerms}_{a,k}$ for the set of terms

$$\{\lambda x_1.\cdots\lambda x_k.a\, x_{i_1}\, \ldots\, x_{i_\ell} \mid i_1 \cdots i_\ell \text{ is a subsequence of } 1\cdots k\}.$$

We define $\mathsf{o}^0 \to \mathsf{o} := \mathsf{o}$ and $\mathsf{o}^{n+1} \to \mathsf{o} := \mathsf{o} \to (\mathsf{o}^n \to \mathsf{o})$.

The following lemma allows us to reduce $t \preceq_\kappa t'$ on any order-2 type κ to (finitely many instances of) that on order-0 type o.

▶ **Lemma 17.** *Let Σ be a ranked alphabet; κ be $(\mathsf{o}^{k_1} \to \mathsf{o}) \to \cdots \to (\mathsf{o}^{k_m} \to \mathsf{o}) \to \mathsf{o}$; a_i^j be a j-ary terminal not in Σ for $1 \leq i \leq m$ and $0 \leq j \leq k_i$; and t, t' be $\lambda^{\rightarrow}$-terms whose type is κ and whose terminals are in Σ. Then $t \preceq_\kappa t'$ if and only if $t\, u_1\, \ldots\, u_m \preceq_{\mathsf{o}} t'\, u_1\, \ldots\, u_m$ for every $u_i \in \cup_{j\leq k_i} \mathbf{CTerms}_{a_i^j,k_i}$.*

Proof. The "only if" direction is trivial by the definition of $\preceq_\kappa$. To show the opposite, assume the latter holds. We need to show that $t\, s_1\, \ldots\, s_m \preceq_{\mathsf{o}} t'\, s_1\, \ldots\, s_m$ holds for every combination of $s_1, \ldots, s_m$ such that $\vdash_{\mathtt{ST}} s_i : \kappa_i$ for each i. Without loss of generality, we can assume that t, t', $s_1, \ldots, s_m$ are $\beta\eta$ long normal forms, and hence that

$$\begin{array}{ll} t = \lambda f_1.\cdots\lambda f_m.t_0 & f_1 : \mathsf{o}^{k_1} \to \mathsf{o}, \ldots, f_m : \mathsf{o}^{k_m} \to \mathsf{o} \vdash_{\mathtt{ST}} t_0 : \mathsf{o} \\ t' = \lambda f_1.\cdots\lambda f_m.t'_0 & f_1 : \mathsf{o}^{k_1} \to \mathsf{o}, \ldots, f_m : \mathsf{o}^{k_m} \to \mathsf{o} \vdash_{\mathtt{ST}} t'_0 : \mathsf{o} \\ s_i = \lambda x_1.\ldots\lambda x_{k_i}.s_{i,0} & x_1 : \mathsf{o}, \ldots, x_{k_i} : \mathsf{o} \vdash_{\mathtt{ST}} s_{i,0} : \mathsf{o} \quad \text{(for each } i\text{)} \end{array}$$

For each $i \le m$, let $\mathbf{FV}(s_{i,0}) = \{x_{q(i,1)}, \ldots, x_{q(i,\ell_i)}\}$, and $u_i \in \mathbf{CTerms}_{a_i^{\ell_i},k_i}$ be the term $\lambda x_1.\cdots\lambda x_{k_i}.a_i^{\ell_i}\, x_{q(i,1)}\, \cdots\, x_{q(i,\ell_i)}$. Let θ and θ' be the substitutions $[u_1/f_1, \ldots, u_m/f_m]$ and $[s_1/f_1, \ldots, s_m/f_m]$ respectively. It suffices to show that $\theta t_0 \preceq_{\mathsf{o}} \theta t'_0$ implies $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_0$, which we prove by induction on $|t'_0|$.

By the condition $f_1 : \mathsf{o}^{k_1} \to \mathsf{o}, \ldots, f_m : \mathsf{o}^{k_m} \to \mathsf{o} \vdash_{\mathtt{ST}} t_0 : \mathsf{o}$, t_0 must be of the form $h\, t_1\, \cdots\, t_\ell$ where h is f_i or a terminal a in Σ, and ℓ may be 0. Then we have

$$\mathcal{T}(\theta t_0) = \begin{cases} a\mathcal{T}(\theta t_1)\cdots\mathcal{T}(\theta t_\ell) & (h = a) \\ a_i^{\ell_i}\mathcal{T}(\theta t_{q(i,1)})\cdots\mathcal{T}(\theta t_{q(i,\ell_i)}) & (h = f_i) \end{cases}$$

Similarly, t'_0 must be of the form $h'\, t'_1\, \cdots\, t'_{\ell'}$ and the corresponding equality on $\mathcal{T}(\theta t'_0)$ holds. By the assumption $\theta t_0 \preceq_{\mathsf{o}} \theta t'_0$, we have $\mathcal{T}(\theta t_0) \preceq \mathcal{T}(\theta t'_0)$. We perform case analysis on the rule used for deriving $\mathcal{T}(\theta t_0) \preceq \mathcal{T}(\theta t'_0)$ (recall Definition 5).

- Case of the first rule: In this case, the roots of $\mathcal{T}(\theta t_0)$ and $\mathcal{T}(\theta t'_0)$ are the same and hence $h = h'$ and $\ell = \ell'$. We further perform case analysis on h.
  - Case $h = a$: For $1 \le j \le \ell$, since $\mathcal{T}(\theta t_j) \preceq \mathcal{T}(\theta t'_j)$, by induction hypothesis, we have $\theta' t_j \preceq_{\mathsf{o}} \theta' t'_j$. Hence $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_0$.
  - Case $h = f_i$: For $1 \le j \le \ell_i$, since $\mathcal{T}(\theta t_{q(i,j)}) \preceq \mathcal{T}(\theta t'_{q(i,j)})$, by induction hypothesis, we have $\theta' t_{q(i,j)} \preceq_{\mathsf{o}} \theta' t'_{q(i,j)}$. Hence, $[\theta' t_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0} \preceq_{\mathsf{o}} [\theta' t'_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$. By the definition of $q(i,j)$, $\theta' t_0 \longrightarrow [\theta' t_j/x_j]_{j \le k_i} s_{i,0} = [\theta' t_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$, and similarly, $\theta' t'_0 \longrightarrow [\theta' t'_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$; hence we have $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_0$.
- Case of the second rule: We further perform case analysis on h'.
  - Case $h' = a$: We have $\mathcal{T}(\theta t_0) \preceq \mathcal{T}(\theta t'_p)$ for some $1 \le p \le \ell'$. Hence by induction hypothesis, we have $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_p$, and then $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_0$.
  - Case $h' = f_i$: We have $\mathcal{T}(\theta t_0) \preceq \mathcal{T}(\theta t'_{q(i,p)})$ for some $1 \le p \le \ell_i$. Hence by induction hypothesis, we have $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_{q(i,p)}$. Also, by the definition of $q(i,p)$, $x_{q(i,p)}$ occurs in $s_{i,0}$. Since $s_{i,0}$ is a $\beta\eta$ long normal form of order-0, the order-0 variable $x_{q(i,p)}$ occurs as a leaf of $s_{i,0}$; hence $\mathcal{T}(\theta' t'_{q(i,p)}) \preceq [\mathcal{T}(\theta' t'_{q(i,j)})/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$. Therefore $\theta' t_0 \preceq_{\mathsf{o}} [\theta' t'_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$. Since $\theta' t'_0 \longrightarrow [\theta' t'_{q(i,j)}/x_{q(i,j)}]_{j \le \ell_i} s_{i,0}$, we have $\theta' t_0 \preceq_{\mathsf{o}} \theta' t'_0$.

□

As a corollary, we obtain a second-order version of Kruskal's tree theorem.

▶ **Theorem 18.** *Let Σ be a ranked alphabet, κ be an at most order-2 type, and $t_0, t_1, t_2, \ldots$ be an infinite sequence of $\lambda^{\to}$-terms whose type is κ and whose terminals are in Σ. Then, there exist $i < j$ such that $t_i \preceq_\kappa t_j$.*

Proof. Since κ is at most order-2, it must be of the form $(\mathsf{o}^{k_1} \to \mathsf{o}) \to \cdots \to (\mathsf{o}^{k_m} \to \mathsf{o}) \to \mathsf{o}$. Let a_i^j be a j-ary terminal not in Σ for $1 \le i \le m$ and $0 \le j \le k_i$; $(\cup_{j \le k_1} \mathbf{CTerms}_{a_1^j,k_1}) \times \cdots \times (\cup_{j \le k_m} \mathbf{CTerms}_{a_m^j,k_m})$ be $\{(u_{1,1}, \ldots, u_{1,m}), \ldots, (u_{p,1}, \ldots, u_{p,m})\}$; b be a p-ary terminal not in $\Sigma \cup \{a_i^j \mid 1 \leq i \leq m,\ 0 \leq j \leq k_i\}$; and s_i be the term $b\,(t_i\,u_{1,1}\,\cdots\,u_{1,m})\,\cdots\,(t_i\,u_{p,1}\,\cdots\,u_{p,m})$ for each $i \in \{0, 1, 2, \ldots\}$. Since the set of terminals in $s_0, s_1, s_2, \ldots$ is finite, by Kruskal's tree theorem, there exist i, j such that $s_i \preceq_{\mathsf{o}} s_j$ and $i < j$. Since b occurs just at the root of s_k for each k, $s_i \preceq_{\mathsf{o}} s_j$ implies $t_i\,u_{k,1}\,\cdots\,u_{k,m} \preceq_{\mathsf{o}} t_j\,u_{k,1}\,\cdots\,u_{k,m}$ for every $k \in \{1, \ldots, p\}$. Thus, by Lemma 17, we have $t_i \preceq_{\kappa} t_j$ as required. ☐

7 Related Work

As mentioned in Section 1, to our knowledge, pumping lemmas for higher-order word languages have been established only up to order-2 [7], whereas we have proved (unconditionally) a pumping lemma for order-2 tree languages and order-3 word languages. Hayashi's pumping lemma for indexed languages (i.e., order-2 word languages) is already quite complex, and it is unclear how to generalize it to arbitrary orders. In contrast, our proof of a pumping lemma works for arbitrary orders, although it relies on the conjecture on higher-order Kruskal's tree theorem. Parys [21] and Kobayashi [12] studied pumping lemmas for collapsible pushdown automata and higher-order recursion schemes respectively. Unfortunately, they are not applicable to word/tree *languages* generated by (non-deterministic) grammars.

As also mentioned in Section 1, the strictness of hierarchy of higher-order word languages has already been shown by using a complexity argument [5, 8]. We can use our pumping lemma (if the conjecture is discharged) to obtain a simple alternative proof of the strictness, using the language $\{a^{\mathbf{exp}_n(k)} \mid k \geq 0\}$ as a witness of the separation between the classes of order-$(n+1)$ word languages and order-n word languages. In fact, the pumping lemma would imply that there is no order-n grammar that generates $\{a^{\mathbf{exp}_n(k)} \mid k \geq 0\}$, whereas an order-$(n+1)$ grammar that generates the same language can be easily constructed.

We are not aware of studies of the higher-order version of Kruskal's tree theorem (Conjecture 7) or the periodicity of tree functions expressed by the simply-typed λ-calculus (Conjecture 8), which seem to be of independent interest. Zaionc [27, 28] characterized the class of (first-order) word/tree functions definable in the simply-typed λ-calculus. To obtain higher-order Kruskal's tree theorem, we may need some characterization of *higher-order* definable tree functions instead.

We have heavily used the results of Parys' work [23] and our own previous work [1], which both use intersection types for studying properties of higher-order languages. Other uses of intersection types in studying higher-order grammars/languages are found in [10, 15, 22, 12, 3, 14, 13].

8 Conclusion

We have proved a pumping lemma for higher-order languages of arbitrary orders, modulo the assumption that a higher-order version of Kruskal's tree theorem holds. We have also proved the assumption indeed holds for the second-order case, yielding a pumping lemma for order-2 tree languages and order-3 word languages. Proving (or disproving) the higher-order Kruskal's tree theorem is left for future work.

Acknowledgments

We would like to thank Pawel Parys for discussions on his type system, and anonymous referees for useful comments. This work was supported by JSPS Kakenhi 15H05706.

References

1 Kazuyuki Asada and Naoki Kobayashi. On Word and Frontier Languages of Unsafe Higher-Order Grammars. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *LIPIcs*, pages 111:1–111:13. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.

2 Yehoshua Bar-Hillel, Micha A. Perles, and Eli Shamir. On formal properties of simple phrase structure grammars. *Z. Phonetik Sprachwiss. und Kommunikat.*, (14):143–172, 1961.

3 Lorenzo Clemente, Pawel Parys, Sylvain Salvati, and Igor Walukiewicz. The diagonal problem for higher-order recusion schemes is decidable. In *Proceedings of LICS 2016*, 2016.

4 Werner Damm. The IO- and OI-hierarchies. *Theor. Comput. Sci.*, 20:95–207, 1982.

5 Joost Engelfriet. Iterated stack automata and complexity classes. *Info. Comput.*, 95(1):21–75, 1991.

6 Joost Engelfriet and Heiko Vogler. High level tree transducers and iterated pushdown tree transducers. *Acta Inf.*, 26(1/2):131–192, 1988.

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

8 Alexander Kartzow. Personal communication, via Pawel Parys, 2013.

9 Teodor Knapik, Damian Niwinski, and Pawel Urzyczyn. Deciding monadic theories of hyperalgebraic trees. In *TLCA 2001*, volume 2044 of *LNCS*, pages 253–267. Springer, 2001.

10 Naoki Kobayashi. Types and higher-order recursion schemes for verification of higher-order programs. In *Proc. of POPL*, pages 416–428. ACM Press, 2009.

11 Naoki Kobayashi. Model checking higher-order programs. *Journal of the ACM*, 60(3), 2013.

12 Naoki Kobayashi. Pumping by typing. In *Proceedings of LICS 2013*, pages 398–407. IEEE Computer Society, 2013.

13 Naoki Kobayashi, Kazuhiro Inaba, and Takeshi Tsukada. Unsafe order-2 tree languages are context-sensitive. In *Proceedings of FoSSaCS 2014*, volume 8412 of *LNCS*, pages 149–163. Springer, 2014.

14 Naoki Kobayashi, Kazutaka Matsuda, Ayumi Shinohara, and Kazuya Yaguchi. Functional programs as compressed data. *Higher-Order and Symbolic Computation*, 2013.

15 Naoki Kobayashi and C.-H. Luke Ong. A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In *Proceedings of LICS 2009*, pages 179–188. IEEE Computer Society Press, 2009.

16 Gregory M. Kobele and Sylvain Salvati. The IO and OI hierarchies revisited. *Inf. Comput.*, 243:205–221, 2015.

17 J. B. Kruskal. Well-quasi-ordering, the tree theorem, and vazsonyi's conjecture. *Transactions of the American Mathematical Society*, 95(2):210–225, 1960.

18 A. N. Maslov. The hierarchy of indexed languages of an arbitrary level. *Soviet Math. Dokl.*, 15:1170–1174, 1974.

19 C. St. J. A. Nash-Williams. On well-quasi-ordering finite trees. *Mathematical Proceedings of the Cambridge Philosophical Society*, 59(4):833–835, 1963.

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

21 Pawel 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.

22 Pawel Parys. How many numbers can a lambda-term contain? In *Proceedings of FLOPS 2014*, volume 8475 of *LNCS*, pages 302–318. Springer, 2014.

23 Pawel Parys. Intersection types and counting. *CoRR*, abs/1701.05303, 2017. A shorter version will appear in Post-proceedings of ITRS 2016.

24 Sylvain Salvati and Igor Walukiewicz. Krivine machines and higher-order schemes. In *Proceedings of ICALP 2011*, volume 6756 of *LNCS*, pages 162–173. Springer, 2011.

25 Kazushige 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. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

26 Mitchell Wand. An algebraic formulation of the Chomsky hierarchy. In *Category Theory Applied to Computation and Control*, volume 25 of *LNCS*, pages 209–213. Springer, 1974.

27 Marek Zaionc. Word operation definable in the typed lambda-calculus. *Theor. Comput. Sci.*, 52:1–14, 1987.

28 Marek Zaionc. On the "lambda"-definable tree operations. In *Algebraic Logic and Universal Algebra in Computer Science, Conference, Ames, Iowa, USA, June 1-4, 1988, Proceedings*, volume 425 of *Lecture Notes in Computer Science*, pages 279–292, 1990.

A More Details on Section 3

We provide more details on how Parys' type system [23] can be modified to obtain Lemmas 11 and 12. Some familiarity (especially, intuitions on flags and markers) with Parys' type system is required to understand this section. In Section A.1, we first present a variant of Parys' type system and state key lemmas. We then prove Lemma 11 using the lemmas. After preparing some basic lemmas about the type system A.2 we prove the key lemmas in Sections A.3 and A.4. In Section A.5, we modify the type system to show how to extract a triple (G, H, u) that satisfies the requirement of Lemma 12.

A.1 A Variant of Parys' Type System and its Key Properties

Below we fix a grammar $\mathcal{G}$.

The set of types, ranged over by γ, is defined by:

$$\gamma\ \text{(types)}\ ::= (F, M, \rho) \qquad \rho\ \text{(raw types)}\ ::= \mathsf{o} \mid \xi \to \rho$$
$$\xi\ \text{(intersection types)}\ ::= \{\gamma_1, \dots, \gamma_k\}$$

Here, F and M range over the finite powerset of the set of natural numbers, and F and M must be disjoint. Intuitively, (F, M, o) is the type of trees which contain flags of orders in F and markers of orders in M. The type $(F, M, \xi \to \rho)$ describes a function term which, when viewed as a function, takes an argument that has all the types in ξ and returns a value of type ρ, and when viewed as a term, contains flags of orders in F and markers of orders in M. Thus, each type expresses the "dual" views of a term, both as a function and a term. In order for $\xi \to \rho$ to be well-formed, for each $(F, M, \rho') \in \xi$, it must be the case that $F, M \subseteq \{i \mid 0 \le i < \mathtt{order}(\xi \to \rho)\}$. For a set of natural numbers S and a natural number n, we define $S{\restriction_{<n}} := \{m \in S \mid m < n\}$ and $S{\restriction_{\ge n}} := \{m \in S \mid m \ge n\}$. We assume some total order $<$ on types.

A type environment is a set of bindings of the form $x : \gamma$, which may contain more than one binding on the same variable. We use the meta-variable Θ for a type environment. We require that all the marker sets occurring in Θ are mutually disjoint, i.e., if $x : (F, M, \rho), x' : (F', M', \rho') \in \Theta$, then either $M \cap M' = \emptyset$ or $x : (F, M, \rho) = x' : (F', M', \rho')$. We write $\mathit{Markers}(\Theta)$ for the set of markers occurring in Θ, i.e., $\uplus\{M \mid x : (F, M, \rho) \in \Theta\}$. Also, we write $\mathit{Markers}(\xi)$ for $\mathit{Markers}(x : \xi)$. We write $\Theta \le \Theta'$ if $\Theta' \subseteq \Theta$ and $M = \emptyset$ for each $(x : (F, M, \rho)) \in \Theta \setminus \Theta'$. We define $\Theta_1 + \Theta_2$ as $\Theta_1 \cup \Theta_2$ only if $M = \emptyset$ for any $(x : (F, M, \rho)) \in \Theta_1 \cap \Theta_2$.

The operation $Comp_n(\{(F_1, c_1), \ldots, (F_k, c_k)\}, M) = (F, c)$ on flags, markers and counters is defined by:

$$
\begin{array}{l}
f'_0 = 0 \qquad f'_\ell = \begin{cases} f_{\ell-1} & \text{if } \ell - 1 \in M \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } \ell \in \{1, \ldots, n\} \\
f_\ell = f'_\ell + |\{i \mid \ell \in F_i\}| \quad \text{for each } \ell \in \{0, \ldots, n-1\} \\
F = \{\ell \in \{0, \ldots, n-1\} \mid f_\ell > 0\} \setminus M \qquad c = f'_n + c_1 + \cdots + c_k
\end{array}
$$

Here, (although we use a set notation) please note that the first argument of $Comp_n$ is a *multiset*. In fact, $Comp_1(\{(\{0\}, 0), (\{0\}, 0)\}, \{0\}) = 2$, but $Comp_1(\{(\{0\}, 0)\}, \{0\}) = 1$. We write $\cup_{\mathtt{mul}}$ for the union of multisets: $(X \cup_{\mathtt{mul}} Y)(x) := X(x) + Y(x)$. Note that, by unfolding the definition we have:

$$
\begin{array}{l}
f_\ell > 0 \quad \Leftrightarrow \quad \exists j \in \{0, \ldots, \ell\}.\ j \in \cup_{i \le k} F_i \ \wedge\ \{j, \ldots, \ell - 1\} \subseteq M \\
f'_n = \big|\{i \mid n-1 \in F_i\}\big| + \cdots + \big|\{i \mid n-j \in F_i\}\big| = \sum_{i \le k} \big|\{n-1, \ldots, n-j\} \cap F_i\big| \\
\qquad \text{where } j \text{ is such that: } 0 \le j \le n;\ n-1, \ldots, n-j \in M;\ n-(j+1) \notin M.
\end{array}
$$

A type judgment (or more precisely, a type-based transformation judgment) is of the form $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ where $n > 0$, Θ is a type environment that contains types of orders at most $n-1$, $\mathtt{eorder}(t) \le n$, and s is a $\lambda^{\to}$-term. The flags and markers in the judgment must be at most $n-1$. We present typing rules below. For some technical convenience, we made some changes to Parys' original type system [23], which are summarized below.

- We have added the output s of the transformation. Intuitively, it simulates the behavior of t.
- We allow markers to be placed at any node of a derivation, not just at leaves (see the rule (PTr-Mark) below).
- We added a rule ((PTr-NT) below) for explicitly unfolding non-terminals. Here we assume that a grammar is expressed as a set of equations of the form $\{A_1 = t_1, \ldots, A_\ell = t_\ell\}$ where $A_1, \ldots, A_\ell$ are distinct from each other. A set of rewriting rules $\{A\, x_1\, \ldots\, x_k \to t_i \mid i \in \{1, \ldots, p\}\}$ is expressed as the equation $A = \lambda x_1. \cdots \lambda x_k.(t_1 + \cdots + t_p)$. In the original formulation of Parys, infinite λ-terms (represented as regular trees) were considered instead.
- We consider judgments $\Theta \vdash_n t : \gamma \triangleright c$ only when $n > 0$. (In the original type system of Parys [23], n may be 0.)
- In rule (PTr-App), we allow only a single derivation for each argument type.

$$
\frac{\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s \qquad M = \emptyset}{\Theta, x : (F, M, \rho) \vdash_n t : \gamma \triangleright c \Rightarrow s} \quad \text{(PTr-Weak)}
$$

$$
\frac{\begin{array}{c}\Theta \vdash_n t : (F', M', \rho) \triangleright c' \Rightarrow s \\ M \subseteq \{j \mid \mathtt{eorder}(t) \le j < n\} \qquad Comp_n(\{(F', c')\}, M \uplus M') = (F, c)\end{array}}{\Theta \vdash_n t : (F, M \uplus M', \rho) \triangleright c \Rightarrow s}
$$

(PTr-Mark)

$$
\frac{}{x : \gamma \vdash_n x : \gamma \triangleright 0 \Rightarrow x_\gamma} \quad \text{(PTr-Var)}
$$

$$
\frac{\Theta \vdash_n t_i : \gamma \triangleright c \Rightarrow s_i \qquad i = 1 \vee i = 2}{\Theta \vdash_n t_1 + t_2 : \gamma \triangleright c \Rightarrow s_i} \quad \text{(PTr-Choice)}
$$

$$\frac{\Theta, x : \xi \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s}{\Theta \vdash_n \lambda x.t : (F, M \setminus \mathit{Markers}(\xi), \xi \to \rho) \triangleright c \Rightarrow \lambda \widetilde{x}_\xi.s} \quad \text{(PTr-Abs)}$$

Here, $\lambda \widetilde{x}_\xi.s$ stands for $\lambda x_{\gamma_1}.\cdots\lambda x_{\gamma_k}.s$ when $\xi = \{\gamma_1, \dots, \gamma_k\}$ with $\gamma_1 < \cdots < \gamma_k$.

$$\frac{\begin{array}{c}\texttt{eorder}(t_0) = \ell \\ \Theta_0 \vdash_n t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c_0 \Rightarrow s_0 \\ \Theta_i \vdash_n t_1 : (F'_i, M'_i, \rho_i) \triangleright c_i \Rightarrow s_i \qquad F'_i\restriction_{<\ell} = F_i \qquad M'_i\restriction_{<\ell} = M_i \text{ for each } i \in \{1, \dots, k\} \\ M = M_0 \uplus (\biguplus_{i \in \{1,\dots,k\}} M'_i) \\ \mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\texttt{mul}} \{(F'_i\restriction_{\geq \ell}, c_i) \mid i \in \{1, \dots, k\}\}, M) = (F, c) \\ (F_1, M_1, \rho_1) < \cdots < (F_k, M_k, \rho_k)\end{array}}{\Theta_0 + (\sum_{i \in \{1,\dots,k\}} \Theta_i) \vdash_n t_0 t_1 : (F, M, \rho) \triangleright c \Rightarrow s_0\, s_1 \, \cdots \, s_k} \quad \text{(PTr-App)}$$

$$\frac{\begin{array}{c}\texttt{ar}(a) = k \\ \Theta_i \vdash_n t_i : (F_i, M_i, \mathsf{o}) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \dots, k\} \\ M = M_1 \uplus \cdots \uplus M_k \\ \mathit{Comp}_n(\{(\{0\}, 0), (F_1, c_1), \dots, (F_k, c_k)\}, M) = (F, c)\end{array}}{\Theta_1 + \cdots + \Theta_k \vdash_n a\, t_1 \, \cdots \, t_k : (F, M, \mathsf{o}) \triangleright c \Rightarrow a\, s_1 \, \cdots \, s_k} \quad \text{(PTr-Const)}$$

$$\frac{\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s \qquad A = t \in \mathcal{G}}{\Theta \vdash_n A : \gamma \triangleright c \Rightarrow s} \quad \text{(PTr-NT)}$$

We state two key properties of the type system below. The following lemma states that if $\mathcal{L}(\mathcal{G})$ is infinite, there exists a pumpable derivation tree in which a part of the derivation can be repeated arbitrarily many times.

▶ **Lemma 19** (existence of a pumpable derivation). *Let $\mathcal{G}$ be an order-n tree grammar and S be its start symbol. If $\mathcal{L}(\mathcal{G})$ is infinite, then there exists a derivation for $\emptyset \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c_1 + c_2 + c_3 \Rightarrow C[D[s]]$ with $c_1, c_2 > 0$, in which for some non-terminal A, $\Theta \vdash_n A : \gamma \triangleright c_1 + c_2 \Rightarrow D[s]$ is derived from $\Theta \vdash_n A : \gamma \triangleright c_1 \Rightarrow s$. Furthermore, the contexts C and D are linear.*

The following lemma states that any simply-typed λ-term s obtained by the transformation generates a member of $\mathcal{L}(\mathcal{G})$, and its size is bounded below by c.

▶ **Lemma 20** (soundness of transformation). *Let $\mathcal{G}$ be an order-n tree grammar and S be its start symbol. If $\emptyset \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$, then s is a $\lambda^{\to}$-term of order at most n, and $\mathcal{T}(s) \in \mathcal{L}(\mathcal{G})$, with $c \leq |\mathcal{T}(s)|$.*

We will prove Lemmas 19 and 20 above in Sections A.3 and A.4 respectively. Using the lemmas above, we can prove Lemma 11.

Proof of Lemma 11. Suppose that $\mathcal{L}(\mathcal{G})$ is infinite. By Lemma 19, we have a pumpable derivation for $\emptyset \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c_1 + c_2 + c_3 \Rightarrow C[D[s]]$ with $c_1, c_2 > 0$, in which $\Theta \vdash_n A : \gamma \triangleright c_1 + c_2 \Rightarrow D[s]$ is derived from $\Theta \vdash_n A : \gamma \triangleright c_1 \Rightarrow s$, and contexts C, D are linear. The orders of s, C, and D are at most n; by inserting a dummy subterm, we can assume that the orders of them are n. By repeating the subderivation from $\Theta \vdash_n A : \gamma \triangleright c_1 \Rightarrow s$ to $\Theta \vdash_n A : \gamma \triangleright c_1 + c_2 \Rightarrow D[s]$, we obtain a derivation for

$$\Theta \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c_1 + kc_2 + c_3 \Rightarrow C[D^k[s]]$$

for any $k \geq 0$. By Lemma 20, we have $\{\mathcal{T}(C[D^k[t]]) \mid k \geq 1\} \subseteq \mathcal{L}(\mathcal{G})$. Let $(\ell_k)_k$ be a strictly increasing sequence. Then, the set $\{c_1 + \ell_k c_2 + c_3 \mid k \geq 1\}$ is infinite. Thus, by the condition $|\mathcal{T}(C[D^{\ell_k}[t]])| \geq c_1 + \ell_k c_2 + c_3$, $\{|\mathcal{T}(C[D^{\ell_k}[t]])| \mid k \geq 1\}$ must be infinite, which also implies that $\{\mathcal{T}(C[D^{\ell_k}[t]]) \mid k \geq 1\}$ is infinite. ☐

A.2 Basic Definitions and Lemmas

Here we prepare some definitions and lemmas that are commonly used in Sections A.3 and A.4.

We first define a refined notion of reductions. For proving the key lemmas (Lemmas 19 and 20), we consider a specific reduction sequence in which redexes of higher-order are reduced first. Thus we consider a restricted version $\longrightarrow_n$ of the reduction relation, where only redexes of order-n can be reduced (in addition to unfolding of non-terminals).

We define *order-n reduction*, written by $\longrightarrow_n$, by the following rules.

$$C[(\lambda x.t)s] \longrightarrow_n C[[s/x]t] \quad \text{if} \quad (\mathtt{eorder}(\lambda x.t) = n)$$
$$C[A] \longrightarrow_n C[t] \quad \text{if} \quad ((A = t) \in \mathcal{G}).$$

Also we define a reduction on the nondeterministic choice, written $\longrightarrow_{\mathrm{c}}$, as the following:

$$C[t_1 + t_2] \longrightarrow_{\mathrm{c}} C[t_i] \quad (i = 1, 2).$$

The following lemma states that any reduction sequence for generating a tree can be normalized, so that reductions are applied in a decreasing order.

▶ **Lemma 21.** *Let $\mathcal{G}$ be an order-n grammar and S be its start symbol. If $\pi \in \mathcal{L}(\mathcal{G})$, then*

$$S \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0 \longrightarrow_{\mathrm{c}}^* \pi.$$

Proof. Given a reduction sequence $S \longrightarrow^* \pi$, we can move any unfolding of a non-terminal to the left, and any reduction of choice to the right, and obtain a reduction sequence

$$S \longrightarrow_n^* t_n \longrightarrow^* t_0 \longrightarrow_{\mathrm{c}}^* \pi$$

in which only β-reductions are applied in $t_n \longrightarrow^* t_0$ (recall that we use only the nondeterministic choice of the ground type, by the assumption for grammars in Section A.1). Note that if the largest order of redexes in t is k, reducing the rightmost, innermost order-k redex neither introduces any new redex of order higher than k, nor copies any redex of order k. Thus, we can obtain a normalizing[7] sequence $t_n \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0'$, where t_0' does not contain any β-redex. By Church-Rosser theorem, $t_0 = t_0'$. Thus, we have

$$S \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0 \longrightarrow_{\mathrm{c}}^* \pi$$

as required. ☐

▶ **Lemma 22.** *If $x_1{:}\rho_1, \ldots, x_k{:}\rho_k \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$, then $\mathbf{FV}(s) \subseteq \{(x_1)_{\rho_1}, \ldots, (x_k)_{\rho_k}\}$.*

Proof. By straightforward induction on a derivation tree of $x_1{:}\rho_1, \ldots, x_k{:}\rho_k \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$ and by case analysis on the last rule of the derivation. ☐

[7] Here, we consider non-terminals and the choice operator as constants.

We say $(F, M, \xi_1 \to \cdots \to \xi_k \to \mathsf{o})$ is *n-clear* if $n \notin M \cup (\bigcup_{i \leq k} \mathit{Markers}(\xi_i))$.

▶ **Lemma 23.** *For $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ with $n > 0$, if γ is $(n-1)$-clear, then $c = 0$.*

Proof. The proof proceeds by induction on the derivation $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ and its case analysis. The cases of (PTr-Var), (PTr-Choice), (PTr-Abs), and (PTr-NT) are clear. The cases of (PTr-Mark) and (PTr-Const) are similar to (and easier than) the case of (PTr-App).

In the case of (PTr-App), we have

$$\frac{\begin{array}{c}\mathtt{eorder}(t_0) = \ell \\ \Theta_0 \vdash_n t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c_0 \Rightarrow s_0 \\ \Theta_i \vdash_n t_1 : (F'_i, M'_i, \rho_i) \triangleright c_i \Rightarrow s_i \qquad F'_i{\restriction_{<\ell}} = F_i \qquad M'_i{\restriction_{<\ell}} = M_i \text{ for each } i \in \{1, \ldots, k\} \\ M = M_0 \uplus (\biguplus_{i \in \{1,\ldots,k\}} M'_i) \\ \mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\mathtt{mul}} \{(F'_i{\restriction_{\geq \ell}}, c_i) \mid i \in \{1, \ldots, k\}\}, M) = (F, c) \\ (F_1, M_1, \rho_1) < \cdots < (F_k, M_k, \rho_k)\end{array}}{\Theta_0 + (\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n t_0 t_1 : (F, M, \rho) \triangleright c \Rightarrow s_0\, s_1 \, \cdots \, s_k}$$

It is clear from the assumption that $(F_0, M_0, \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\} \to \rho)$ is $(n-1)$-clear. Also, each (F'_i, M'_i, ρ_i) is $(n-1)$-clear since $\mathtt{eorder}(t_1) \leq n-1$ and due to the well-formedness of types. Hence, by induction hypothesis we have $c_0 = 0$ and $c_i = 0$ for any i. It is clear that, in general, $\mathit{Comp}_n(\{(F_1, 0), \ldots, (F_k, 0)\}, M) = (F, 0)$ if $n - 1 \notin M$. Thus $c = 0$, as required. ☐

▶ **Lemma 24.** *For $\Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$ with $\mathtt{eorder}(t) \leq n-1$, if $c > 0$, then $n - 1 \in M$.*

Proof. This follows from Lemma 23 (and the well-formedness of types). ☐

The following lemma corresponds to [23, Lemma 24]

▶ **Lemma 25.** *If $\Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$ then $\mathit{Markers}(\Theta) \subseteq M$.*

Proof. By straightforward induction on the derivation of $\Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$ and case analysis of the last rule of the derivation. ☐

▶ **Lemma 26** ([23, Lemma 26])**.** *For $F_0, M' \subseteq \{0, \ldots, n-1\}$, if*

$$\mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\mathtt{mul}} \{(\emptyset, c_i) \mid i \in \{1, \ldots, k\}\}, M') = (F', c')$$

and $F_0 \cap M' = \emptyset$ then

$$F' = F_0 \qquad c' = c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i.$$

A.3 Proof of Lemma 19

To prove Lemma 19, we first introduce a linear type system for typing the output of the transformation, in Section A.3.1. The linear type system is required to guarantee that the contexts C and D are linear. We will then prove, in Section A.3.2, a certain completeness property of the transformation relation, that if $\pi \in \mathcal{G}$, then there exists a derivation $\emptyset \vdash_n S : (\emptyset, \{0, \ldots, n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$ such that $\mathbf{exp}_n(c) \geq |\pi|$, and s is well-typed in the linear type system. We will then prove Lemma 19 in Section A.3.3

A.3.1 Linear Type System and Translation

The syntax of linear types is given by:

$$\begin{aligned}
&\delta ::= \beta^m \\
&\beta ::= \mathsf{o} \mid \delta_1 \to \delta_2 \\
&m ::= 1 \mid \omega
\end{aligned}$$

When $\delta = \beta^m$, we write $\mathbf{mul}(\delta)$ for m and call it the *multiplicity* of δ. Intuitively, β^1 (β^ω, resp.) represents the type of values that can be used once (arbitrarily many times, resp.). We require the well-formedness condition that in every function type $(\delta_1 \to \delta_2)^m$, if m or $\mathbf{mul}(\delta_1)$ is 1, then so is $\mathbf{mul}(\delta_2)$, and exclude out types containing ill-formed types below.

We define partial operations on multiplicities, types, and type environments by:

$$\begin{aligned}
&\omega + \omega = \omega \\
&1 \cdot m = m \qquad \omega \cdot \omega = \omega \\
&\beta^{m_1} + \beta^{m_2} = \beta^{m_1 + m_2} \\
&m\beta^{m'} = \beta^{m \cdot m'} \\
&(\Gamma_0 + \Gamma_1)(x) = \begin{cases} \Gamma_0(x) + \Gamma_1(x) & \text{if } x \in \mathit{dom}(\Gamma_0) \cap \mathit{dom}(\Gamma_1) \\ \Gamma_i(x) & \text{if } x \in \mathit{dom}(\Gamma_i) \setminus \mathit{dom}(\Gamma_{1-i}) \end{cases} \\
&(m\Gamma)(x) = m(\Gamma(x)) \quad \text{if } x \in \mathit{dom}(\Gamma)
\end{aligned}$$

The linear type judgment relation $\Gamma \vdash_{\mathtt{lin}} s : \delta$ is defined by the following typing rules; note that these rules are applicable only when the above partial operations on multiplicities, types, and environments are well-defined.

$$\frac{\Gamma \vdash_{\mathtt{lin}} s : \delta}{\Gamma, x : \beta^\omega \vdash_{\mathtt{lin}} s : \delta} \quad \text{(LT-Weak)}$$

$$\frac{}{x : \delta \vdash_{\mathtt{lin}} x : \delta} \quad \text{(LT-Var)}$$

$$\frac{\mathtt{ar}(a) = k \qquad \Gamma_i \vdash_{\mathtt{lin}} s_i : \mathsf{o}^m \quad \text{for each } i \in \{1, \dots, k\}}{\Gamma_1 + \cdots + \Gamma_k \vdash_{\mathtt{lin}} a\, s_1 \, \cdots \, s_k : \mathsf{o}^m} \quad \text{(LT-Const)}$$

$$\frac{\Gamma, x : \delta_1 \vdash_{\mathtt{lin}} s : \delta_2}{m\Gamma \vdash_{\mathtt{lin}} \lambda x.s : (\delta_1 \to \delta_2)^m} \quad \text{(LT-Abs)}$$

$$\frac{\Gamma_0 \vdash_{\mathtt{lin}} s_0 : (\delta_1 \to \delta)^m \qquad \Gamma_1 \vdash_{\mathtt{lin}} s_1 : \delta_1}{\Gamma_0 + \Gamma_1 \vdash_{\mathtt{lin}} s_0 s_1 : \delta} \quad \text{(LT-App)}$$

$$\frac{\Gamma \vdash_{\mathtt{lin}} s : \beta^\omega}{\Gamma \vdash_{\mathtt{lin}} s : \beta^1} \quad \text{(LT-Dereliction)}$$

▶ **Lemma 27.** *If $\Gamma \vdash_{\mathtt{lin}} s : \delta$ and x occurs in s, then $x : \delta \in \Gamma$ for some x.*

Proof. Straightforward induction on the derivation of $\Gamma \vdash_{\mathtt{lin}} s : \delta$. ◻

▶ **Lemma 28.** *If $\Gamma, x : \beta^1 \vdash_{\mathtt{lin}} s : \delta$, then t contains exactly one occurrence of x.*

Proof. This follows by straightforward induction on the derivation of $\Gamma, x:\beta^1 \vdash_{\texttt{lin}} s:\delta$. We discuss only the case where the last rule is (LT-ABS). In that case, $s = \lambda y.s'$ and $\delta = (\delta_1 \to \delta_2)^m$, where m must be 1 (because $\omega \cdot 1$ is undefined). Thus, we have $\Gamma, x:\beta^1, y:\delta_1 \vdash_{\texttt{lin}} s':\delta_2$. By the induction hypothesis, x occurs exactly once in s', hence also in s. ☐

▶ **Lemma 29.** *If $\Gamma \vdash_{\texttt{lin}} s:\beta^\omega$, then $\mathbf{mul}(\delta) = \omega$ for every $x:\delta \in \Gamma$.*

Proof. This follows by induction on the derivation of $\Gamma \vdash_{\texttt{lin}} s:\beta^\omega$, with case analysis on the last rule used. Since the other cases are trivial, we discuss only the case where the last rule is (LT-APP). In that case, we have $s = s_0 s_1$ and:

$$\Gamma_0 \vdash_{\texttt{lin}} s_0 : (\delta_1 \to \beta^\omega)^m$$
$$\Gamma_1 \vdash_{\texttt{lin}} s_1 : \delta_1$$

Since the well-formedness condition on linear types, both m and $\mathbf{mul}(\delta_1)$ must be ω. Thus, the results follows immediately from the induction hypothesis. ☐

▶ **Lemma 30** (substitution). *Suppose $\Gamma_0, x:\delta' \vdash_{\texttt{lin}} s_0:\delta$ and $\Gamma_1 \vdash_{\texttt{lin}} s_1:\delta'$. If $\Gamma_0+\Gamma_1$ is well-defined, then $\Gamma_0+\Gamma_1 \vdash_{\texttt{lin}} [s_1/x]s_0:\delta$.*

Proof. This follows by induction on the derivation of $\Gamma_0, x:\delta' \vdash_{\texttt{lin}} s_0:\delta$, with case analysis on the last rule used. We discuss only the main cases; the other cases are trivial or similar.

- Case (LT-VAR): In this case, $s_0 = x$ and $\Gamma_0 = \emptyset$. Thus, the result follows immeidately.
- Case (LT-APP): In this case, we have $s_0 = s_{0,0}s_{0,1}$ and:

  $$\Gamma_{0,0} \vdash_{\texttt{lin}} s_{0,0} : (\delta_1 \to \delta)^m$$
  $$\Gamma_{0,1} \vdash_{\texttt{lin}} s_{0,1} : \delta_1$$
  $$\Gamma_{0,0} + \Gamma_{0,1} = \Gamma_0, x:\delta'$$

  Let $\Gamma'_{0,i}$ be the environment obtained by removing $x:\delta'$ from $\Gamma_{0,i}$. We perform case analysis on $\mathbf{mul}(\delta')$.
  - If $\mathbf{mul}(\delta') = \omega$, then we have $\Gamma'_{0,0}+\Gamma_1 \vdash_{\texttt{lin}} [s_1/x]s_{0,0} : (\delta_1 \to \delta)^m$, because:
    1. If $\Gamma_{0,0} = \Gamma'_{0,0}, x:\delta'$, then the result follows from the induction hypothesis.
    2. If $\Gamma_{0,0} = \Gamma'_{0,0}$, then by Lemma 27, x does not occur in s_0. Thus, we have $[s_1/x]s_{0,0} = s_{0,0}$, hence also $\Gamma'_{0,0} \vdash_{\texttt{lin}} [s_1/x]s_0 :: (\delta_1 \to \delta)^m$. By Lemma 29, Γ_1 contains only non-linear types. Therefore we obtaned the required result by using (LT-WEAK).

    Similarly, we also have $\Gamma'_{0,1}+\Gamma_1 \vdash_{\texttt{lin}} [s_1/x]s_{0,1} : \delta_1$. Since Γ_1 contains only non-linear types, we have:

    $$(\Gamma'_{0,0}+\Gamma_1) + (\Gamma'_{0,1}+\Gamma_1) = (\Gamma'_{0,0}+\Gamma'_{0,1}) + \Gamma_1 = \Gamma_0+\Gamma_1.$$

    Thus, by using (LT-APP), we have the required result.
  - If $\mathbf{mul}(\delta') = 1$, then by Lemma 28, x occurs exactly once in either $s_{0,0}$ or $s_{0,1}$. Since the other case is similar, let us consider only the case where x occurs in $s_{0,0}$. Then by Lemma 27, we have $\Gamma'_{0,0}, x:\delta' \vdash_{\texttt{lin}} s_{0,0} : (\delta_1 \to \delta)^m$ and $\Gamma'_{0,1} \vdash_{\texttt{lin}} s_{0,1} : \delta_1$. By the induction hypothesis, we have $\Gamma'_{0,0}+\Gamma_1 \vdash_{\texttt{lin}} [s_1/x]s_{0,0} : (\delta_1 \to \delta)^m$. By applying (LT-APP), we obtain $(\Gamma'_{0,0}+\Gamma_1)+\Gamma'_{0,1} \vdash_{\texttt{lin}} ([s_1/x]s_{0,0})s_{0,1} : \delta$. The result follows, since $(\Gamma'_{0,0}+\Gamma_1)+\Gamma'_{0,1} = (\Gamma'_{0,0}+\Gamma'_{0,1})+\Gamma_1 = \Gamma_0+\Gamma_1$, and $([s_1/x]s_{0,0})s_{0,1} = [s_1/x](s_{0,0}s_{0,1})$.

☐

▶ **Lemma 31** (subject reduction). *If $\Gamma \vdash_{\texttt{lin}} s:\delta$ and $s \longrightarrow s'$, then $\Gamma \vdash_{\texttt{lin}} s':\delta$.*

Proof. This follows by induction on the derivation of $\Gamma \vdash_{\texttt{lin}} s : \delta$, with case analysis on the last rule used. Since the other cases are trivial, we discuss only the case where the last rule is (LT-App), in which case $s = s_0 s_1$. If the reduction $s \longrightarrow s'$ comes from that of s_0 or s_1, the result follows immediately. Thus, we can focus on the case where $s_0 = \lambda x.s'_0$ and $s' = [s_1/x]s'_0$. By (LT-App), we have:

$$\Gamma_0 \vdash_{\texttt{lin}} \lambda x.s'_0 : (\delta' \to \delta)^m \qquad \Gamma_1 \vdash_{\texttt{lin}} s_1 : \delta' \qquad \Gamma = \Gamma_0 + \Gamma_1.$$

By the first condition, we also have $\Gamma_0, x : \delta' \vdash_{\texttt{lin}} s_0 : \delta$. By Lemma 30, we have $\Gamma_0 + \Gamma_1 \vdash_{\texttt{lin}} [s_1/x]s_0$ as required. ☐

▶ **Lemma 32.** *If there exists a derivation sequence for $\emptyset \vdash_{\texttt{lin}} C[s] : \delta$ in which a linear type is assigned to every subterm containing s, then C is a linear context.*

Proof. By the assumption that a linear type is assigned to every subterm containing s, we have $x : \beta^1 \vdash_{\texttt{lin}} C[x] : \delta$, where x is a fresh variable, and β^1 is the type assigned to s in the derivation of $\emptyset \vdash_{\texttt{lin}} C[s] : \delta$. (Note that m must be 1 whenever (LT-Abs) is applied to a term containing s.) Let t be the call-by-name normal formm of $C[x]$. By Lemma 31, we have $x : \beta^1 \vdash_{\texttt{lin}} t : \delta$. By Lemma 28, t contains exactly one occurrence of x. ☐

We now give a translation $(\cdot)^!$ from intersection types to linear types. Our intention is that if $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$, $\delta^!$ represents the type of s (although it does not always hold, actually). We regard a type with a non-empty marker set as a linear type. In the case of a function type $(F, M, \xi \to \gamma)$, markers in the argument type ξ are passed to a return value; thus we take them into account to determine the linearity of γ.

$$(F, M, \mathsf{o})^! = \begin{cases} \mathsf{o}^1 & \text{if } M \neq \emptyset \\ \mathsf{o}^\omega & \text{otherwise} \end{cases}$$

$$(F, M, \xi \to \rho)^! =$$

$$\begin{cases} (F, M, \rho)^! & \text{if } \xi = \emptyset \\ \left(\gamma^! \to (F, M \uplus \mathit{Markers}(\gamma), \xi' \to \rho)^!\right)^1 & \text{if } M \neq \emptyset,\ \xi = \{\gamma\} \cup \xi', \text{ and } \gamma < \xi' \\ \left(\gamma^! \to (F, M \uplus \mathit{Markers}(\gamma), \xi' \to \rho)^!\right)^\omega & \text{if } M = \emptyset,\ \xi = \{\gamma\} \cup \xi', \text{ and } \gamma < \xi' \end{cases}$$

Here, $\gamma < \xi$ means $\gamma < \gamma'$ holds for every $\gamma' \in \xi$. Note that for every γ, $\gamma^!$ is a well-formed linear type.

The translation is extended for type envrionments by:

$$\Gamma^! = \{x_\gamma : \gamma^! \mid x : \gamma \in \Gamma\}.$$

The translation judgment $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ is transformed to a linear type judgment by:

$$(\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s)^! = \begin{cases} \Theta^! \vdash_{\texttt{lin}} s : \beta^1 & \text{if } c > 0 \text{ and } \gamma^! = \beta^m \\ \Theta^! \vdash_{\texttt{lin}} s : \gamma^! & \text{otherwise} \end{cases}$$

Given a derivation tree $\mathcal{D}$ for $\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$, we write $\mathcal{D}^!$ for the derivation tree obtained by each judgment $\Theta' \vdash_n t' : \gamma' \triangleright c' \Rightarrow s'$ with $(\Theta' \vdash_n t' : \gamma' \triangleright c' \Rightarrow s')^!$. Note that $\mathcal{D}^!$ may not be a valid derivation tree in the linear type system. We say that $\mathcal{D}^!$ is *admissible* if, for each derivation step $\dfrac{J_1 \quad \cdots \quad J_k}{J}$ in $\mathcal{D}^!$, J can be obtained from $J_1, \ldots, J_k$ in the linear type system.

▶ **Example 33.** Let $\mathcal{D}_0$ be:

$$\dfrac{\dfrac{x:(\emptyset,\{0\},\mathtt{o})\vdash_1 x:(\emptyset,\{0\},\mathtt{o})\triangleright 0\Rightarrow x_{(\emptyset,\{0\},\mathtt{o})}}{x:(\emptyset,\{0\},\mathtt{o})\vdash_1 \mathtt{a}\,x:(\emptyset,\{0\},\mathtt{o})\triangleright 1\Rightarrow \mathtt{a}\,x_{(\emptyset,\{0\},\mathtt{o})}}}{\emptyset\vdash_1 \lambda x.\mathtt{a}\,x:(\emptyset,\emptyset,(\emptyset,\{0\},\mathtt{o})\to\mathtt{o})\triangleright 1\Rightarrow \lambda x_{(\emptyset,\{0\},\mathtt{o})}.\mathtt{a}\,x_{(\emptyset,\{0\},\mathtt{o})}}$$

$\mathcal{D}_0{}^!$ is:

$$\dfrac{\dfrac{x_{(\emptyset,\{0\},\mathtt{o})}:\mathtt{o}^1\vdash_{\mathtt{lin}} x_{(\emptyset,\{0\},\mathtt{o})}:\mathtt{o}^1}{x_{(\emptyset,\{0\},\mathtt{o})}:\mathtt{o}^1\vdash_{\mathtt{lin}} \mathtt{a}\,x_{(\emptyset,\{0\},\mathtt{o})}:\mathtt{o}^1}}{\emptyset\vdash_{\mathtt{lin}} \lambda x_{(\emptyset,\{0\},\mathtt{o})}.\mathtt{a}\,x_{(\emptyset,\{0\},\mathtt{o})}:(\mathtt{o}^1\to\mathtt{o}^1)^1},$$

which is admissible.

▶ **Example 34.** Not every valid derivation in the intersection type system is mapped to an admissible derivation in the linear type system. Let $\mathcal{D}_1$ be:

$$\dfrac{\dfrac{\dfrac{\cdots}{x:(\{0\},\emptyset,\mathtt{o})\vdash_2 \mathtt{a}\,x\,x:(\{0\},\emptyset,\mathtt{o})\triangleright 0\Rightarrow \mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})}}}{\emptyset\vdash_2 \lambda x.\mathtt{a}\,x\,x:(\{0\},\emptyset,(\{0\},\emptyset,\mathtt{o})\to\mathtt{o})\triangleright 0\Rightarrow \lambda x_{(\{0\},\emptyset,\mathtt{o})}.\mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})}} \quad \dfrac{\emptyset\vdash_2 \mathtt{c}:(\{0\},\emptyset,\mathtt{o})\triangleright 0\Rightarrow\mathtt{c}}{\emptyset\vdash_2 \mathtt{c}:(\{0\},\{1\},\mathtt{o})\triangleright 0\Rightarrow\mathtt{c}}\,\textsc{PTr-Mark}}{\emptyset\vdash_2 (\lambda x.\mathtt{a}\,x\,x)\mathtt{c}:(\{0\},\{1\},\mathtt{o})\triangleright 0\Rightarrow (\lambda x_{(\{0\},\emptyset,\mathtt{o})}.\mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})})\mathtt{c}}$$

However, $\mathcal{D}_1{}^!$ is:

$$\dfrac{\dfrac{\dfrac{\cdots}{x:\mathtt{o}^\omega\vdash_{\mathtt{lin}} \mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})}:\mathtt{o}^\omega}}{\emptyset\vdash_{\mathtt{lin}} \lambda x_{(\{0\},\emptyset,\mathtt{o})}.\mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})}:(\mathtt{o}^\omega\to\mathtt{o}^\omega)^\omega} \quad \dfrac{\emptyset\vdash_{\mathtt{lin}}\mathtt{c}:\mathtt{o}^\omega}{\emptyset\vdash_{\mathtt{lin}}\mathtt{c}:\mathtt{o}^1}}{\emptyset\vdash_{\mathtt{lin}} (\lambda x_{(\{0\},\emptyset,\mathtt{o})}.\mathtt{a}\,x_{(\{0\},\emptyset,\mathtt{o})}\,x_{(\{0\},\emptyset,\mathtt{o})})\mathtt{c}:\mathtt{o}^1}$$

which is not admissible. Note that the argument type does not match in the last inference step. In the next subsection (in Theorem 35), we show that if $\pi\in\mathcal{L}(\mathcal{G})$, we can construct a derivation $\mathcal{D}$ such that $\mathcal{D}^!$ is an admissible derivation.

A.3.2 Completeness

We write $\mathcal{D}\models\Theta\vdash_n t:\gamma\triangleright c\Rightarrow s$ if $\mathcal{D}$ is a derivation tree for $\Theta\vdash_n t:\gamma\triangleright c\Rightarrow s$.

Here we show the following theorem:

▶ **Theorem 35** (completeness). *Let $\mathcal{G}$ be an order-n grammar $\mathcal{G}$, and let S be its start symbol. If $\pi\in\mathcal{L}(\mathcal{G})$, then $\mathcal{D}\models\emptyset\vdash_n S:(\emptyset,\{0,\dots,n-1\},\mathtt{o})\triangleright c\Rightarrow s$ for some $\mathcal{D}$ and c such that $\mathbf{exp}_n(c)\geq|\pi|$. Furthermore, $\mathcal{D}^!$ is an admissible derivation in the linear type system.*

The theorems follow from the following three lemmas.

▶ **Lemma 36** (base case). *If $t\longrightarrow_{\mathtt{c}}^{*}\pi$, then $\mathcal{D}\models\emptyset\vdash_1 t:(\emptyset,\{0\},\mathtt{o})\triangleright c\Rightarrow\pi$ for some c such that $2^c\geq\max(|\pi|,1)$. Furthermore, $\mathcal{D}^!$ is an admissible derivation.*

▶ **Lemma 37** (subject expansion for closed, ground-type terms). *If $t\longrightarrow_n t'$ and $\mathcal{D}'\models\emptyset\vdash_n t':(\emptyset,\{0,\dots,n-1\},\mathtt{o})\triangleright c\Rightarrow s'$, then $\mathcal{D}\models\emptyset\vdash_n t:(\emptyset,\{0,\dots,n-1\},\mathtt{o})\triangleright c\Rightarrow s$ for some $\mathcal{D}$ and s. Furthermore, if $\mathcal{D}'^!$ is an admissible derivation, so is $\mathcal{D}^!$.*

▶ **Lemma 38** (increase of order). *If $\mathcal{D}'\models\emptyset\vdash_n t:(\emptyset,\{0,\dots,n-1\},\mathtt{o})\triangleright c'\Rightarrow s$ and $c'>0$, then $\mathcal{D}\models\emptyset\vdash_{n+1} t:(\emptyset,\{0,\dots,n\},\mathtt{o})\triangleright c\Rightarrow s$ for some $\mathcal{D}$, and c such that $2^c\geq c'$ and $c>0$. Furthermore, if $\mathcal{D}'^!$ is an admissible derivation, so is $\mathcal{D}^!$.*

Proof of Theorem 35. Suppose $S \longrightarrow^*_{\mathcal{G}} \pi$, then

$$S = t_n \longrightarrow^*_n t_{n-1} \longrightarrow^*_{n-1} \cdots \longrightarrow^*_1 t_0 \longrightarrow^*_{\mathrm{c}} \pi.$$

By Lemma 36, we have

$$\mathcal{D}_0 \models \emptyset \vdash_1 t_0 : (\emptyset, \{0\}, \mathtt{o}) \triangleright c_0 \Rightarrow \pi$$

for some $\mathcal{D}_0$ and c_0 such that $2^{c_0} \geq |\pi|$ and ${\mathcal{D}_0}^!$ is admissible. By repeated applications of Lemmas 37 and 38, we have $\mathcal{D} \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathtt{o}) \triangleright c \Rightarrow s$ for some $\mathcal{D}$ and c such that $\mathbf{exp}_n(c) \geq |\pi|$ and $\mathcal{D}^!$ is admissible. ☐

We prove the three lemmas above in the rest of this subsection.

Proof of Lemma 36

Lemma 36 is a trivial corollary of the following two lemmas.

▶ **Lemma 39.** *Let π be a tree (i.e., a term consisting of only tree constructors). Then*

1. $\mathcal{D} \models \emptyset \vdash_1 \pi : (\emptyset, \{0\}, \mathtt{o}) \triangleright c \Rightarrow \pi$ *for some $\mathcal{D}$ and c such that $2^c \geq \max(|\pi|, 1)$ and $\mathcal{D}^!$ is an admissible derivation.*
2. $\mathcal{D}' \models \emptyset \vdash_1 \pi : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0 \Rightarrow \pi$. *for some $\mathcal{D}$ such that $\mathcal{D}^!$ is an admissible derivation.*

Proof. The proof proceeds by induction on the structure of π. Since the second property is trivial, we discuss only the first property. Suppose $\pi = a\, \pi_1 \cdots \pi_k$ where $k = \mathtt{ar}(a)$ may be 0. By the induction hypothesis,

$$\begin{array}{ll} \mathcal{D}_i \models \emptyset \vdash_1 \pi_i : (\emptyset, \{0\}, \mathtt{o}) \triangleright c_i \Rightarrow \pi_i & \quad 2^{c_i} \geq \max(|\pi_i|, 1) \\ \mathcal{D}'_i \models \emptyset \vdash_1 \pi_i : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0 \Rightarrow \pi_i. & \end{array}$$

for each $i \in \{1, \dots, k\}$. If $k > 0$, then pick $j \in \{1, \dots, k\}$ such that $c_j = \max(c_1, \dots, c_k)$. Then, we have the following derivation $\mathcal{D}$:

$$\dfrac{\mathcal{D}''_1 \quad \cdots \quad \mathcal{D}''_k}{\emptyset \vdash_1 \pi : (\emptyset, \{0\}, \mathtt{o}) \triangleright c \Rightarrow \pi}\ \textsc{PTr-Const}$$

where $c = c_j + k$, $\mathcal{D}''_j = \mathcal{D}_j$ and $\mathcal{D}''_i = \mathcal{D}'_i$ for $i \in \{1, \dots, k\} \setminus \{j\}$. We have

$$\begin{array}{l} 2^c - |\pi| = 2^{c_j + k} - (1 + |\pi_1| + \cdots + |\pi_k|) \\ \geq 2^{c_j + k} - (1 + 2^{c_1} + \cdots + 2^{c_k}) \geq 2^{c_j} \cdot 2^k - (1 + k2^{c_k}) \\ \geq 2^{c_j} \cdot (k+1) - (1 + k2^{c_k}) \qquad (\text{by } 2^k \geq k+1) \\ = 2^{c_j} - 1 \geq 0. \end{array}$$

If $k = 0$, then we have the following derivation $\mathcal{D}$:

$$\dfrac{\dfrac{}{\emptyset \vdash_1 a : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0 \Rightarrow \pi}\ \textsc{PTr-Const}}{\emptyset \vdash_1 a : (\emptyset, \{0\}, \mathtt{o}) \triangleright 1 \Rightarrow \pi}\ \textsc{PTr-Mark}$$

as required. ☐

▶ **Lemma 40.** *If $\mathcal{D}' \vdash_n t' : \gamma \triangleright c \Rightarrow s$ and $t \longrightarrow_{\mathrm{c}} t'$, then $\mathcal{D} \vdash_n t : \gamma \triangleright c \Rightarrow s$ for some $\mathcal{D}$. Furthermore if ${\mathcal{D}'}^!$ is admissible, so is $\mathcal{D}^!$.*

Proof. This follows by straightforward induction on the structure of the context used for deriving $t \longrightarrow_{\mathrm{c}} t'$. ☐

Proof of Lemma 37

▶ **Lemma 41** (de-substitution). *If* $\mathcal{D} \models \Theta \vdash_n [t_1/x]t_0 : \gamma \triangleright c \Rightarrow s$*, then*

$$\begin{aligned}&\mathcal{D}_0 \models \Theta_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : \gamma \triangleright c_0 \Rightarrow s_0\\&\mathcal{D}_i \models \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\}\\&\Theta = \Theta_0 + \Theta_1 \cup \cdots \cup \Theta_k\\&c = c_0 + c_1 + \cdots + c_k.\end{aligned}$$

Furthermore, if $\mathcal{D}^!$ *is admissible, so are* $\mathcal{D}_0{}^!, \mathcal{D}_1{}^!, \ldots, \mathcal{D}_k{}^!$.

Proof. We first discuss the case where t_0 is a variable. If $t_0 = x$, the required result holds for $k = 1$, $\mathcal{D}_i = \mathcal{D}$ and

$$\mathcal{D}_0 = \frac{}{x : \gamma \vdash_n x : \gamma \triangleright 0 \Rightarrow x_\gamma}\ \text{PTr-Var}.$$

Note that $\mathcal{D}_0{}^! = \frac{}{x_\gamma : \gamma^! \vdash_{\mathtt{lin}} x_\gamma : \gamma^!}$, which is admissible. If $t_0 = y \neq x$, then the required result holds for $k = 0$ and $\mathcal{D}_0 = \mathcal{D}$.

We show the other cases by induction on the derivation tree $\mathcal{D}$, with case analysis on the last rule used.

- Case (PTr-Weak): In this case, we have $\mathcal{D}' \models \Theta' \vdash_n [t_1/x]t_0 : \gamma \triangleright c \Rightarrow s$ and $\Theta = \Theta', x : (F, \emptyset, \rho)$ with $\mathcal{D} = \frac{\mathcal{D}'}{\Theta', y : (F, M, \rho) \vdash_n [t_1/x]t_0 : \gamma \triangleright c \Rightarrow s}$. By the induction hypothesis, we have:

$$\begin{aligned}&\mathcal{D}'_0 \models \Theta'_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : \gamma \triangleright c_0 \Rightarrow s_0\\&\mathcal{D}_i \models \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\}\\&\Theta' = \Theta'_0 + \Theta_1 \cup \cdots \cup \Theta_k\\&c = c_0 + c_1 + \cdots + c_k\end{aligned}$$

  Let $\Theta_0 = \Theta'_0, y{:}(F, \emptyset, \rho)$ and $\mathcal{D}_0 = \frac{\mathcal{D}'_0}{\mathcal{D}'_0 \models \Theta_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : \gamma \triangleright c_0 \Rightarrow s_0}\ \text{PTr-Weak}$.

  Then we have the required result. Note that $\mathcal{D}_0{}^! = \frac{\mathcal{D}'_0{}^!}{\Theta'_0{}^!, y_{(F,\emptyset,\rho)} : (F, \emptyset, \rho)^! \vdash_{\mathtt{lin}} s_0 : (\gamma, c_0)^!}$ is admissible if $\mathcal{D}'_0{}^!$ is so, because $(F, \emptyset, \rho)^!$ is non-linear.
- Case (PTr-Mark): In this case, we have:

$$\begin{aligned}&\mathcal{D}' \models \Theta \vdash_n [t_1/x]t_0 : (F', M', \rho) \triangleright c' \Rightarrow s\\&\mathit{Comp}_n(\{(F', c')\}, M \uplus M') = (F, c)\\&M \subseteq \{j \in F \mid \mathtt{eorder}(t_0) \leq j < n\}\\&\gamma = (F, M \uplus M', \rho)\end{aligned}$$

  By the induction hypothesis, we have:

$$\begin{aligned}&\mathcal{D}'_0 \models \Theta_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : (F', M', \rho) \triangleright c'_0 \Rightarrow s_0\\&\mathcal{D}_i \models \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\}\\&\Theta = \Theta_0 + \Theta_1 + \cdots + \Theta_k\\&c' = c'_0 + c_1 + \cdots + c_k\end{aligned}$$

  Let $\mathcal{D}_0 = \frac{\mathcal{D}'_0}{\Theta_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s_0}\ \text{PTr-Mark}$, where $\mathit{Comp}_n(\{(F', c'_0)\}, M \uplus M') = (F, c_0)$. It remains to check that $c = c_0 + c_1 + \cdots + c_k$. By the definition of Comp_n, we have $c_0 - c'_0 = c - c_0$. Thus, we have $c = c_0 + c_1 + \cdots + c_k$ as required.

- Case (PTr-Var): In this case t_0 must be a variable, which has been discussed already.
- Case (PTr-Choice): Trivial by the induction hypothesis.
- Case (PTr-Abs): In this case, we have:

$$\begin{array}{l} t_0 = \lambda y.t'_0 \\ \gamma = (F, M \setminus \mathit{Markers}(\xi)), \xi \to \rho) \\ \mathcal{D}' \models \Theta, y:\xi \vdash_n [t_1/x]t'_0 : (F, M, \rho) \triangleright c \Rightarrow s' \\ s = \lambda \tilde{x}_\xi.s' \end{array}$$

By the induction hypothesis, we have:

$$\begin{array}{l} \mathcal{D}'_0 \models \Theta_0, y:\xi_0, x:\{\gamma_1,\dots,\gamma_k\} \vdash_n t'_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s_0 \\ \mathcal{D}'_i \models \Theta_i, y:\xi_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for } i \in \{1,\dots,k\} \\ \Theta = \Theta_0 + \cdots + \Theta_k \\ \xi = \xi_0 + \cdots + \xi_k \\ c = c_0 + \cdots + c_k \end{array}$$

By the convention on bound variables, we can assume that y does not occur in t_1. We have thus $\xi_i \le \emptyset$ for each $i \in \{1,\dots,k\}$. Thus by repeated applications of weakening and strengthening, we have:

$$\begin{array}{l} \mathcal{D}''_0 \models \Theta_0, y:\xi, x:\{\gamma_1,\dots,\gamma_k\} \vdash_n t'_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s_0 \\ \mathcal{D}_i \models \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for } i \in \{1,\dots,k\} \end{array}$$

The required result holds for:

$$\mathcal{D}_0 = \frac{\mathcal{D}''_0}{\Theta_0, x:\{\gamma_1,\dots,\gamma_k\} \vdash_n \lambda y.t'_0 : (F, M \setminus \mathit{Markers}(\xi), \xi \to \rho) \triangleright c_0 \Rightarrow \lambda \widetilde{y}_\xi.s_0}.$$

To check that ${\mathcal{D}_0}^!$ is admissible if so is $\mathcal{D}^!$, it suffices to check that the last consecutive applications of abstractions to obtain the typing of $\lambda\widetilde{y}_\xi.s_0$ is valid. Suppose that $\xi = \{\gamma'_1,\dots,\gamma'_\ell\}$ with $\gamma'_1 < \cdots < \gamma'_\ell$, and that $M \setminus (\mathit{Markers}(\gamma'_{j+1}) \uplus \cdots \uplus \mathit{Markers}(\gamma'_k)) = \emptyset$. We need to show $(\Theta_0, y:\{\gamma'_1,\dots,\gamma'_j\}, x:\{\gamma_1,\dots,\gamma_k\})^!$ is non-linear. By $\mathcal{D}''_0$ and Lemma 25, we have:

$$\mathit{Markers}(\Theta_0, x:\{\gamma_1,\dots,\gamma_k\}) \uplus \mathit{Markers}(\xi) \subseteq M.$$

Thus, we have

$$\mathit{Markers}(\Theta_0, y{:}\{\gamma'_1,\dots,\gamma'_j\}, x{:}\{\gamma_1,\dots,\gamma_k\}) \subseteq M \setminus (\mathit{Markers}(\gamma'_{j+1}) \uplus \cdots \uplus \mathit{Markers}(\gamma'_k)) = \emptyset$$

as required.
- Case (PTr-App): In this case, we have:

$$\begin{array}{l} \mathtt{eorder}(t_{0,0}) = \ell \qquad t_0 = t_{0,0}t_{0,1} \\ \mathcal{D}'_0 \models \Theta'_0 \vdash_n [t_1/x]t_{0,0} : (F'_0, M'_0, \{(F'_1, M'_1, \rho_1),\dots,(F'_p, M'_p, \rho_p)\} \to \rho) \triangleright c'_0 \Rightarrow s'_0 \\ \mathcal{D}'_i \models \Theta'_i \vdash_n [t_1/x]t_{0,1} : (F''_i, M''_i, \rho_i) \triangleright c'_i \Rightarrow s'_i \quad F''_i\restriction_{<\ell} = F'_i \quad M''_i\restriction_{<\ell} = M'_i \text{ for each } i \in \{1,\dots,p\} \\ M = M'_0 \uplus M''_1 \uplus \cdots \uplus M''_p \\ \mathit{Comp}_n(\{(F'_0, c'_0)\} \cup \{(F''_i\restriction_{\ge\ell}, c'_i) \mid i \in \{1,\dots,p\}\}, M) = (F, c) \\ \gamma = (F, M, \rho) \end{array}$$

By the induction hypothesis, we have:

$$\begin{array}{l} \mathcal{D}'_{0,0} \models \Theta'_{0,0}, x:\{\gamma_{0,1},\dots,\gamma_{0,k_0}\} \vdash_n t_{0,0} : (F'_0, M'_0, \{(F'_1, M'_1, \rho_1),\dots,(F'_p, M'_p, \rho_p)\} \to \rho) \triangleright c_{0,0} \Rightarrow s''_0 \\ \mathcal{D}'_{i,0} \models \Theta'_{i,0}, x:\{\gamma_{i,1},\dots,\gamma_{0,k_i}\} \vdash_n t_{0,1} : (F''_i, M''_i, \rho_i) \triangleright c_{i,0} \Rightarrow s''_i \\ \mathcal{D}'_{i,j} \models \Theta'_{i,j} \vdash_n t_1 : \gamma_{i,j} \triangleright c_{i,j} \Rightarrow s_{i,j} \text{ for each } i \in \{0,\dots,p\}, j \in \{1,\dots,k_i\} \\ c'_i = \Sigma_{j\in\{0,\dots,k_i\}} c_{i,j} \\ \Theta'_i = \Sigma_{j\in\{0,\dots,k_i\}} \Theta_{i,j} \end{array}$$

Let $\{\gamma_{i,j} \mid i \in \{0,\dots,p\}, j \in \{1,\dots,k_i\}\} = \{\gamma_1,\dots,\gamma_k\}$ with $\gamma_1 < \cdots < \gamma_k$. For each $q \in \{1,\dots,k\}$, we pick a pair (i_q, j_q) such that $\gamma_{i'} = \gamma_{i,j}$, and write I for $\{(i,j) \mid i \in \{0,\dots,p\}, j \in \{1,\dots,k_i\}\} \setminus \{(i_q,j_q) \mid q \in \{1,\dots,k\}\}$. Let $\Theta_0 = (\Sigma_{i\in\{0,\dots,p\}}\Theta'_{i,0}) + (\Sigma_{(i,j)\in I}\Theta'_{i,j})$. Note that Θ_0 is well defined, because $\Theta'_{i,j}$ contains no markers for each $(i,j) \in I$. Let Θ_q and c_q be Θ_{i_q,j_q} and c_{i_q,j_q} respectively for each $q \in \{1,\dots,k\}$. Let $\mathcal{D}_0$ be:

$$\dfrac{\dfrac{\mathcal{D}'_{0,0} \quad \mathcal{D}'_{1,0} \quad \cdots \quad \mathcal{D}'_{p,0}}{(\Sigma_{i\in\{0,\dots,p\}}\Theta'_{i,0}), x : \{\gamma_1,\dots,\gamma_k\} \vdash_n t_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s''_0 s''_1 \cdots s''_p}\ \text{PTr-App}}{\dfrac{\cdots}{\Theta_0, x : \{\gamma_1,\dots,\gamma_k\} \vdash_n t_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s''_0 s''_1 \cdots s''_p}\ \text{PTr-Weak}}\ \text{PTr-Weak}$$

where $c - c_0 = \Sigma_{i\in\{0,\dots,p\}}(c'_i - c_{i,0})$. Let $\mathcal{D}_q$ be $\mathcal{D}'_{(i_p,j_p)}$. Then we have the required result. The condition $c = c_0 + c_1 + \cdots + c_p$ is verified by:

$$\begin{aligned} c - c_0 &= \Sigma_{i\in\{0,\dots,p\}}(c'_i - c_{i,0}) = \Sigma_{i\in\{0,\dots,p\},j\in\{1,\dots,q\}}c_{i,j} \\ &= (\Sigma_{q\in\{1,\dots,k\}}c_{(i_q,j_q)}) + (\Sigma_{(i,j)\in I}c_{i,j}) = (\Sigma_{q\in\{1,\dots,k\}}c_{(i_q,j_q)}) = c_1 + \cdots + c_q \end{aligned}$$

Note that $c_{i,j} = 0$ for $(i,j) \in I$.

- Case (PTr-Const): Similar to the case for (PTr-App) above.
- Case (PTr-NT): In this case, t_0 does not contain a variable. Thus, the result follows immeidately fro $k = 0$ and $\mathcal{D}_0 = \mathcal{D}$.

□

Lemma 37 is a special case of the following lemma

▶ **Lemma 42** (subject expansion). *If $t \longrightarrow_n t'$ and $\mathcal{D}' \models \Theta \vdash_n t' : \gamma \triangleright c \Rightarrow s'$, then $\mathcal{D} \models \Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ for some $\mathcal{D}$ and s. Furthermore, if $\mathcal{D}'^!$ is an admissible derivation, so is $\mathcal{D}^!$.*

Proof. This follows by induction on the context used for deriving $t \longrightarrow_n t'$. Since the induction steps are trivial, we discuss only the base case, where the context is $[\,]$. In this case, either $t = A$ with $A = t' \in \mathcal{G}$, or $t = (\lambda x.t_0)t_1$ with $t' = [t_1/x]t_0$. In the former case, the result holds for $\mathcal{D} = \dfrac{\mathcal{D}'}{\Theta \vdash_n A : \gamma \triangleright c \Rightarrow s'}$ and $s = s'$.

In the latter case, by Lemma 41, we have:

$$\begin{aligned} &\mathcal{D}_0 \models \Theta_0, x : \{\gamma_1,\dots,\gamma_k\} \vdash_n t_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s_0 \\ &\mathcal{D}_i \models \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \\ &c = c_0 + c_1 + \cdots + c_k \\ &\gamma_1 < \cdots < \gamma_k \\ &\Theta = \Theta_0 + (\textstyle\sum_{i\in\{1,\dots,k\}} \Theta_i) \end{aligned}$$

where $\gamma = (F, M, \rho)$. Furthermore, $\mathcal{D}_0{}^!, \dots, \mathcal{D}_k{}^!$ are admissible if $\mathcal{D}_i{}^!$ is. By (PTr-Abs), we have:

$$\Theta_0 \vdash_n \lambda x.t_0 : (F, M \setminus (\biguplus_i \mathit{Markers}(\gamma_i)), \{\gamma_1,\dots,\gamma_k\} \to \rho) \triangleright c_0 \Rightarrow \lambda x_{\gamma_1}.\cdots\lambda x_{\gamma_k}.s_0.$$

Since $\texttt{eorder}(\lambda x.t_0) = n$, by applying the following special case of (PTr-App):

$$\frac{\begin{array}{c}\texttt{eorder}(t_0) = n \\ \Theta_0 \vdash_n t_0' : (F_0, M_0, \{\gamma_1, \ldots, \gamma_k\} \to \rho) \triangleright c_0 \Rightarrow s_0 \\ \Theta_i \vdash_n t_1 : \gamma_i \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\}, j \in J_i \\ M' = M_0 \uplus (\biguplus_{i \in \{1,\ldots,k\}} \mathit{Markers}(\gamma_i)) \\ \mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\texttt{mul}} \{(\emptyset, c_i) \mid i \in \{1, \ldots, k\}\}, M') = (F', c') \\ \gamma_1 < \cdots < \gamma_k\end{array}}{\Theta_0 + (\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n t_0' t_1 : (F', M', \rho) \triangleright c \Rightarrow s_0\, s_1\, \cdots\, s_k}$$

(PTr-App-Order-n)

we obtain

$$\Theta \vdash_n (\lambda x.t_0)t_1 : (F', M', \rho) \triangleright c' \Rightarrow (\lambda x_{\gamma_1}. \cdots \lambda x_{\gamma_k}.s_0)s_1 \cdots s_k$$

for M', F', c' such that:

$$M' = (M \setminus (\biguplus_i \mathit{Markers}(\gamma_i))) \uplus (\biguplus_{i \in \{1,\ldots,k\}} \mathit{Markers}(\gamma_i))$$
$$\mathit{Comp}_n(\{(F, c_0)\} \cup_{\texttt{mul}} \{(\emptyset, c_i) \mid i \in \{1, \ldots, k\}\}, M') = (F', c')$$

By Lemma 25, $\biguplus_i \mathit{Markers}(\gamma_i) \subseteq M$. Therefore, $M = M'$ follows immediately from $M' = (M \setminus (\biguplus_i \mathit{Markers}(\gamma_i))) \uplus (\biguplus_{i \in \{1,\ldots,k\}} \mathit{Markers}(\gamma_i))$. By the condition $\Theta_0, x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t_0 : (F, M, \rho) \triangleright c_0 \Rightarrow s_0$, we have $F \cap M = \emptyset$. Thus, by Lemma 26, we have also $F = F'$ and $c = c'$. Therefore,

$$\mathcal{D} \models \Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$$

for $s = (\lambda x_{\gamma_1}. \cdots \lambda x_{\gamma_k}.s_0)s_1 \cdots s_k$ and

$$\mathcal{D} = \frac{\dfrac{\mathcal{D}_0}{\Theta_0 \vdash_n \lambda x.t_0 : (F, M \setminus (\biguplus_i \mathit{Markers}(\gamma_i)), \{\gamma_1, \ldots, \gamma_k\} \to \rho) \triangleright c_0 \Rightarrow \lambda x_{\gamma_1}. \cdots \lambda x_{\gamma_k}.s_0} \quad \mathcal{D}_1 \quad \cdots \quad \mathcal{D}_k}{\Theta \vdash_n t : \gamma \triangleright c \Rightarrow s}$$

If ${\gamma_i}^! = \beta_i^{m_i}$, then the type δ_i of s_i in the conclusion of $\mathcal{D}_i$ is β_i^1 if $c_i > 0$ and $\beta_i^{m_i}$ otherwise. In the former case, $n-1 \in \mathit{Markers}(\gamma_i)$ by Lemma 24. So, $\delta_i = {\gamma_i}^!$ in any case. Thus, if $\mathcal{D}'$ is admissible (which implies $\mathcal{D}_0, \ldots, \mathcal{D}_k$ are also admissible), then $\mathcal{D}$ is also admissible. ☐

Proof of Lemma 38

▶ **Lemma 43.** *If $\mathcal{D}' \models \Theta \vdash_n t : (F, M, \rho) \triangleright c' \Rightarrow s$, then $\mathcal{D} \models \Theta \vdash_{n+1} t : (F', M, \rho) \triangleright 0 \Rightarrow s$ for some $\mathcal{D}$, where $F' = F \cup \{n\}$ if $c' > 0$ and $F' = F$ otherwise. Furthermore, if ${\mathcal{D}'}^!$ is an admissible derivation, so is $\mathcal{D}^!$.*

Proof. This follows by induction on the derivation tree $\mathcal{D}'$. We discuss below only the case where the last rule is (PTr-Abs), since the other cases follow immediately from the induction hypothesis. In this case, we have $t = \lambda x.t_0$ and:

$$\mathcal{D}_0' \models \Theta, x : \xi \vdash_n t_0 : (F, M_0, \rho') \triangleright c' \Rightarrow s_0$$
$$M = M_0 \setminus \mathit{Markers}(\xi) \qquad \rho = \xi \to \rho' \qquad s = \lambda \widetilde{x}_\xi.s_0$$

By the induction hypothesis, we have:

$$\mathcal{D}_0 \models \Theta, x : \xi \vdash_{n+1} t_0 : (F', M_0, \rho') \triangleright 0 \Rightarrow s_0$$

Thus, we have

$$\mathcal{D} = \frac{\mathcal{D}_0}{\mathcal{D} \models \Theta \vdash_{n+1} t : (F', M, \rho) \triangleright 0 \Rightarrow s.}\ \textsc{PTr-Abs}$$

It remains to check that the last step of $\mathcal{D}^!$ is admissible. Suppose $\xi = \{\gamma_1, \ldots, \gamma_k\}$ with $\gamma_1 < \cdots < \gamma_k$. It suffices to check that if $(\Theta, x : \gamma_1, \ldots, x : \gamma_i)^!$ contains a linear type, i.e., $\mathit{Markers}(\Theta, x : \gamma_1, \ldots, x : \gamma_i) \neq \emptyset$, then $(F', M \uplus \mathit{Markers}(\xi_1) \uplus \cdots \uplus \mathit{Markers}(\xi_i), \rho)^!$ is linear, i.e., $M \uplus \mathit{Markers}(\xi_1) \uplus \cdots \uplus \mathit{Markers}(\xi_i) \neq \emptyset$. This follows from $\mathit{Markers}(\Theta) \subseteq M$ (by Lemma 25). ☐

Lemma 38 is a special case of the following lemma.

▶ **Lemma 44.** *If $\mathcal{D}' \models \Theta \vdash_n t : (F, M, \rho) \triangleright c' \Rightarrow s$ and $c' > 0$, then $\mathcal{D} \models \Theta \vdash_{n+1} t : (F, M \uplus \{n\}, \rho) \triangleright c \Rightarrow s$ for some $\mathcal{D}$, and c such that $2^c \geq c'$ and $c > 0$. Furthermore, if $\mathcal{D}'^!$ is an admissible derivation, so is $\mathcal{D}^!$.*

Proof. Note that, by the well-formedness condition, the orders of types in Θ are at most $n - 1$, and $\texttt{eorder}(t) \leq n$. The proof proceeds by induction on the derivation tree $\mathcal{D}'$, with the case analysis on the last rule used. The cases for (PTr-Weak), (PTr-Choice), and (PTr-NT) follow immediately from the induction hypothesis. We discuss the other cases below.

- Case (PTr-Mark): In this case, we have:

  $\mathcal{D}'_0 \models \Theta \vdash_n t : (F_0, M_0, \rho) \triangleright c'_0 \Rightarrow s$
  $M_1 \subseteq \{j \in F_0 \mid \texttt{eorder}(t) \leq j < n\}$
  $M = M_0 \uplus M_1$
  $\mathit{Comp}_n(\{(F_0, c'_0)\}, M) = (F, c')$

  We need to consider only the case where $M_1 \neq \emptyset$, hence $\texttt{eorder}(t) \leq n - 1$. By the assumption $c' > 0$, if $c'_0 = 0$, then $c' = 1$ and $n - 1 \in M_1 \cap F_0$. By Lemma 43, we have

  $$\mathcal{D}_0 \models \Theta \vdash_{n+1} t : (F_0, M_0, \rho) \triangleright 0 \Rightarrow s$$

  By applying (PTr-Mark) (to add markers $M_1 \uplus \{n\}$), we have the following derivation $\mathcal{D}$:

  $$\frac{\mathcal{D}_0}{\Theta \vdash_{n+1} t : (F, M \uplus \{n\}, \rho) \triangleright 1 \Rightarrow s}\ \textsc{PTr-Mark}$$

  as required.

  If $c'_0 > 0$, then by the induction hypothesis, we have:

  $$\mathcal{D}_0 \models \Theta \vdash_{n+1} t : (F_0, M_0 \uplus \{n\}, \rho) \triangleright c_0 \Rightarrow s$$

  for some $\mathcal{D}_0$ and c_0 such that $2^{c_0} \geq c'_0$. Let $\mathcal{D}$ be:

  $$\frac{\mathcal{D}_0}{\Theta \vdash_{n+1} t : (F', M \uplus \{n\}, \rho) \triangleright c \Rightarrow s}\ \textsc{PTr-Mark},$$

  where $(F', c) = \mathit{Comp}_{n+1}(\{(F_0, c_0)\}, M \uplus \{n\})$. Since F_0 and $M_0 \uplus \{n\}$ are disjoint, $n \notin F'$. Thus, $F' = F$. We check the condition $2^c \geq c'$. Since $c' > 0$, by Lemma 24, we have $n - 1 \in M$. Let f_n, f'_n, f'_{n+1} be those that occur in the calculation of $\mathit{Comp}_{n+1}(\{(F_0, c_0)\}, M \uplus \{n\})$.

Note that f'_n is the same as the one that occurs in the calculation of $Comp_n(\{(F_0, c'_0)\}, M) = (F, c')$. Thus, we have $c = f'_{n+1} + c_0 = f_n + c_0 = f'_n + c_0$ and $c' = f'_n + c'_0$. Therefore, we have:

$$2^c - c' = 2^{f'_n + c_0} - (f'_n + c'_0) = 2^{f'_n} 2^c_0 - (f'_n + c'_0) = (2^{f'_n} - 1)2^{c_0} - f'_n + (2^{c_0} - c'_0) \geq 2^{f'_n} - 1 - f'_n \geq 0.$$

It remains to check that the admissibility of $\mathcal{D}'^!$ implies that of $\mathcal{D}^!$. If $\mathcal{D}'^!$ is admissible, so is $\mathcal{D}'_0$. By the induction hypothesis, $\mathcal{D}_0{}^!$ is also admissible. Thus, $\mathcal{D}^!$ is also admissible. Note that the last step in $\mathcal{D}^!$ does not change the judgment.

- Case (PTr-Var): This case does not occur, since the flag counter is 0.
- Case (PTr-Abs): In this case, we have $t = \lambda x.t_0$ and:

$$\mathcal{D}'_0 \models \Theta, x : \xi \vdash_n t_0 : (F, M_0, \rho_0) \triangleright c' \Rightarrow s'_0$$
$$M = M_0 \setminus \mathit{Markers}(\xi) \qquad \rho = \xi \to \rho_0$$
$$\mathcal{D}' = \cfrac{\mathcal{D}'_0}{\Theta \vdash_n t : (F, M, \rho) \triangleright c' \Rightarrow \lambda \widetilde{x}_\xi.s'_0}\ \text{PTr-Abs}$$

By the induction hypothesis, we have

$$\mathcal{D}_0 \models \Theta, x : \xi \vdash_{n+1} t_0 : (F, M_0 \uplus \{n\}, \rho_0) \triangleright c \Rightarrow s_0$$

for some $\mathcal{D}_0, c, s_0$ such that $2^c \geq c'$. By using (PTr-Abs), we have the required derivation:

$$\mathcal{D} = \cfrac{\mathcal{D}'}{\Theta \vdash_{n+1} t : (F, M \uplus \{n\}, \rho) \triangleright c \Rightarrow \lambda \widetilde{x}_\xi.s_0}\ \text{PTr-Abs}$$

To check the admissibility condition, suppose $\mathcal{D}'^!$ is admissible. Then so is $\mathcal{D}'_0{}^!$; hence $\mathcal{D}_0{}^!$ is also admissible by the induction hypothesis. It remains to check that the last step of $\mathcal{D}^!$ is admissible, which is indeed the case, since a linear type is assigned to $\lambda \widetilde{x}_\xi.s_0$ (recall $c' > 0$).

- Case (PTr-App): In this case, we have:

$$t = t_0 t_1 \qquad \mathtt{eorder}(t_0) = \ell$$
$$\mathcal{D}'_0 \models \Theta_0 \vdash_n t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c'_0 \Rightarrow s_0$$
$$\mathcal{D}'_i \models \Theta_i \vdash_n t_1 : (F'_i, M'_i, \rho_i) \triangleright c'_i \Rightarrow s_i \qquad F_i = F'_i \restriction_{<\ell} \qquad M_i = M'_i \restriction_{<\ell}$$
$$\text{(for each } i \in \{1, \ldots, k\})$$
$$M = M_0 \uplus \cdots \uplus M_k$$
$$Comp_n(\{(F_0, c'_0)\} \cup \{(F'_i \restriction_{\geq \ell}, c'_i) \mid i \in \{1, \ldots, k\}\}, M) = (F, c')$$
$$s = s_0 s_1 \cdots s_k$$
$$\Theta = \Theta_0 + \cdots + \Theta_k$$

We consider two cases:

  - Case where $c'_i = 0$ for every $i \in \{0, \ldots, k\}$: By Lemma 43, we have:

$$\mathcal{D}''_0 \models \Theta_0 \vdash_{n+1} t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright 0 \Rightarrow s_0$$

and

$$\mathcal{D}''_i \models \Theta_i \vdash_{n+1} t_1 : (F'_i, M'_i, \rho_i) \triangleright 0 \Rightarrow s_i$$

for each $i \in \{1, \ldots, k\}$. By the conditions $Comp_n(\{(F_0, c'_0)\} \cup \{(F'_i \restriction_{\geq \ell}, c'_i) \mid i \in \{1, \ldots, k\}\}, M) = (F, c')$ and $c' > 0$, it must be the case that $n - 1 \in M$, hence $n - 1 \in M_j$ for some $j \in \{0, \ldots, k\}$. By applying (PTr-Mark) to $\mathcal{D}''_j$, we obtain a derivation $\mathcal{D}_j$, whose conclusion is the same as that of $\mathcal{D}''_j$ except that marker n has been added. Note that the last step of $\mathcal{D}_j{}^!$ is admissible (and in fact, does not change the judgment) since $n-1 \in M_j \neq \emptyset$. Let $\mathcal{D}_i$ be $\mathcal{D}''_i$ for $i \in \{0,\dots,k\} \setminus \{j\}$. Let $\mathcal{D}$ be:

$$\frac{\mathcal{D}_0 \quad \mathcal{D}_1 \quad \cdots \quad \mathcal{D}_k}{\Theta \vdash_{n+1} t_0 t_1 : (F', M \uplus \{n\}, \rho) \triangleright c \Rightarrow s_0 s_1 \cdots s_k}\ \text{PTr-App}$$

where

$$\mathit{Comp}_{n+1}(\{(F_0,0)\} \cup \{(F'_1{\restriction}_{\geq \ell},0),\dots,(F'_k{\restriction}_{\geq \ell},0)\}, M \uplus \{n\}) = (F',c)$$

Let f_i, f'_i be those occurring in the calculation of $\mathit{Comp}_{n+1}(\cdots)$ above. Note that f_i $(i \leq n-1)$ and f'_i $(i \leq n)$ are equivalent to those to occurring in the calculation of $\mathit{Comp}_n(\cdots) = (F,c')$. Thus, we have:

$$F' = \{\ell \in \{0,\dots,n\} \mid f_\ell > 0\} \setminus (M \uplus \{n\}) = \{\ell \in \{0,\dots,n-1\} \mid f_\ell > 0\} \setminus M = F.$$

Furthermore, $c = f'_{n+1} = f_n = f'_n = c'$. It remains to check that the admissibility of $\mathcal{D}'^!$ implies that of $\mathcal{D}^!$, which is trivial from the construction of $\mathcal{D}$ above (recall that the last step of $\mathcal{D}_j{}^!$ is admissible).

- Case where $c'_i > 0$ for some $i \in \{0,\dots,k\}$. Pick j such that $c'_j = \max(c'_0,\dots,c'_k)$. Let $(F'_0,,M'_0,\gamma_0)$ be $(F_0, M_0, \{(F_1,M_1,\rho_1),\dots,(F_k,M_k,\rho_k)\} \to \rho)$. For $i \in \{0,\dots,k\}$, let $t'_i = t_0$ if $i=0$, and $t'_i t_1$ otherwise. By applying the induction hypothesis to $\mathcal{D}'_j$, we have:

$$\mathcal{D}_j \models \Theta_j \vdash_{n+1} t'_j : (F'_j, M'_j \uplus \{n\}, \rho_i) \triangleright c_j \Rightarrow s_j.$$

For $i \in \{0,\dots,k\} \setminus \{j\}$, by Lemma 43, we have

$$\mathcal{D}_i \models \Theta_i \vdash_{n+1} t'_i : (F''_i, M'_i, \rho_i) \triangleright 0 \Rightarrow s_i$$

where $F''_i = F'_i \uplus \{n\}$ if $c'_i > 0$ and $F''_i = F'_i$ otherwise. Let $\mathcal{D}$ be:

$$\frac{\mathcal{D}_0 \quad \mathcal{D}_1 \quad \cdots \quad \mathcal{D}_k}{\Theta \vdash_{n+1} t_0 t_1 : (F', M \uplus \{n\}, \rho) \triangleright c \Rightarrow s_0 s_1 \cdots s_k}\ \text{PTr-App}$$

where

$$\mathit{Comp}_{n+1} = (\{(F'_j, c_j)\} \cup_{\mathtt{mul}} \{(F''_i, 0) \mid i \in \{0,\dots,k\} \setminus \{j\}\}, M \uplus \{n\}).$$

If $\mathcal{D}'^!$ is admissible, then the last step of $\mathcal{D}^!$ is also admissible: the change from $\mathcal{D}'_i{}^!$ to $\mathcal{D}_i{}^!$ may change the type of s_i in the conclusion to a non-linear type, but then we can use (LT-Dereliction) to adjust the linearity. Thus, it remains to check $F' = F$ and $2^c \geq c'$. Let f_i, f'_i be those occurring in the calculation of $\mathit{Comp}_{n+1}(\cdots)$ above. Note that f_i $(i \leq n-1)$ and f'_i $(i \leq n)$ are equivalent to those to occurring in the calculation of $\mathit{Comp}_n(\cdots) = (F,c')$. We have:

$$F' = \{\ell \in \{0,\dots,n\} \mid f_\ell > 0\} \setminus (M \uplus \{n\}) = \{\ell \in \{0,\dots,n-1\} \mid f_\ell > 0\} \setminus M = F.$$

We also have:

$$c = f'_{n+1} + c_j = f_n + c_j = f'_n + \{i \mid n \in F''_i\} + c_j = f'_n + (\{i \mid c'_i > 0\} - 1) + c_j.$$

Let $p = \{i \mid c'_i > 0\} - 1$. Then, we have:

$$\begin{aligned}
&2^c - c' \geq 2^c - (f'_n + (p+1)c'_j) && (\text{by } c'_j = \max(c_0, \dots, c_j)) \\
&= 2^{f'_n+p+c_j} - (f'_n + (p+1)c'_j) = 2^{f'_n+p} \cdot 2^{c_j} - (f'_n + (p+1)c'_j) \\
&\geq 2^{f'_n+p} \cdot c'_j - (f'_n + (p+1)c'_j) && (\text{by } 2^{c_j} \geq c'_j) \\
&\geq (f'_n + p + 1) \cdot c'_j - (f'_n + (p+1)c'_j) && (\text{by } 2^x \geq x+1) \\
&= f'_n(c'_j - 1) \geq 0. && (\text{by } c'_j \geq 1)
\end{aligned}$$

- Case (PTr-Const): Similar to the case for (PTr-App) above.

□

A.3.3 Existence of Pumpable Derivation

We are now ready to prove Lemma 19.

Proof of Lemma 19. Let $\mathcal{G}$ be an order-n tree grammar and S be its start symbol. Suppose that $\mathcal{L}(\mathcal{G})$ is infinite. By Theorem 35, for any c', there exists a derivation tree $\mathcal{D}$ for $\emptyset \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathtt{o}) \triangleright c \Rightarrow s$ such that $c \geq c'$ and $\mathcal{D}^!$ is an admissible derivation. We can assume that the derivation does not contain a path in which two judgments $\Theta \vdash_n t : \gamma \triangleright c' \Rightarrow s'$ and $\Theta \vdash_n t : \gamma \triangleright c' \Rightarrow s''$ that are equivalent except the outputs s' and s'', since otherwise we can "shrink" that part without changing the conclusion. Since the number of premises occurring at each node of $\mathcal{D}$ is bounded above by a constant, the height of derivation trees must also be unbounded. Since $\mathcal{D}$ can contain only a subterm occurring in $\mathcal{G}$, and the numbers of possible types and type environments are also bounded above by a constant, for sufficiently large c, the derivation tree $\mathcal{D}$ must contain a path in which three type judgments $\Theta \vdash_n A : \gamma \triangleright c'_0 \Rightarrow s_0$ $\Theta \vdash_n A : \gamma \triangleright c'_1 \Rightarrow s_1$ $\Theta \vdash_n A : \gamma \triangleright c'_2 \Rightarrow s_2$ such that $(c'_2 > c'_1 > c'_0 \geq 0$. By the transformation rules, s_2 and s must be of the forms $C[s_2]$ and $D[s_1]$. Furthermore, $\mathcal{D}^!$ is an admissible derivation, which implies $\emptyset \vdash_{\mathtt{lin}} C[D[s_1]] : \mathtt{o}^1$. Moreover, since $c'_1 > 0$, the flag counters of any sub-terms containing A (of the judgment $\Theta \vdash_n A : \gamma \triangleright c'_1 \Rightarrow s_1$) are non-zero. Thus, by the admissibility of $\mathcal{D}^!$, linear types are assinged to any terms containing s_1 (hence also those containing $D[s_1]$) in the derivation of $\emptyset \vdash_{\mathtt{lin}} C[D[s_1]] : \mathtt{o}^1$. Thus, by Lemma 32, the contexts C and D are linear. Let $c_1 = c'_1$, $c_2 = c'_2 - c'_1$, and $c_3 = c - c'_2$. Then we have the required result. □

A.4 Proof of Lemma 20

Similarly to [23], the soundness (Lemma 20) is proved in three steps (Lemmas 51, 52, and 53 below). For n and a derivation tree $\mathcal{D}$, we write $@_n(\mathcal{D})$ for the number of order-n (PTr-App) used in $\mathcal{D}$, and write $\mathtt{NT}(\mathcal{D})$ for the number of (PTr-NT) used in $\mathcal{D}$.

▶ **Lemma 45** (substitution lemma). *Given*

$$\mathcal{D}_0 \models \Theta_0,\, x : \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \vdash_n t : (F_0, M_0, \rho) \triangleright c_0 \Rightarrow s$$

$$\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i \qquad \textit{for each } i \in \{1, \dots, k\}$$

$$\Theta_0 + (\textstyle\sum_{i \in \{1,\dots,k\}} \Theta_i) \textit{ is well-defined}$$

$$@_n(\mathcal{D}_i) = 0 \qquad \textit{for each } i \in \{0, 1, \dots, k\}$$

$$\mathtt{eorder}(t_1) < n$$

there exists $\mathcal{D}$ *such that* $@_n(\mathcal{D}) = 0$ *and*

$$\mathcal{D} \models \Theta_0 + (\textstyle\sum_{i\in\{1,\dots,k\}} \Theta_i) \vdash_n [t_1/x]t : (F_0, M_0, \rho)$$
$$\triangleright (c_0 + \textstyle\sum_{i\in\{1,\dots,k\}} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i\in\{1,\dots,k\}} s.$$

Proof. The proof proceeds by induction on $\mathcal{D}_0$ with case analysis on the rule used last for $\mathcal{D}_0$.

- Case of (PTr-Weak): Clear.
- Case of (PTr-Mark): Let the last rule of $\mathcal{D}_0$ be:

$$\frac{\begin{array}{c}\mathcal{D}'_0 \models \Theta_0,\, x : \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \vdash_n t : (F', M', \rho) \triangleright c' \Rightarrow s \\ M' \subseteq M_0 \qquad M_0 \setminus M' \subseteq \{j \mid \mathtt{eorder}(t) \leq j < n\} \qquad \mathit{Comp}_n(\{(F', c')\}, M_0) = (F_0, c_0)\end{array}}{\Theta_0,\, x : \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \vdash_n t : (F_0, M_0, \rho) \triangleright c_0 \Rightarrow s}$$

  Then by induction hypothesis for $\mathcal{D}'_0$, we have

$$\mathcal{D}' \models \Theta_0 + (\textstyle\sum_{i\in\{1,\dots,k\}} \Theta_i) \vdash_n [t_1/x]t : (F', M', \rho)$$
$$\triangleright (c' + \textstyle\sum_{i\in\{1,\dots,k\}} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i\in\{1,\dots,k\}} s$$

  with $@_n(\mathcal{D}') = 0$.
  Since $\mathtt{eorder}([t_1/x]t) = \mathtt{eorder}(t)$,

$$M_0 \setminus M' \subseteq \{j \mid \mathtt{eorder}([t_1/x]t) \leq j < n\},$$

  and we can check that $\mathit{Comp}_n(\{(F', c')\}, M_0) = (F_0, c_0)$ implies

$$\mathit{Comp}_n(\{(F', c' + \textstyle\sum_{i\in\{1,\dots,k\}} c_i)\}, M_0) = (F_0, c_0 + \textstyle\sum_{i\in\{1,\dots,k\}} c_i)$$

  by calculation of Comp_n. Hence by (PTr-Mark), we have

$$\mathcal{D} := \frac{\mathcal{D}'}{\begin{array}{l}\Theta_0 + (\sum_{i\in\{1,\dots,k\}} \Theta_i) \vdash_n [t_1/x]t : (F_0, M_0, \rho) \\ \qquad \triangleright (c_0 + \sum_{i\in\{1,\dots,k\}} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i\in\{1,\dots,k\}} s\end{array}}$$

  $@_n(\mathcal{D}) = @_n(\mathcal{D}') = 0$, as required.
- Case of (PTr-Var): In this case, we further perform case analysis on whether the variable t is x or not. In the case $t = x$, let the last rule of $\mathcal{D}_0$ be:

$$\frac{}{x : \gamma \vdash_n x : \gamma \triangleright 0 \Rightarrow x_\gamma}$$

  and then

$$\begin{aligned} \Theta_0 &= \emptyset \\ k &= 1 \\ (F_1, M_1, \rho_1) &= \gamma \\ (F_0, M_0, \rho) &= \gamma \\ c_0 &= 0 \\ s &= x_\gamma. \end{aligned}$$

Now the goal is

$$\Theta_1 \vdash_n t_1 : (F_1, M_1, \rho_1) \triangleright c_1 \Rightarrow s_1 \,,$$

which is just $\mathcal{D}_1$.

Next, in the case $t = y \neq x$, let the last rule of $\mathcal{D}_0$ be:

$$\frac{}{y : \gamma \vdash_n y : \gamma \triangleright 0 \Rightarrow y_\gamma}$$

and then

$$\begin{aligned} \Theta_0 &= y : \gamma \\ k &= 0 \\ (F_0, M_0, \rho) &= \gamma \\ c_0 &= 0 \\ s &= y_\gamma . \end{aligned}$$

Now the goal is

$$y : \gamma \vdash_n y : \gamma \triangleright 0 \Rightarrow y_\gamma \,,$$

which is just $\mathcal{D}_0$ as above.

- Case of (PTr-Choice): Clear.
- Case of (PTr-Abs): Let the last rule of $\mathcal{D}_0$ be:

$$\frac{\mathcal{D}'_0 \models \Theta_0,\, x : \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\}, x' : \xi \vdash_n t' : (F_0, M, \rho') \triangleright c_0 \Rightarrow s'}{\Theta_0,\, x : \{(F_1, M_1, \rho_1), \ldots, (F_k, M_k, \rho_k)\} \vdash_n \lambda x'.t' : (F_0, M \setminus \mathit{Markers}(\xi), \xi \to \rho') \triangleright c_0 \Rightarrow \lambda \widetilde{x'}_\xi . s'}$$

where x' is fresh, and then we have

$$\begin{aligned} t &= \lambda x'.t' \\ M_0 &= M \setminus \mathit{Markers}(\xi) \\ \rho &= \xi \to \rho' \\ s &= \lambda \widetilde{x'}_\xi . s' . \end{aligned}$$

Since x' was chosen as a fresh variable, $(\Theta_0,\, x' : \xi) + (\sum_{i \in \{1,\ldots,k\}} \Theta_i)$ is well-defined. Hence, by induction hypothesis for $\mathcal{D}'_0$, we have

$$\begin{aligned} &\mathcal{D}' \models (\Theta_0,\, x' : \xi) + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n [t_1/x]t' : (F_0, M, \rho') \\ &\qquad \triangleright (c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i) \Rightarrow [s_i / x_{(F_i, M_i, \rho_i)}]_{i \in \{1,\ldots,k\}} s' . \end{aligned}$$

with $@_n(\mathcal{D}') = 0$. Let $\mathcal{D}$ be $\mathcal{D}'$ plus (PTr-Abs), so that we have $@_n(\mathcal{D}) = @_n(\mathcal{D}') = 0$ and

$$\begin{aligned} &\mathcal{D} \models \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n \lambda x'.[t_1/x]t' : (F_0, M \setminus \mathit{Markers}(\xi), \xi \to \rho') \\ &\qquad \triangleright (c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i) \Rightarrow \lambda \widetilde{x'}_\xi . [s_i / x_{(F_i, M_i, \rho_i)}]_{i \in \{1,\ldots,k\}} s' \end{aligned}$$

i.e.,

$$\begin{aligned} &\mathcal{D} \models \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n [t_1/x](\lambda x'.t') : (F_0, M_0, \rho) \\ &\qquad \triangleright (c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i) \Rightarrow [s_i / x_{(F_i, M_i, \rho_i)}]_{i \in \{1,\ldots,k\}} (\lambda x'_{\gamma_1} . \cdots \lambda x'_{\gamma_k} . s'), \end{aligned}$$

as required.

- Case of (PTr-App): Let the last rule of $\mathcal{D}_0$ be:

$$\frac{\begin{array}{c}\mathtt{eorder}(t_0') = \ell' \\ \mathcal{D}_0' \models \Theta_0' \vdash_n t_0' : (F_0', M_0', \{(F_1', M_1', \rho_1'), \dots, (F_{k'}', M_{k'}', \rho_{k'}')\} \to \rho) \triangleright c_0' \Rightarrow s_0' \\ \left.\begin{array}{l}\mathcal{D}_{i'}' \models \Theta_{i'}' \vdash_n t_1' : (F_{i'}'', M_{i'}'', \rho_{i'}') \triangleright c_{i'}' \Rightarrow s_{i'}' \\ F_{i'}''{\restriction_{<\ell'}} = F_{i'}' \qquad M_{i'}''{\restriction_{<\ell'}} = M_{i'}'\end{array}\right\}\text{for each } i' \in \{1, \dots, k'\} \\ M_0 = M_0' \uplus (\biguplus_{i' \in \{1,\dots,k'\}} M_{i'}'') \\ \mathit{Comp}_n(\{(F_0', c_0')\} \cup_{\mathtt{mul}} \{(F_{i'}''{\restriction_{\geq \ell'}}, c_{i'}') \mid i' \in \{1, \dots, k'\}\}, M_0) = (F_0, c_0) \\ (F_1', M_1', \rho_1') < \dots < (F_{k'}', M_{k'}', \rho_{k'}')\end{array}}{\Theta_0' + (\sum_{i' \in \{1,\dots,k'\}} \Theta_{i'}') \vdash_n t_0' t_1' : (F_0, M_0, \rho) \triangleright c_0 \Rightarrow s_0'\, s_1' \, \cdots \, s_{k'}'}$$

and then we have

$$\begin{aligned}&\Theta_0,\, x : \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} = \Theta_0' + (\textstyle\sum_{i' \in \{1,\dots,k'\}} \Theta_{i'}') \\ &t = t_0' t_1' \\ &s = s_0'\, s_1' \, \cdots \, s_{k'}'.\end{aligned}$$

Note that $\ell' < n$ since $@_n(\mathcal{D}_0) = 0$.

Let:

$$\begin{aligned}&\Theta_0' = \Theta_0'',\, x : \xi_0 \\ &\Theta_{i'}' = \Theta_{i'}'',\, x : \xi_{i'} \quad \text{for each } i' \in \{1, \dots, k'\}\end{aligned}$$

where $x \notin \mathit{dom}(\Theta_0'')$ and $x \notin \mathit{dom}(\Theta_{i'}'')$; then

$$\begin{aligned}&\Theta_0 = \Theta_0'' + (\textstyle\sum_{i' \in \{1,\dots,k'\}} \Theta_{i'}'') \\ &\{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} = \xi_0 \cup (\textstyle\bigcup_{i' \in \{1,\dots,k'\}} \xi_{i'}).\end{aligned} \tag{12}$$

Also, we define

$$\begin{aligned}&I_0 := \{i \in \{1, \dots, k\} \mid (F_i, M_i, \rho_i) \in \xi_0\} \\ &I_{i'} := \{i \in \{1, \dots, k\} \mid (F_i, M_i, \rho_i) \in \xi_{i'}\} \quad \text{for each } i' \in \{1, \dots, k'\}.\end{aligned}$$

Then $\{1, \dots, k\} = I_0 \cup (\bigcup_{i' \in \{1,\dots,k'\}} I_{i'})$ and

$$\textstyle\sum_{i \in I} \Theta_i = (\sum_{i \in I_0} \Theta_i) + \sum_{i' \in \{1,\dots,k'\}} (\sum_{i \in I_{i'}} \Theta_i)$$

where note that the right hand side is well-defined.

Now we have

$$\begin{aligned}&\mathcal{D}_0' \models \Theta_0'',\, x : \xi_0 \vdash_n t_0' : (F_0', M_0', \{(F_1', M_1', \rho_1'), \dots, (F_{k'}', M_{k'}', \rho_{k'}')\} \to \rho) \triangleright c_0' \Rightarrow s_0' \\ &\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i \quad \text{for each } i \in I_0 \\ &\Theta_0'' + (\textstyle\sum_{i \in I_0} \Theta_i) \text{ is well-defined (by (12) and the assumption)} \\ &@_n(\mathcal{D}_i) = 0 \quad \text{for each } i \in I_0 \qquad \mathtt{eorder}(t_1) < n.\end{aligned}$$

Hence by induction hypothesis we have

$$\begin{aligned}\mathcal{D}_0'' \models \Theta_0'' + (\textstyle\sum_{i \in I_0} \Theta_i) \vdash_n [t_1/x]t_0' : (F_0', M_0', \{(F_1', M_1', \rho_1'), \dots, (F_{k'}', M_{k'}', \rho_{k'}')\} \to \rho) \\ \triangleright c_0' + (\textstyle\sum_{i \in I_0} c_i) \Rightarrow [s_i / x_{(F_i, M_i, \rho_i)}]_{i \in I_0} s_0'.\end{aligned}$$

with $@_n(\mathcal{D}''_0) = 0$. Also, for each $i' \in \{1,\ldots,k'\}$, we have

$$\mathcal{D}'_{i'} \models \Theta''_{i'}, x : \xi_{i'} \vdash_n t'_1 : (F''_{i'}, M''_{i'}, \rho'_{i'}) \triangleright c'_{i'} \Rightarrow s'_{i'}$$
$$\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i \quad \text{for each } i \in I_{i'}$$
$$\Theta''_{i'} + (\textstyle\sum_{i \in I_{i'}} \Theta_i) \text{ is well-defined (by (12) and the assumption)}$$
$$@_n(\mathcal{D}_i) = 0 \quad \text{for each } i \in I_0 \qquad \mathtt{eorder}(t_1) < n.$$

Hence by induction hypothesis we have

$$\mathcal{D}''_{i'} \models \Theta''_{i'} + (\textstyle\sum_{i \in I_{i'}} \Theta_i) \vdash_n [t_1/x]t'_1 : (F''_{i'}, M''_{i'}, \rho'_{i'})$$
$$\triangleright c'_{i'} + (\textstyle\sum_{i \in I_{i'}} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_{i'}} s'_{i'}.$$

with $@_n(\mathcal{D}''_{i'}) = 0$.

Now we have:

$$\mathtt{eorder}([t_1/x]t'_0) \ (= \mathtt{eorder}(t'_0)) = \ell'$$
$$\mathcal{D}''_0 \models \Theta''_0 + (\textstyle\sum_{i \in I_0} \Theta_i) \vdash_n [t_1/x]t'_0 : (F'_0, M'_0, \{(F'_1, M'_1, \rho'_1), \ldots, (F'_{k'}, M'_{k'}, \rho'_{k'})\} \to \rho)$$
$$\triangleright c'_0 + (\textstyle\sum_{i \in I_0} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_0} s'_0$$
$$\left.\begin{array}{l} \mathcal{D}''_{i'} \models \Theta''_{i'} + (\sum_{i \in I_{i'}} \Theta_i) \vdash_n [t_1/x]t'_1 : (F''_{i'}, M''_{i'}, \rho'_{i'}) \\ \qquad \triangleright (c'_{i'} + (\sum_{i \in I_{i'}} c_i)) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_{i'}} s'_{i'} \\ F''_{i'}\restriction_{<\ell'} = F'_{i'} \qquad M''_{i'}\restriction_{<\ell'} = M'_{i'} \end{array}\right\} \text{for each } i' \in \{1,\ldots,k'\}$$
$$M_0 = M'_0 \uplus (\textstyle\biguplus_{i' \in \{1,\ldots,k'\}} M''_{i'})$$
$$\mathit{Comp}_n\big(\{(F'_0, c'_0 + (\textstyle\sum_{i \in I_0} c_i))\} \cup_{\mathtt{mul}}$$
$$\{(F''_{i'}\restriction_{\geq \ell'}, c'_{i'} + (\textstyle\sum_{i \in I_{i'}} c_i)) \mid i' \in \{1,\ldots,k'\}\},$$
$$M_0\big) = (F_0, c_0 + (\textstyle\sum_{i \in I_0} c_i) + \sum_{i' \in \{1,\ldots,k'\}} (\sum_{i \in I_{i'}} c_i))$$
$$(F'_1, M'_1, \rho'_1) < \cdots < (F'_{k'}, M'_{k'}, \rho'_{k'})$$

where the equation on Comp_n can be easily checked by a direct calculation. Hence we can use (PTr-App), by which we obtain $\mathcal{D}$ with $@_n(\mathcal{D}) = 0$ since $\ell' < n$; and we have

$$\mathcal{D} \models \Theta''_0 + (\textstyle\sum_{i \in I_0} \Theta_i) + (\sum_{i' \in \{1,\ldots,k'\}} (\Theta''_{i'} + (\sum_{i \in I_{i'}} \Theta_i)))$$
$$\vdash_n ([t_1/x]t'_0)([t_1/x]t'_1) : (F_0, M_0, \rho)$$
$$\triangleright \big(c_0 + (\textstyle\sum_{i \in I_0} c_i) + \sum_{i' \in \{1,\ldots,k'\}} (\sum_{i \in I_{i'}} c_i)\big)$$
$$\Rightarrow ([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_0} s'_0)\,([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_1} s'_1) \cdots ([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_{k'}} s'_{k'})$$

i.e., we have

$$\mathcal{D} \models \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \vdash_n [t_1/x]t : (F_0, M_0, \rho)$$
$$\triangleright (c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i) \Rightarrow [s_i/x_{(F_i,M_i,\rho_i)}]_{i \in \{1,\ldots,k\}} s$$

as required, where note that

$$(\textstyle\sum_{i \in I_0} c_i) + \sum_{i' \in \{1,\ldots,k'\}} (\sum_{i \in I_{i'}} c_i) = \sum_{i \in \{1,\ldots,k\}} c_i$$

follows from Lemma 24 applied to $\Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i$, and

$$([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_0} s'_0) = ([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in \{1,\ldots,k\}} s'_0)$$
$$([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in I_{i'}} s'_{i'}) = ([s_i/x_{(F_i,M_i,\rho_i)}]_{i \in \{1,\ldots,k\}} s'_{i'}) \qquad (i' \in \{1,\ldots,k'\})$$

follow from Lemma 22.

- Case of (PTr-Const): This case is analogous to the case of (PTr-App).
- Case of (PTr-NT): This case is clear (note that t' is a closed term for $A = t' \in \mathcal{G}$).

$\square$

▶ **Lemma 46.** *Suppose*

$$n > 0$$
$$\mathcal{D} \models \Theta \vdash_n (\lambda x.t')\, t_1 : (F, M, \rho) \triangleright c \Rightarrow s$$
the last rule of $\mathcal{D}$ *is* (PTr-App)
$$\texttt{eorder}(\lambda x.t') = n$$
$$@_n(\mathcal{D}) = 1$$
$$\texttt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta.$$

Then there exist Θ', v, *and* $\mathcal{D}'$ *such that*

$$\Theta \leq \Theta'$$
$$s \longrightarrow^* v$$
$$\mathcal{D}' \models \Theta' \vdash_n [t_1/x]t' : (F, M, \rho) \triangleright c \Rightarrow v$$
$$\texttt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta'$$
$$@_n(\mathcal{D}') = 0$$

Proof. Since the last rule of $\mathcal{D}$ is (PTr-App), we have:

$$\mathcal{D}_0 \models \Theta_0 \vdash_n \lambda x.t' : (F_0, M_0, \{\gamma_1, \ldots, \gamma_k\} \to \rho) \triangleright c_0 \Rightarrow s_0 \tag{13}$$
$$\gamma_i = (F_i, M_i, \rho_i) \text{ for each } i \in \{1, \ldots, k\} \tag{14}$$
$$\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\} \tag{15}$$
$$M = M_0 \uplus (\biguplus_{i \in \{1,\ldots,k\}} M_i) \tag{16}$$
$$\mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\texttt{mul}} \{(\emptyset, c_i) \mid i \in \{1, \ldots, k\}\}, M) = (F, c) \tag{17}$$
$$\gamma_1 < \cdots < \gamma_k \tag{18}$$
$$\Theta = \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \tag{19}$$
$$s = s_0\, s_1\, \cdots\, s_k \tag{20}$$

We can assume that (13) is not derived by (PTr-Mark), since $\texttt{eorder}(\lambda x.t') = n$. Hence (13) is derived by (PTr-Abs) possibly with (PTr-Weak); thus we have

$$\overline{\mathcal{D}}_0 \models \Theta_0', x : \{\gamma_1, \ldots, \gamma_k\} \vdash_n t' : (F_0, M_0', \rho) \triangleright c_0 \Rightarrow s' \tag{21}$$
$$\Theta_0 \leq \Theta_0' \tag{22}$$
$$M_0 = M_0' \setminus (\biguplus_{i \in \{1,\ldots,k\}} M_i) \tag{23}$$
$$s_0 = \lambda x_{\gamma_1}. \cdots \lambda x_{\gamma_k}.s'. \tag{24}$$

By applying Lemma 25 to (21), we have

$$\biguplus_{i \in \{1,\ldots,k\}} M_i = \mathit{Markers}(\{\gamma_1, \ldots, \gamma_k\}) \subseteq \mathit{Markers}(\Theta_0', x : \{\gamma_1, \ldots, \gamma_k\}) \subseteq M_0'.$$

By this and (23), we have $M_0' = M_0 \uplus (\biguplus_{i \in \{1,\ldots,k\}} M_i)$. Then, by (16) we have $M = M_0'$. By this and (21), we have $F_0 \cap M = F_0 \cap M_0' = \emptyset$. Hence, by applying Lemma 26 to (17), we have

$$F = F_0 \qquad c = c_0 + \textstyle\sum_{i \in \{1,\ldots,k\}} c_i. \tag{25}$$

Thus, (21) is equal to:

$$\overline{\mathcal{D}}_0 \models \Theta'_0,\, x : \{\gamma_1, \dots, \gamma_k\} \vdash_n t' : (F, M, \rho) \triangleright c_0 \Rightarrow s'. \tag{26}$$

By (19) and (22), $\Theta'_0 + (\sum_{i\in\{1,\dots,k\}} \Theta_i)$ is well-defined. Hence, by applying Lemma 45 to (15) and (26), we have

$$\begin{aligned}&\mathcal{D}' \models \Theta'_0 + (\textstyle\sum_{i\in\{1,\dots,k\}} \Theta_i) \vdash_n [t_1/x]t' :\\ &\qquad (F, M, \rho) \triangleright c_0 + \textstyle\sum_{i\in\{1,\dots,k\}} c_i \Rightarrow [s_i/x_{\gamma_i}]_{i\in\{1,\dots,k\}} s'\end{aligned}$$

and $@_n(\mathcal{D}') = 0$. Now we define $\Theta' := \Theta'_0 + (\sum_{i\in\{1,\dots,k\}} \Theta_i)$ and $v := [s_i/x_{\gamma_i}]_{i\in\{1,\dots,k\}} s'$; then $\mathcal{D}' \models \Theta' \vdash_n u : (F, M, \rho) \triangleright c \Rightarrow v$ by (25). By (19) and (22), we have $\Theta \le \Theta'$, which implies $\mathtt{order}(\gamma') < n$ for any $(x' : \gamma') \in \Theta'$. Also we have $s = (\lambda x_{\gamma_1}.\cdots\lambda x_{\gamma_k}.s')\, s_1 \cdots s_k \longrightarrow^* [s_i/x_{\gamma_i}]_{i\in\{1,\dots,k\}} s'$ by (20) and (24). $\square$

For $\mathcal{D} \models \Theta \vdash_n t : \gamma \triangleright c \Rightarrow s$ such that $@_n(\mathcal{D}) > 0$ and $\mathtt{order}(\gamma') < n$ for any $x : \gamma' \in \Theta$, we define $\mathcal{D}$-*evaluation context* $E^{\mathcal{D}}$ and $\mathcal{D}$-*redex* $r^{\mathcal{D}}$ such that (i) $t = E^{\mathcal{D}}[r^{\mathcal{D}}]$ and (ii) either $r^{\mathcal{D}}$ is of the form $(\lambda x.t_0)t_1$ where $\mathtt{eorder}(\lambda x.t_0) = n$, or $r^{\mathcal{D}}$ is a non-terminal A. These are defined by induction on $\mathcal{D}$ and by case analysis on the rule used last:

- Cases of (PTr-Weak) and (PTr-Mark): Let $\mathcal{D}_0$ be the subderivation of $\mathcal{D}$. Then $E^{\mathcal{D}} := E^{\mathcal{D}_0}$ and $r^{\mathcal{D}} := r^{\mathcal{D}_0}$
- Case of (PTr-Var): This case does not happen by the assumption.
- Case of (PTr-Choice): Let $t = t_1 + t_2$, $\mathcal{D}_0$ be the subderivation of $\mathcal{D}$, and the root term of $\mathcal{D}_0$ be t_i. Then $E^{\mathcal{D}} := E^{\mathcal{D}_0} + t_2$ (if $i = 1$), $t_1 + E^{\mathcal{D}_0}$ (if $i = 2$), and $r^{\mathcal{D}} := r^{\mathcal{D}_0}$.
- Case of (PTr-Abs): Let $t = \lambda x.t'$ and $\mathcal{D}_0$ be the subderivation of $\mathcal{D}$. Then $E^{\mathcal{D}} := \lambda x.E^{\mathcal{D}_0}$ and $r^{\mathcal{D}} := r^{\mathcal{D}_0}$.
- Case of (PTr-App): Suppose that $t = t_0\, t_1$ and we have the following as the premises and a side condition of the last rule:

  $$\begin{aligned}&\mathcal{D}_0 \models \Theta_0 \vdash_n t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c_0 \Rightarrow s_0\\ &\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F'_i, M'_i, \rho_i) \triangleright c_i \Rightarrow s_i \quad \text{for each } i \in \{1, \dots, k\}\\ &(F_1, M_1, \rho_1) < \cdots < (F_k, M_k, \rho_k)\end{aligned}$$

  When $@_n(\mathcal{D}_i) > 0$ for some $i = 1, \dots, k$, let i_0 be the largest such i. We define $E^{\mathcal{D}} := t_0\, E^{\mathcal{D}_{i_0}}$ and $r^{\mathcal{D}} := r^{\mathcal{D}_{i_0}}$.

  Otherwise, if $@_n(\mathcal{D}_0) > 0$, then we define $E^{\mathcal{D}} := E^{\mathcal{D}_0}\, t_1$ and $r^{\mathcal{D}} := r^{\mathcal{D}_0}$.

  Otherwise, we have $\mathtt{eorder}(t_0) = n$. Since $@_n(\mathcal{D}_0) = 0$, t_0 is not an application term. Also, t_0 is not a variable by the assumption. If t_0 is a non-terminal, we define $E^{\mathcal{D}} := [\,]\, t_1$ and $r^{\mathcal{D}} := t_0$. If t_0 is a λ-abstraction, then we define $E^{\mathcal{D}} := [\,]$ and $r^{\mathcal{D}} := t$.
- Case of (PTr-Const): Suppose that $t = a\, t_0 \cdots t_k$ and let $\mathcal{D}_i$ be the subderivation of $\mathcal{D}$ whose root term is t_i. Let i_0 be the largest i such that $@_n(\mathcal{D}_i) > 0$. Then we define $E^{\mathcal{D}} := a\, t_0 \cdots t_{i_0-1} E^{\mathcal{D}_{i_0}} t_{i_0+1} \cdots t_k$ and $r^{\mathcal{D}} := r^{\mathcal{D}_{i_0}}$.
- Case of (PTr-NT): We define $E^{\mathcal{D}} := [\,]$ and $r^{\mathcal{D}} := t$.

Then, we define $\mathcal{D}$-*reduction*, written by $\longrightarrow_{\mathcal{D}}$, as follows:

$$\begin{aligned}E^{\mathcal{D}}[(\lambda x.t_0)\, t_1] &\longrightarrow_{\mathcal{D}} [t_0/x]t_1 && (\text{if } r^{\mathcal{D}} = (\lambda x.t_0)\, t_1)\\ E^{\mathcal{D}}[A] &\longrightarrow_{\mathcal{D}} t' && (\text{if } r^{\mathcal{D}} = A,\ (A = t') \in \mathcal{G}).\end{aligned}$$

We write $\le_{\mathrm{lg}}$ for the lexicographic order on pairs of natural numbers: $(n, n') \le_{\mathrm{lg}} (m, m')$ iff $n < m$ or $n = m$ and $n' \le m'$, and write $<_{\mathrm{lg}}$ for its strict order.

▶ **Lemma 47** (subject reduction). *Suppose*

$$n > 0$$
$$t \longrightarrow_{\mathcal{D}''} u \quad \textit{for some } \mathcal{D}''$$
$$\mathcal{D} \models \Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$$
$$\texttt{order}(\gamma) < n \textit{ for any } (x : \gamma) \in \Theta.$$

Then there exist Θ', v, *and* $\mathcal{D}'$ *such that*

$$\Theta \leq \Theta'$$
$$s \longrightarrow^* v$$
$$\mathcal{D}' \models \Theta' \vdash_n u : (F, M, \rho) \triangleright c \Rightarrow v$$
$$\texttt{order}(\gamma') < n \textit{ for any } (x' : \gamma') \in \Theta'$$
$$(@_n(\mathcal{D}'), \texttt{NT}(\mathcal{D}')) \leq_{\mathrm{lg}} (@_n(\mathcal{D}), \texttt{NT}(\mathcal{D})).$$

Proof. The proof proceeds by induction on $\mathcal{D}$ with case analysis on the last rule used. Since the other cases are straightforward, we discuss only the cases for (PTr-Choice) and (PTr-App).

- Case of (PTr-Choice): We have:

$$t = t_1 + t_2 \qquad s = s_i \qquad i = 1 \text{ or } 2.$$
$$\mathcal{D}_i \models \Theta \vdash_n t_i : (F, M, \rho) \triangleright c \Rightarrow s_i.$$

By symmetry, we assume that $t \longrightarrow_{\mathcal{D}''} u$ reduces t_1 side and let $\mathcal{D}''_1$ be the subderivation of $\mathcal{D}''$. Thus we have u_1 such that $t = t_1 + t_2 \longrightarrow_{\mathcal{D}''} u_1 + t_2 = u$.

When $i = 1$, by induction hypothesis for $t_1 \longrightarrow_{\mathcal{D}''_1} u_1$, there exist Θ', v, and $\mathcal{D}'_1$ such that

$$\Theta \leq \Theta'$$
$$(s =)\ s_1 \longrightarrow^* v$$
$$\mathcal{D}'_1 \models \Theta' \vdash_n u_1 : (F, M, \rho) \triangleright c \Rightarrow v$$
$$\texttt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta'$$
$$(@_n(\mathcal{D}'_1), \texttt{NT}(\mathcal{D}'_1)) \leq_{\mathrm{lg}} (@_n(\mathcal{D}_1), \texttt{NT}(\mathcal{D}_1)).$$

By (PTr-Choice) we have

$$\mathcal{D}' := \frac{\mathcal{D}'_1}{\Theta' \vdash_n u_1 + t_2 : (F, M, \rho) \triangleright c \Rightarrow v}$$

and $(@_n(\mathcal{D}'), \texttt{NT}(\mathcal{D}')) \leq_{\mathrm{lg}} (@_n(\mathcal{D}), \texttt{NT}(\mathcal{D}))$.

When $i = 2$, By (PTr-Choice) we have

$$\mathcal{D}' := \frac{\mathcal{D}_2}{\Theta \vdash_n u_1 + t_2 : (F, M, \rho) \triangleright c \Rightarrow s_2}$$

and $(@_n(\mathcal{D}'), \texttt{NT}(\mathcal{D}')) = (@_n(\mathcal{D}), \texttt{NT}(\mathcal{D}))$. Also we have $s = s_2 \longrightarrow^0 s_2$. The conditions for $\Theta' := \Theta$ are trivial.

- Case of (PTR-APP): Let $t = t_0 t_1$, and $\mathcal{D}''_i$ $(i = 0, 1, \ldots, k'')$ be the subderivations of $\mathcal{D}''$ determined by all the premises of the last (PTR-APP) in $\mathcal{D}''$, where the root of $\mathcal{D}''_0$ is t_0.

  If t_i $(i = 0$ or $1)$ is reduced in the reduction $t = t_0\, t_1 \longrightarrow_{\mathcal{D}''} u$ (i.e., if $@_n(\mathcal{D}''_i) > 0$ for some $i = 0, \ldots, k''$ or t_0 is a non-terminal), then the result follows immediately from the induction hypothesis for the subderivations of $\mathcal{D}$ whose root is t_i.

  Otherwise, we have $t_0 = \lambda x.t'$ and $u = [t_1/x]t'$ with $\mathtt{eorder}(t_0) = n$. Then the result follows from Lemma 46.

◻

▶ **Lemma 48** (progress). *Suppose*

$$n > 0$$
$$\mathcal{D} \models \Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$$
$$\mathtt{order}(\gamma) < n \text{ for any } (x : \gamma) \in \Theta$$
$$(@_n(\mathcal{D}), \mathtt{NT}(\mathcal{D})) \geq_{\mathrm{lg}} (1, 0).$$

Then there exist Θ'*,* u*,* v*, and* $\mathcal{D}'$ *such that*

$$\Theta \leq \Theta'$$
$$t \longrightarrow_{\mathcal{D}} u \qquad s \longrightarrow^* v$$
$$\mathcal{D}' \models \Theta' \vdash_n u : (F, M, \rho) \triangleright c \Rightarrow v$$
$$\mathtt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta'$$
$$(@_n(\mathcal{D}'), \mathtt{NT}(\mathcal{D}')) <_{\mathrm{lg}} (@_n(\mathcal{D}), \mathtt{NT}(\mathcal{D})).$$

Proof. The proof proceeds by induction on $\mathcal{D}$ with case analysis on the last rule used for $\mathcal{D}$. Since the other cases are straightforward, we discuss only the case for (PTR-APP).

Let $t = t_0 t_1$, and now we have:

$$\mathcal{D}_0 \models \Theta_0 \vdash_n t_0 : (F_0, M_0, \{\gamma_1, \ldots, \gamma_k\} \to \rho) \triangleright c_0 \Rightarrow s_0 \tag{27}$$
$$\gamma_i = (F_i, M_i, \rho_i) \text{ for each } i \in \{1, \ldots, k\} \tag{28}$$
$$\mathcal{D}_i \models \Theta_i \vdash_n t_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\} \tag{29}$$
$$M = M_0 \uplus (\biguplus_{i \in \{1,\ldots,k\}} M_i) \tag{30}$$
$$\mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\mathtt{mul}} \{(\emptyset, c_i) \mid i \in \{1, \ldots, k\}\}, M) = (F, c) \tag{31}$$
$$\gamma_1 < \cdots < \gamma_k \tag{32}$$
$$\Theta = \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta_i) \tag{33}$$
$$s = s_0\, s_1\, \cdots\, s_k \tag{34}$$

We further perform case analysis on the subderivations $\mathcal{D}_i$:

- Case where $@_n(\mathcal{D}_i) > 0$ for some $i = 1, \ldots, k$: Let i_0 be the largest such i. By induction hypothesis for (29) where $i = i_0$, there exist Θ'_{i_0}, u_1, v_{i_0}, and $\mathcal{D}'_{i_0}$ such that

$$\Theta_{i_0} \leq \Theta'_{i_0}$$
$$t_1 \longrightarrow_{\mathcal{D}_{i_0}} u_1 \qquad s_{i_0} \longrightarrow^* v_{i_0}$$
$$\mathcal{D}'_{i_0} \models \Theta'_{i_0} \vdash_n u_1 : (F_{i_0}, M_{i_0}, \rho_{i_0}) \triangleright c_{i_0} \Rightarrow v_{i_0}$$
$$\mathtt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta'_{i_0}$$
$$(@_n(\mathcal{D}'_{i_0}), \mathtt{NT}(\mathcal{D}'_{i_0})) <_{\mathrm{lg}} (@_n(\mathcal{D}_{i_0}), \mathtt{NT}(\mathcal{D}_{i_0})).$$

For any $i \neq i_0$, by Lemma 47 applied to (29), we have Θ'_i, v_i, and $\mathcal{D}'_i$ such that

$$\Theta_i \leq \Theta'_i$$
$$s_i \longrightarrow^* v_i$$
$$\mathcal{D}'_i \models \Theta'_i \vdash_n u_1 : (F_i, M_i, \rho_i) \triangleright c_i \Rightarrow v_i$$
$$\mathtt{order}(\gamma') < n \text{ for any } (x' : \gamma') \in \Theta'_i$$
$$(@_n(\mathcal{D}'_i), \mathtt{NT}(\mathcal{D}'_i)) \leq_{\mathrm{lg}} (@_n(\mathcal{D}_i), \mathtt{NT}(\mathcal{D}_i)).$$

Then we have a derivation:

$$\mathcal{D}' \models \Theta_0 + (\textstyle\sum_{i \in \{1,\ldots,k\}} \Theta'_i) \vdash_n t_0\, u_1 : (F, M, \rho) \triangleright c \Rightarrow s_0\, v_1 \,\cdots\, v_k.$$

For $\Theta' := \Theta_0 + (\sum_{i \in \{1,\ldots,k\}} \Theta'_i)$, it is clear that $\Theta \leq \Theta'$ and $\mathtt{order}(\gamma') < n$ for any $(x' : \gamma') \in \Theta'$. Also, $\mathcal{D}'$ satisfies the required condition: if $@_n(\mathcal{D}'_i) < @_n(\mathcal{D}_i)$ for some $i = 1, \ldots, k$, then $@_n(\mathcal{D}') < @_n(\mathcal{D})$, and if $@_n(\mathcal{D}'_i) = @_n(\mathcal{D}_i)$ for all $i = 1, \ldots, k$, then $@_n(\mathcal{D}') = @_n(\mathcal{D})$ and $\mathtt{NT}(\mathcal{D}') < \mathtt{NT}(\mathcal{D})$. Finally, we have $t = t_0\, t_1 \longrightarrow_{\mathcal{D}} t_0\, u_1$ (by the definition of $E^{\mathcal{D}}$) and $s = s_0\, s_1 \,\cdots\, s_k \longrightarrow^* s_0\, v_1 \,\cdots\, v_k$.

- Case where $@_n(\mathcal{D}_i) = 0$ for any $i = 1, \ldots, k$ and $@_n(\mathcal{D}_0) > 0$: Similar to (and easier than) the previous case.
- Case where $@_n(\mathcal{D}_i) = 0$ for any $i = 0, \ldots, k$: Now t_0 is a non-terminal or λ-abstraction. The case of non-terminal is straightforward; we consider the case of λ-abstraction. Let $t_0 = \lambda x.t'$. Since $@_n(\mathcal{D}) \geq 1$ and $@_n(\mathcal{D}_i) = 0$ for any $i = 0, \ldots, k$, we have $\mathtt{eorder}(\lambda x.t') = n$. Then the result follows from Lemma 46, with $u := [t_1/x]t'$.

◻

▶ **Lemma 49.** *If* $Comp_n(\{(F_1, c_1), \ldots, (F_k, c_k)\}, M) = (F, c)$ *and* $Comp_{n-1}(\{(F_1{\restriction_{<n-1}}, c'_1), \ldots, (F_k{\restriction_{<n-1}}, c'_k)\}, M{\restriction_{<n-1}}) = (F', c')$, *then* $F{\restriction_{<n-1}} = F'$.

Proof. Trivial by the definition of $Comp_n$. ◻

▶ **Lemma 50** (decrease of the order). *Suppose (i)* $n > 1$ *(ii)* $\mathcal{D} \models \Theta \vdash_n t : (F, M, \rho) \triangleright c \Rightarrow s$ *with* $@_n(\mathcal{D}) = 0$, *(iii)* $\mathtt{eorder}(t) < n$ *(iv)* Θ *contains neither flag* $n-1$ *nor marker* $n-1$. *Then there exist* $\mathcal{D}'$ *and* c' *such that* $\mathcal{D}' \models \Theta \vdash_{n-1} t : (F{\restriction_{<n-1}}, M{\restriction_{<n-1}}, \rho) \triangleright c' \Rightarrow s$ *with* $c' \geq c + |F \cap \{n-1\}|$. *Furthermore, the two derivation trees have the same structure (except the labels), and for each node of* $\mathcal{D}$ *labeled by* $\Theta_1 \vdash_n t_1 : (F_1, M_1, \rho_1) \triangleright c_1 \Rightarrow s_1$ *with* $c_1 > 0$, *the corresponding node of* $\mathcal{D}'$ *is labeled by* $\Theta_1 \vdash_n t_1 : (F_1{\restriction_{<n-1}}, M_1{\restriction_{<n-1}}, \rho_1) \triangleright c'_1 \Rightarrow s_1$ *with* $c'_1 > 0$.

Proof. The proof proceeds by induction on $\mathcal{D}$, with case analysis on the last rule used.

- Case (PTr-Abs): In this case, we have:

$$\Theta, x : \xi \vdash_n t_0 : (F, M_0, \rho_0) \triangleright c \Rightarrow s_0$$
$$t = \lambda x.t_0 \qquad M = M_0 \setminus Markers(\xi) \qquad \rho = \xi \to \rho_0 \qquad s = \lambda x_{\rho_1}.\cdots\lambda x_{\rho_k}.s$$

Since $\mathtt{eorder}(t_0) < n$, ξ does not contain $n-1$ as a flag or a marker. Therefore, by the induction hypothesis, we have a derivation for

$$\Theta, x : \xi \vdash_{n-1} t_0 : (F{\restriction_{<n-1}}, M_0{\restriction_{<n-1}}, \rho_0) \triangleright c' \Rightarrow s_0$$

for some c' such that $c' \geq c + |F \cap \{n-1\}|$. By using (PTR-ABS), we obtain a derivation for

$$\Theta \vdash_{n-1} t_0 : (F{\upharpoonright}_{<n-1}, M{\upharpoonright}_{<n-1}, \rho) \triangleright c' \Rightarrow s$$

as required.

- Case (PTR-APP): In this case, we have:

$$\begin{array}{l} t = t_0 t_1 \qquad \texttt{eorder}(t_0) = \ell < n \\ \Theta_0 \vdash_n t_0 : (F_0, M_0, \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c_0 \Rightarrow s_0 \\ \Theta_i \vdash_n t_1 : (F'_i, M'_i, \rho_i) \triangleright c_i \Rightarrow s_i \qquad F'_i{\upharpoonright}_{<\ell} = F_i \qquad M'_i{\upharpoonright}_{<\ell} = M_i \text{ for each } i \in \{1, \dots, k\} \\ M = M_0 \uplus (\biguplus_{i \in \{1,\dots,k\}} M'_i) \\ \mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\texttt{mul}} \{(F'_i{\upharpoonright}_{\geq \ell}, c_i) \mid i \in \{1, \dots, k\}\}, M) = (F, c) \\ (F_1, M_1, \rho_1) < \dots < (F_k, M_k, \rho_k) \\ \Theta = \Theta_0 + (\sum_{i \in \{1,\dots,k\}} \Theta_i) \\ s = s_0\, s_1 \, \cdots \, s_k \end{array}$$

By the induction hypothesis, we have

$$\begin{array}{l} \Theta_0 \vdash_{n-1} t_0 : (F_0{\upharpoonright}_{<n-1}, M_0{\upharpoonright}_{<n-1}, \{(F_1, M_1, \rho_1), \dots, (F_k, M_k, \rho_k)\} \to \rho) \triangleright c'_0 \Rightarrow s_0 \\ \Theta_i \vdash_{n-1} t_1 : (F'_i{\upharpoonright}_{<n-1}, M'_i{\upharpoonright}_{<n-1}, \rho_i) \triangleright c'_i \Rightarrow s_i \\ c'_0 \geq c_0 + |F_0 \cap \{n-1\}| \qquad c'_i \geq c_i + |F'_i \cap \{n-1\}| \end{array}$$

Let (F', c') be:

$$\mathit{Comp}_{n-1}(\{(F_0{\upharpoonright}_{<n-1}, c'_0)\} \cup_{\texttt{mul}} \{((F'_i{\upharpoonright}_{<n-1}){\upharpoonright}_{\geq \ell}, c'_i) \mid i \in \{1, \dots, k\}\}, M{\upharpoonright}_{n-1}).$$

By using (PTR-APP), we obtain a derivation tree for

$$\Theta \vdash_{n-1} t : (F', M{\upharpoonright}_{<n-1}, \rho) \triangleright c' \Rightarrow s.$$

By Lemma 49, we have $F' = F{\upharpoonright}_{<n-1}$. Thus, it remains to show $c' \geq c + |F \cap \{n-1\}|$, as the derivation satisfies the other conditions. We consider f_ℓ and f'_ℓ in the computation of $\mathit{Comp}_n(\{(F_0, c_0)\} \cup_{\texttt{mul}} \{(F'_i{\upharpoonright}_{\geq \ell}, c_i) \mid i \in \{1, \dots, k\}\}, M)$. If $n-1 \in M$, then $n-1 \notin F$. We have $f'_n = f_{n-1} = f'_{n-1} + |F_0 \cap \{n-1\}| + \sum_i |F'_i \cap \{n-1\}|$, and hence:

$$\begin{array}{l} c = f'_n + c_0 + \sum_i c_i = f'_{n-1} + (c_0 + |F_0 \cap \{n-1\}|) + \sum_i (c_i + |F'_i \cap \{n-1\}|) \\ \leq f'_{n-1} + c'_0 + \sum_i c'_i = c' \end{array}$$

as required. If $n-1 \notin M$, then we have $c = 0$ by Lemma 24. If $n-1 \notin F$, we obtain $c + |F \cap \{n-1\}| = 0 \leq c'$ immediately. Otherwise, $f_{n-1} = f'_{n-1} + |F_0 \cap \{n-1\}| + \sum_i |F'_i \cap \{n-1\}| > 0$. Thus, we have

$$\begin{array}{l} c + |F \cap \{n-1\}| = 1 \leq f'_{n-1} + |F_0 \cap \{n-1\}| + \sum_i |F'_i \cap \{n-1\}| \\ \leq f'_{n-1} + c'_0 + \sum_i c'_i \leq c' \end{array}$$

as required.

- Case (PTR-CONST): In this case, we have:

$$\begin{array}{l} t = a\, t_1 \, \cdots \, t_k \qquad s = a\, s_1 \, \cdots \, s_k \\ \Theta_i \vdash_n t_i : (F_i, M_i, \mathtt{o}) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \dots, k\} \\ M = M_1 \uplus \cdots \uplus M_k \\ \mathit{Comp}_n(\{(\{0\}, 0), (F_1, c_1), \dots, (F_k, c_k)\}, M) = (F, c) \\ \Theta = \Theta_1 + \cdots + \Theta_k \end{array}$$

By the induction hypothesis, we have:

$$\Theta_i \vdash_{n-1} t_i : (F_i{\restriction_{<n-1}}, M_i{\restriction_{<n-1}}, \mathsf{o}) \triangleright c'_i \Rightarrow s_i \text{ for each } i \in \{1,\dots,k\}$$
$$c'_i \geq c_i + |F_i \cap \{n-1\}|$$

Let $\mathit{Comp}_{n-1}(\{(\{0\},0),(F_1{\restriction_{<n-1}}, c'_1),\dots,(F_k{\restriction_{<n-1}}, c'_k)\}, M{\restriction_{<n-1}}) = (F', c')$. By Lemma 49, $F' = F{\restriction_{<n-1}}$. Thus, by using (PTr-Const), we obtain a derivation for

$$\Theta \vdash_{n-1} t : (F{\restriction_{<n-1}}, M{\restriction_{<n-1}}, \mathsf{o}) \triangleright c' \Rightarrow s.$$

It remains to show $c' \geq c + |F \cap \{n-1\}|$, as the derivation satisfies the other conditions. We consider f_ℓ and f'_ℓ in the computation of $\mathit{Comp}_n(\{(\{0\},0),(F_1,c_1),\dots,(F_k,c_k)\}, M) = (F,c)$. If $n-1 \in M$, then $n-1 \notin F$. Thus, we have:

$$c + |F \cap \{n-1\}| = c = f'_n + c_1 + \cdots + c_k = f_{n-1} + c_1 + \cdots + c_k$$
$$= (f'_{n-1} + |F_1 \cap \{n-1\}| + \cdots + |F_k \cap \{n-1\}|) + c_1 + \cdots + c_k$$
$$= f'_{n-1} + (c_1 + |F_1 \cap \{n-1\}|) + \cdots + (c_k + |F_k \cap \{n-1\}|) \leq f'_{n-1} + c'_1 + \cdots + c'_k = c'$$

If $n-1 \notin M$, then by Lemma 24, it must be the case that $c = 0$. If $n-1 \notin F$, then we obtain $c + |F \cap \{n-1\}| = 0 \leq c'$ immediately. Otherwise, $f_{n-1} = f'_{n-1} + |F_1 \cap \{n-1\}| + \cdots + |F_k \cap \{n-1\}| > 0$. Thus, we have:

$$c + |F \cap \{n-1\}| = 1 \leq f'_{n-1} + |F_1 \cap \{n-1\}| + \cdots + |F_k \cap \{n-1\}|$$
$$\leq f'_{n-1} + c'_1 + \cdots + c'_k \leq c'$$

as required.

- Case (PTr-NT): In this case, the result follows immediately from the induction hypothesis.

□

The following two lemmas are corollaries of Lemmas 48 and 50 respectively.

▶ **Lemma 51.** *If $n > 0$, $\mathcal{D} \models \emptyset \vdash_n t : (\emptyset, \{0,\dots,n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$, and $@_n(\mathcal{D}) > 0$, then there exist u, v, and $\mathcal{D}'$ such that $t \longrightarrow_n^* u$, $s \longrightarrow^* v$, $\mathcal{D}' \models \emptyset \vdash_n u : (\emptyset, \{0,\dots,n-1\}, \mathsf{o}) \triangleright c \Rightarrow v$, and $@_n(\mathcal{D}') < @_n(\mathcal{D})$.*

▶ **Lemma 52.** *If $n > 1$, $\mathcal{D} \models \emptyset \vdash_n t : (\emptyset, \{0,\dots,n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$, and $@_n(\mathcal{D}) = 0$, then $\emptyset \vdash_{n-1} t : (\emptyset, \{0,\dots,n-2\}, \mathsf{o}) \triangleright c' \Rightarrow s$ is derived for some $c' \geq c$.*

▶ **Lemma 53.** *If $\mathcal{D} \models \emptyset \vdash_1 t : (\emptyset, \{0\}, \mathsf{o}) \triangleright c \Rightarrow s$ and $@_1(\mathcal{D}) = 0$, then $\mathcal{T}(s) \in \mathcal{L}(t)$, with $c \leq |\mathcal{T}(s)|$.*

Proof. The proof proceeds by induction on $\mathcal{D}$ and case analysis on the rule used last for $\mathcal{D}$.

Below we use the fact that if $\emptyset \vdash_1 t : (F, \emptyset, \mathsf{o}) \triangleright c \Rightarrow s$ is derived without (PTr-App) then $c = 0$ and $\mathcal{T}(s) \in \mathcal{L}(t)$, where $c = 0$ follows from Lemma 24 and $\mathcal{T}(s) \in \mathcal{L}(t)$ can be shown by straightforward induction on the derivation.

- Case of (PTr-Weak): Trivial.
- Case of (PTr-Mark): We have

$$\frac{\emptyset \vdash_1 t : (F', \{0\} \setminus M, \mathsf{o}) \triangleright c' \Rightarrow s \qquad M \subseteq \{0\} \qquad \mathit{Comp}_1(\{(F', c')\}, \{0\}) = (\emptyset, c)}{\emptyset \vdash_1 t : (\emptyset, \{0\}, \mathsf{o}) \triangleright c \Rightarrow s}$$

If $M = \emptyset$, then $F' = \emptyset$ and $c' = c$. Hence the required properties follow from the induction hypothesis.

If $M \neq \emptyset$, then we have $\emptyset \vdash_1 t : (F', \emptyset, \mathtt{o}) \triangleright c' \Rightarrow s$. By the above fact, we have $\mathcal{T}(s) \in \mathcal{L}(t)$ and $c' = 0$. Since $c \leq 1$ by $\mathit{Comp}_1(\{(F', c')\}, \{0\}) = (\emptyset, c)$, we have $c \leq |\mathcal{T}(s)|$.

- Case of (PTr-Var): This case does not happen since the environment is empty.
- Case of (PTr-Choice): Clear by induction hypothesis.
- Case of (PTr-Abs): This case does not happen since the simple type of t is $\mathtt{o}$.
- Case of (PTr-App): This case does not happen by the assumption.
- Case of (PTr-Const): We have:

$$\frac{\begin{array}{c}\mathtt{ar}(a) = k \\ \emptyset \vdash_1 t_i : (F_i, M_i, \mathtt{o}) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \dots, k\} \\ \{0\} = M_1 \uplus \cdots \uplus M_k \\ \mathit{Comp}_n(\{(\{0\}, 0), (F_1, c_1), \dots, (F_k, c_k)\}, \{0\}) = (\emptyset, c)\end{array}}{\emptyset \vdash_1 a\, t_1 \,\cdots\, t_k : (\emptyset, \{0\}, \mathtt{o}) \triangleright c \Rightarrow a\, s_1 \,\cdots\, s_k}$$

Now there exists i_0 such that $M_{i_0} = \{0\}$ and $M_i = \emptyset$ for $i \neq i_0$. By the above fact, for $i \neq i_0$, $c_i = 0$ and $\mathcal{T}(s_i) \in \mathcal{L}(t_i)$. Since $F_{i_0} = \emptyset$, by induction hypothesis, $c_{i_0} \leq |\mathcal{T}(s_{i_0})|$ and $\mathcal{T}(s_{i_0}) \in \mathcal{L}(t_{i_0})$. By $\mathit{Comp}_n(\{(\{0\}, 0), (F_1, c_1), \dots, (F_k, c_k)\}, \{0\}) = (\emptyset, c)$, we have

$$c = 1 + \big|\{i \leq k | 0 \in F_i\}\big| + c_1 + \cdots + c_k \leq 1 + |\mathcal{T}(s_1)| + \cdots + |\mathcal{T}(s_k)| = |\mathcal{T}(a\, s_1 \cdots s_k)|$$

and also we have $\mathcal{T}(a\, s_1 \,\cdots\, s_k) \in \mathcal{L}(a\, t_1 \,\cdots\, t_k)$, as required.

- Case of (PTr-NT): Clear by induction hypothesis.

□

Proof of Lemma 20. If $n > 0$, since $\emptyset \vdash_n S : (\emptyset, \{0, \dots, n-1\}, \mathtt{o}) \triangleright c \Rightarrow s$, by using Lemma 51 repeatedly, we have

$$S \longrightarrow_n^* t_{n-1} \qquad s \longrightarrow^* s_{n-1}$$
$$\mathcal{D}_n \models \emptyset \vdash_n t_{n-1} : (\emptyset, \{0, \dots, n-1\}, \mathtt{o}) \triangleright c \Rightarrow s_{n-1}$$
$$@_n(\mathcal{D}_n) = 0.$$

If $n > 1$, by Lemma 52 we have $c_{n-1} \geq c$ and

$$\emptyset \vdash_{n-1} t_{n-1} : (\emptyset, \{0, \dots, n-2\}, \mathtt{o}) \triangleright c_{n-1} \Rightarrow s_{n-1}.$$

By repeating this procedure, we obtain

$$S \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0$$
$$s \longrightarrow^* s_{n-1} \longrightarrow^* \cdots \longrightarrow^* s_0$$
$$c_1 \geq \cdots \geq c_{n-1} \geq c$$
$$\mathcal{D}_1 \models \emptyset \vdash_1 t_0 : (\emptyset, \{0\}, \mathtt{o}) \triangleright c_1 \Rightarrow s_0$$
$$@_1(\mathcal{D}_1) = 0.$$

By Lemma 53, we have

$$\mathcal{T}(s_0) \in \mathcal{L}(t_0) \qquad c_1 \leq |\mathcal{T}(s_0)|$$

and hence

$$\mathcal{T}(s) = \mathcal{T}(s_0) \in \mathcal{L}(t_0) \subseteq \mathcal{L}(S) = \mathcal{L}(\mathcal{G}) \qquad c \leq c_1 \leq |\mathcal{T}(s_0)| = |\mathcal{T}(s)|$$

as required. □

A.5 Proof of Lemma 12

We now discuss how to modify the argument in the previous subsections to obtain a triple (G, H, u) that satisfies the requirement of Lemma 12. Let $\mathcal{G}$ be an order-n, direction-annotated (i.e., each occurrence of a terminal symbol is annotated with a direction) tree grammar.

We first modify typing (or, type-based transformation) rules. In the type system of Section A.1 (and the original type system of Parys [23]), the size of a tree (= the number of order-0 flags) is estimated via the number of order-1 flags placed on the path from the root of the derivation to the (unique occurrence of) order-0 marker. Thus, for a direction-annotated grammar, by ensuring that the order-0 marker can be placed only at the node $a^{\langle 0\rangle}\, T_1\, \cdots\, T_k$ reached following the directions, we can estimate the length of the path following the directions, instead of the size of the whole tree.

Based on the intuition above, we modify the rules (PT-Mark) and (PT-Const) as follows.

$$\frac{\begin{array}{c}\Theta \vdash^{\mathtt{dir}}_n t : (F', M', \rho) \triangleright c' \Rightarrow s \\ M \subseteq \{j \mid \mathtt{eorder}(t) \leq j < n\} \qquad \mathit{Comp}_n(\{(F', c')\}, M \uplus M') = (F, c) \\ \text{if } 0 \in M, \text{ then } t \text{ is of the form } a^{\langle 0\rangle}\, t_1\, \cdots\, t_k\end{array}}{\Theta \vdash^{\mathtt{dir}}_n t : (F, M \uplus M', \rho) \triangleright c \Rightarrow s} \quad \text{(PDT-Mark)}$$

$$\frac{\begin{array}{c}\mathtt{ar}(a) = k \\ \Theta_i \vdash^{\mathtt{dir}}_n t_i : (F_i, M_i, \mathsf{o}) \triangleright c_i \Rightarrow s_i \text{ for each } i \in \{1, \ldots, k\} \\ M = M_1 \uplus \cdots \uplus M_k \qquad 0 \notin \bigcup_{j \in \{1,\ldots,k\}\setminus\{i\}} M_j \\ \mathit{Comp}_n(\{(\{0\}, 0), (F_1, c_1), \ldots, (F_k, c_k)\}, M) = (F, c)\end{array}}{\Theta_1 + \cdots + \Theta_k \vdash^{\mathtt{dir}}_n a^{\langle i\rangle}\, t_1\, \cdots\, t_k : (F, M, \mathsf{o}) \triangleright c \Rightarrow a\, s_1\, \cdots\, s_k} \quad \text{(PDT-Const)}$$

The other rules are unchanged (except $\vdash_n$ is replaced by $\vdash^{\mathtt{dir}}_n$); we write (PDT-X) for the rule obtained from (PD-X), for each X.

The followings are variations of Theorem 35 and Lemma 55.

▶ **Lemma 54.** *Let t be an order-n direction-annotated term. If $t \longrightarrow^* \pi$, then $\mathcal{D} \models \emptyset \vdash^{\mathtt{dir}}_n t : (\emptyset, \{0, \ldots, n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$ for some $\mathcal{D}$ and c such that $\mathbf{exp}_n(c) \geq |\pi|_{\mathrm{p}}$. Furthermore, $\mathcal{D}^!$ is an admissible derivation in the linear type system.*

▶ **Lemma 55.** *Let t be an order-n direction-annotated term. If $\emptyset \vdash_n t : (\emptyset, \{0, \ldots, n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$, then s is a $\lambda^{\rightarrow}$-term of order at most n, and $t \longrightarrow^* \mathcal{T}(s)$, with $c \leq |\mathcal{T}(s)|_{\mathrm{p}}$.*

To prove Lemma 12, we will use Lemma 54 to obtain a pumpable derivation for (the direction-annotated version of) $C[D^i[t]]$ where C, D are linear contexts. In the construction of the pumpable derivation, we need to require that a sub-derivation for a term of the form $D^j[t]$ occurs only once. To show that property, we extend the syntax of terms with labels as follows.

$$t ::= x \mid a\, t_1\, \cdots\, t_k \mid t_0 t_1 \mid \lambda x.t \mid t_0 + t_1 \mid t^{\Psi}.$$

Here, Ψ is a *multiset* of labels of the form $\{\psi_1, \ldots, \psi_k\}$. We often just write t^{ψ} for $t^{\{\psi\}}$. We have omitted non-terminals, since they will not be used below. We identify $(t^{\Psi_1})^{\Psi_2}$ with $t^{\Psi_1 \cup_{\mathtt{mul}} \Psi_2}$. We extend the reduction relation $\longrightarrow$ to that for labeled terms by:

$$C[(\lambda x.t_0)^{\Psi} t_1] \longrightarrow C[([t_1/x]t_0)^{\Psi}] \qquad C[(t_0 + t_1)^{\Psi}] \longrightarrow C[t_i^{\Psi}]\ (i = 0 \vee i = 1),$$

where C is an arbitrary context. Note that the (multi)set of labels for a function is transferred to the residual. For example, we have:

$$(\lambda x.x^{\psi_1})^{\psi_2}\mathtt{c}^{\psi_3} \longrightarrow ([\mathtt{c}^{\psi_3}/x]x^{\psi_1})^{\psi_2} = ((\mathtt{c}^{\psi_3})^{\psi_1})^{\psi_2} = \mathtt{c}^{\{\psi_1,\psi_2,\psi_3\}}.$$

The following is a labeled version of Lemma 21.

▶ **Lemma 56.** *Let t be a labeled, order-n term. If $t \longrightarrow^* \pi$, then*

$$t \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0 \longrightarrow_{\mathrm{c}}^* \pi.$$

Proof. The proof is essentially the same as that of Lemma 21. Note that the β-reduction relation satisfies the Church-Rosser property also for labeled terms (which can be proved by using the standard argument using parallel reduction). ☐

The transformation rules are extended for labeled terms accordingly. We write $\mathbf{Labs}(t)$ for the multiset of labels occurring in t. For a transformation derivation $\mathcal{D}$, we also $\mathbf{Labs}(\mathcal{D})$ for the multiset of labels attached to the term in the conclusion of a judgment occurring in the derivation, except the conclusion of (PDT-Mark) and (PDT-Weak), i.e.,

$$\mathbf{Labs}(\frac{\mathcal{D}}{\Theta \vdash_n^{\mathtt{dir}} t^{\Psi} : \gamma \triangleright c \Rightarrow s}) = \Psi \cup_{\mathtt{mul}} \mathbf{Labs}(\mathcal{D}_1) \cup_{\mathtt{mul}} \cdots \cup_{\mathtt{mul}} \mathbf{Labs}(\mathcal{D}_k)$$

if the last rule is neither (PDT-Mark) nor (PDT-Weak) and t is not of the form $t'^{\Psi'}$ (so that Ψ is the multiset of all the outermost labels), and

$$\mathbf{Labs}(\frac{\mathcal{D}}{\Theta \vdash_n^{\mathtt{dir}} t^{\Psi} : \gamma \triangleright c \Rightarrow s}) = \mathbf{Labs}(\mathcal{D})$$

if the last rule is (PDT-Mark) or (PDT-Weak).

We can strengthen Lemma 54 as follows.

▶ **Lemma 57.** *Let $t = C[t_0^{\psi}]$ be an order-n term, where ψ does not occur in C, t_0. If*

$$t \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0 \longrightarrow_{\mathrm{c}}^* \pi,$$

then there exists a derivation $\mathcal{D}$ for $\emptyset \vdash_n^{\mathtt{dir}} t : (\emptyset, \{0, \ldots, n-1\}, \mathtt{o}) \triangleright c \Rightarrow s$ such that $\mathbf{exp}_n(c) \geq |\pi|_{\mathrm{p}}$. Furthermore, if $\mathbf{Labs}(\pi)(\psi) \leq 1$, then $\mathcal{D}$ contains at most one judgment for t_0^{ψ}. Moreover, $\mathcal{D}^!$ is an admissible derivation in the linear type system.

Proof. The proof of the properties except the part "Furthermore, ..." is almost the same as the proof of Theorem 35. To confirm the part "Furthermore, ...", it suffices to observe the following facts: (i) for the derivation $\mathcal{D}_0$ for π constructed in the proof of (a direction-annoated version of) Lemma 36, $\mathbf{Labs}(\mathcal{D}_0) = \mathbf{Labs}(\pi)$, and (ii) during the construction of derivations (in a backward manner with respect to the reduction sequence) in the proofs of Lemmas 37 and 38, $\mathbf{Labs}(\mathcal{D})$ decreases monotonically. ☐

The following lemma ensures that, if C is a linear context, then there is a reduction sequence that satisfies the assumption of the lemma above.

▶ **Lemma 58.** *Suppose (i) a pair of contexts $[\,] : \kappa \vdash_{\mathtt{ST}} C : \mathtt{o}$ and $[\,] : \kappa \vdash_{\mathtt{ST}} D : \kappa$ is linear, (ii) $\vdash_{\mathtt{ST}} t : \kappa$, and (iii) C, D, t do not contain labels. Define $t^{(k)}$ by: $t^{(0)} = t^{\psi_0}$ and $t^{(i)} = D[t^{(i-1)}]^{\psi_i}$. Then there exists a reduction sequence*

$$C[t^{(k)}] \longrightarrow_n^* t_{n-1} \longrightarrow_{n-1}^* \cdots \longrightarrow_1^* t_0 \longrightarrow_{\mathrm{c}}^* \pi$$

such that $\mathbf{Labs}(\pi)(\psi_i) = 1$.

Proof. Let $C[t^{(k)}] \longrightarrow^* \pi$ be the call-by-name reduction sequence (note that, by the typing assumptions, the reduction sequence yields a tree). By the linearity condition, the reduction sequence must be of the form:

$$C[t^{(k)}] \longrightarrow^* E_k[t^{(k)}\widetilde{u}_k] \longrightarrow^* E_{k-1}[(t^{(k-1)}\widetilde{u}_{k-1}] \longrightarrow^* \cdots E_0[t_{(0)}\widetilde{u}_0] \longrightarrow^* \pi$$

where E_i is a call-by-value evaluation context and $E_i[t^{(i)}\widetilde{u}_i]$ contains exactly one occurrence of ψ_j for each $j \geq i$ (since ψ_j occurs only in a head position of $t^{(i)}\widetilde{u}_i$ and can never be duplicated afterwards). Thus, $\mathbf{Labs}(\pi)(\psi_i) = 1$ for each $i \in \{0, \dots, k\}$. By Lemma 56, we have the required result. ☐

Using the lemmas above, we can prove Lemma 12.

Proof of Lemma 12. Let C', D', t' be the direction-annotated contexts and term obtained from C, D, t respectively by (recursively) replacing each occurrence of $a\, t_1 \cdots t_k$ (where $k > 0$) with:

$$(a^{\langle 1\rangle}\, t_1 \cdots t_k) + \cdots + (a^{\langle k\rangle}\, t_1 \cdots t_k),$$

and replacing each occurrence of a nullary non-terminal a with $a^{\langle 0\rangle}$. By the construction, for any $\pi \in \mathcal{L}(C'[D'^i[t']]$, $\mathbf{rmdir}(\pi) = \mathcal{T}(C[D^i[t]])$. By the assumption that $\{\mathcal{T}(C[D^k[t]]) \mid k \geq 1\}$ is infinite, for any d, there exists k_d such that $|\mathcal{T}(C'[D'^{k_d}[t']])|_{\mathrm{p}} \geq d$. Let $\mathcal{G}_k$ be the grammar just consisting of $S = C'[D'^k[t']]$ that generates a single tree $\mathcal{T}(C'[D'^k[t']])$. By Lemma 57, for any c_0, $\mathcal{D} \models \emptyset \vdash^{\mathtt{dir}}_n C'[D'^{k_{\mathbf{exp}_n(c_0)}}[t']] : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c \Rightarrow s$ holds for some c such that $c \geq c_0$ and a subderivation for $D'^i[t']$ occurs at most once for each i. (To observe the last condition, let u (u', resp.) be the term obtained from $C[D^{k_{\mathbf{exp}_n(c_0)}}[t]]$ ($C'[D'^{k_{\mathbf{exp}_n(c_0)}}[t']]$, resp.) by adding a label ψ_i to $D^i[t]$ ($D'^i[t']$, resp.) for each i. By Lemma 58, we have $u \longrightarrow^*_n \longrightarrow^*_{n-1} \cdots \longrightarrow^*_1 \longrightarrow^*_{\mathrm{c}} \pi$ such that $\mathbf{Labs}(\pi)(\psi_i) = 1$, from which we obtain a reduction sequence $u' \longrightarrow^*_n \longrightarrow^*_{n-1} \cdots \longrightarrow^*_1 \longrightarrow^*_{\mathrm{c}} \pi'$ such that $\mathbf{Labs}(\pi')(\psi_i) = 1$, which satisfies the assumption for Lemma 57.) By choosing a sufficiently large number c_0, we can ensure that there exist q_1, p, q_2 such that $\emptyset \vdash^{\mathtt{dir}}_n C'[D'^{k_{\mathbf{exp}_n(c_0)}}[t']] : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c_1 + c_2 + c_3 \Rightarrow G[H[u]]$ (where $c = c_1 + c_2 + c_3$ and $G[H[u]] = s$) is derived from $\emptyset \vdash^{\mathtt{dir}}_n D'^{p+q_2}[t'] : \gamma \triangleright c_1 + c_2 \Rightarrow H[u]$, which is in turn derived from $\emptyset \vdash^{\mathtt{dir}}_n D'^{q_2}[t'] : \gamma \triangleright c_1 \Rightarrow u$, where $c_1, c_2 > 0$. Thus, by "pumping" the derivation of $\emptyset \vdash^{\mathtt{dir}}_n D'^{p+q_2}[t'] : \gamma \triangleright c_1 + c_2 \Rightarrow H[u]$ from $\emptyset \vdash^{\mathtt{dir}}_n D'^{q_2}[t'] : \gamma \triangleright c_1 \Rightarrow u$, we obtain $\emptyset \vdash^{\mathtt{dir}}_n C'[D'^{pk+q}[t]] : (\emptyset, \{0, \dots, n-1\}, \mathsf{o}) \triangleright c_1 + kc_2 + c_3 \Rightarrow G[H^k[u]]$, where $q = k_{\mathbf{exp}_n(c_0)} - p$. By Lemma 55, we have $\mathcal{T}(G[H^k[u]]) \in \mathcal{L}(\mathcal{G}_{pk+q})$. Because $\mathbf{rmdir}(\pi) = \mathcal{T}(C[D^{pk+q}[t]])$ for every $\pi \in \mathcal{L}(\mathcal{G}_{pk+q})$, we have $\mathbf{rmdir}(\mathcal{T}(G[H^k[u]])) = \mathcal{T}(C[D^{pk+q}[t]])$. By Lemma 55, we also have $|\mathcal{T}(G[H^k[u]])|_{\mathrm{p}} \geq c_1 + kc_2 + c_3$. Thus, we also have the second condition of the lemma. Finally, since $\mathcal{D}^!$ is an admissible derivation, in which a linear type is assinged to every term containing t', G and H are linear by Lemma 32. ☐

A.6 Examples

A.6.1 Example 1

Consider the second-order tree grammar $\mathcal{G}_2$ consisting of the following rules.

$$\begin{array}{ll} S \to R\, A & \\ R\, f \to f\, \mathtt{e} & R\, f \to R\,(T\, f) \\ A\, x \to \mathtt{a}\, x\, x & \\ T\, f\, x \to f(f\, x). & \end{array}$$

The language generated by $\mathcal{G}_2$ is:

$$\{\pi_{2^n} \mid n \geq 0\},$$

where $\pi_0 = \mathtt{e}$ and $\pi_{k+1} = \mathtt{a}\,\pi_k\,\pi_k$. The rules can also be expressed as the set of equations:

$$S = R\,A \qquad R = \lambda f.(f\,\mathtt{e} + R\,(T\,f)) \qquad A = \lambda x.\mathtt{a}\,x\,x \qquad T = \lambda f.\lambda x.f(f\,x).$$

A good way to construct a type derivation in Parys' type system is to first reduce a term to a tree, and then construct type derivations for the terms occurring during the reduction in a backward manner. Consider the following rewriting sequence:

$$\begin{aligned}
&S \longrightarrow R\,A \longrightarrow (\lambda f.(f\,\mathtt{e} + R(T\,f)))A \longrightarrow A\,\mathtt{e} + R(T\,A) \longrightarrow R(T\,A)\\
&\longrightarrow (\lambda f.(f\,\mathtt{e} + R(T\,f)))(T\,A) \longrightarrow (T\,A\,\mathtt{e} + R(T(T\,A))) \longrightarrow T\,A\,\mathtt{e}\\
&\longrightarrow (\lambda f.\lambda x.f(f\,x))\,A\,\mathtt{e} \longrightarrow (\lambda x.A(A\,x))\mathtt{e}\\
&\longrightarrow A(A\,\mathtt{e}) \longrightarrow A((\lambda x.\mathtt{a}\,x\,x)\mathtt{e}) \longrightarrow A(\mathtt{a\,e\,e}) \longrightarrow (\lambda x.\mathtt{a}\,x\,x)(\mathtt{a\,e\,e}) \longrightarrow \mathtt{a(a\,e\,e)(a\,e\,e)}
\end{aligned}$$

Here, we have reduced order-2 redexes first (on the first three lines), and then order-1 redexes (on the last line). Here is the typing for the tree (we omit the outputs of transformations below):

$$\dfrac{\dfrac{\vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0 \quad \vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0}{\vdash_1 \mathtt{a\,e\,e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0} \quad \dfrac{\vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0 \quad \dfrac{\vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0}{\vdash_1 \mathtt{e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 1}\,\text{PT-MARK}}{\vdash_1 \mathtt{a\,e\,e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 3}}{\vdash_1 (\mathtt{a(a\,e\,e)(a\,e\,e)}) : (\emptyset,\{0\},\mathtt{o}) \triangleright 5}$$

By (two steps of of) subject expansion", we obtain:

$$\dfrac{\dfrac{\dfrac{\dfrac{x : (\{0\},\emptyset,\mathtt{o}) \vdash_1 x : (\{0\},\emptyset,\mathtt{o}) \triangleright 0 \quad x : (\emptyset,\{0\},\mathtt{o}) \vdash_1 x : (\emptyset,\{0\},\mathtt{o}) \triangleright 0}{x : (\{0\},\emptyset,\mathtt{o}), x : (\emptyset,\{0\},\mathtt{o}) \vdash_1 \mathtt{a}\,x\,x : (\emptyset,\{0\},\mathtt{o}) \triangleright 2}}{\vdash_1 \lambda x.\mathtt{a}\,x\,x : (\emptyset,\emptyset,(\{0\},\emptyset,\mathtt{o}) \wedge (\emptyset,\{0\},\mathtt{o}) \to \mathtt{o}) \triangleright 2}}{\vdash_1 A : (\emptyset,\emptyset,(\{0\},\emptyset,\mathtt{o}) \wedge (\emptyset,\{0\},\mathtt{o}) \to \mathtt{o}) \triangleright 2} \quad \dfrac{\cdots}{\vdash_1 \mathtt{a\,e\,e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0} \quad \dfrac{\cdots}{\vdash_1 \mathtt{a\,e\,e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 3}}{\vdash_1 A(\mathtt{a\,e\,e}) : (\emptyset,\{0\},\mathtt{o}) \triangleright 5} \cdots (*1)$$

By applying subject expansion to $\vdash_1 \mathtt{a\,e\,e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0$ and $\vdash_1 \mathtt{a\,e\,e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 3$, we also obtain:

$$\dfrac{\dfrac{\dfrac{\dfrac{x : (\{0\},\emptyset,\mathtt{o}) \vdash_1 x : (\{0\},\emptyset,\mathtt{o}) \quad x : (\{0\},\emptyset,\mathtt{o}) \vdash_1 x : (\{0\},\emptyset,\mathtt{o})}{x : (\{0\},\emptyset,\mathtt{o}) \vdash_1 \mathtt{a}\,x\,x : (\{0\},\emptyset,\mathtt{o}) \triangleright 0}}{\vdash_1 \lambda x.\mathtt{a}\,x\,x : (\{0\},\emptyset,(\{0\},\emptyset,\mathtt{o}) \to \mathtt{o}) \triangleright 0}}{\vdash_1 A : (\{0\},\emptyset,(\{0\},\emptyset,\mathtt{o}) \to \mathtt{o}) \triangleright 0} \quad \vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0}{\vdash_1 A\,\mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0}$$

$$\dfrac{\dfrac{\cdots}{\vdash_1 A : (\emptyset,\emptyset,(\{0\},\emptyset,\mathtt{o}) \wedge (\emptyset,\{0\},\mathtt{o}) \to \mathtt{o}) \triangleright 2} \quad \vdash_1 \mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0 \quad \vdash_1 \mathtt{e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 1}{\vdash_1 A\,\mathtt{e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 3}$$

Thus, by replacing the subderivations for $\mathtt{a\,e\,e}$ in (*1) with the two derivations above, we obtain:

$$\dfrac{\dfrac{\cdots}{\vdash_1 A : (\emptyset,\emptyset,(\{0\},\emptyset,\mathtt{o}) \wedge (\emptyset,\{0\},\mathtt{o}) \to \mathtt{o}) \triangleright 2} \quad \dfrac{\cdots}{\vdash_1 A\,\mathtt{e} : (\{0\},\emptyset,\mathtt{o}) \triangleright 0} \quad \dfrac{\cdots}{\vdash_1 A\,\mathtt{e} : (\emptyset,\{0\},\mathtt{o}) \triangleright 3}}{\vdash_1 A(A\,\mathtt{e}) : (\emptyset,\{0\},\mathtt{o}) \triangleright 5}$$

Let $\Theta_1 = x : (\{0\}, \emptyset, \mathsf{o})$, $\Theta_2 = x : (\emptyset, \{0\}, \mathsf{o})$, and $\Theta_3 = \Theta_1 + \Theta_2$. Then we have:

$$\frac{\dfrac{\dfrac{\cdots}{\vdash_1 A : (\emptyset, \emptyset, \rho_1) \triangleright 2} \quad \dfrac{\cdots}{\Theta_1 \vdash_1 A\, x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0} \quad \dfrac{\dfrac{\cdots}{\vdash_1 A : (\emptyset, \emptyset, \rho_1) \triangleright 2} \quad \Theta_1 \vdash_1 x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \Theta_2 \vdash_1 x : (\emptyset, \{0\}, \mathsf{o}) \triangleright 0}{\Theta_3 \vdash_1 A\, x : (\emptyset, \{0\}, \mathsf{o}) \triangleright 2}}{\dfrac{\Theta_3 \vdash_1 A(A\, x) : (\emptyset, \{0\}, \mathsf{o}) \triangleright 4}{\vdash_1 \lambda x.A(A\, x) : (\emptyset, \emptyset, \rho_1) \triangleright 4}} \quad \vdash_1 \mathsf{e} : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \vdash_1 \mathsf{e} : (\emptyset, \{0\}, \mathsf{o}) \triangleright 1}{\vdash_1 (\lambda x.A(A\, x))\mathsf{e} : (\emptyset, \{0\}, \mathsf{o}) \triangleright 5}$$

where $\rho_1 = (\{0\}, \emptyset, \mathsf{o}) \wedge (\emptyset, \{0\}, \mathsf{o}) \to \mathsf{o}$. Now, by increasing the order of the type system to 2, we obtain:

$$\frac{\dfrac{\dfrac{\cdots}{\vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0} \quad \dfrac{\cdots}{\Theta_1 \vdash_2 A\, x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0} \quad \dfrac{\dfrac{\cdots}{\vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0} \quad \Theta_1 \vdash_2 x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \Theta_2 \vdash_2 x : (\emptyset, \{0\}, \mathsf{o}) \triangleright 0}{\Theta_3 \vdash_2 A\, x : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}}{\dfrac{\Theta_3 \vdash_2 A(A\, x) : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}{\vdash_2 \lambda x.A(A\, x) : (\{1\}, \emptyset, \rho_1) \triangleright 0}} \quad \vdash_2 \mathsf{e} : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \vdash_2 \mathsf{e} : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}{\vdash_2 (\lambda x.A(A\, x))\mathsf{e} : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}$$

Note that now every counter (which now represent the numer of order-2 flags) has become 0, and instead order-1 flags have been set in the places where there were non-zero counters.

Let us now put an order-1 marker to the rightmost occurrence of A:

$$\frac{\dfrac{\dfrac{\cdots}{\vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0} \quad \dfrac{\cdots}{\Theta_1 \vdash_2 A\, x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0} \quad \dfrac{\dfrac{\dfrac{\cdots}{\vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0}}{\vdash_2 A : (\emptyset, \{1\}, \rho_1) \triangleright 1} \quad \Theta_1 \vdash_2 x : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \Theta_2 \vdash_2 x : (\emptyset, \{0\}, \mathsf{o}) \triangleright 0}{\Theta_3 \vdash_2 A\, x : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 1}}{\dfrac{\Theta_3 \vdash_2 A(A\, x) : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 2}{\vdash_2 \lambda x.A(A\, x) : (\emptyset, \{1\}, \rho_1) \triangleright 2}} \quad \vdash_2 \mathsf{e} : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \vdash_2 \mathsf{e} : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}{\vdash_2 (\lambda x.A(A\, x))\mathsf{e} : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 3}$$

Let $\Theta_4 = f : (\{1\}, \emptyset, \rho_1), f : (\{0\}, \emptyset, \rho_2), f : (\emptyset, \{1\}, \rho_1)$ where $\rho_2 = (\{0\}, \emptyset, \mathsf{o}) \to \mathsf{o}$. Then, we have:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\cdots}{\Theta_4 \vdash_2 \lambda x.f(f\, x) : (\emptyset, \{1\}, \rho_1) \triangleright 1}}{\vdash_2 \lambda f.\lambda x.f(f\, x) : (\emptyset, \emptyset, \rho_3) \triangleright 1}}{\vdash_2 T : (\emptyset, \emptyset, \rho_3) \triangleright 1} \quad \vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0 \quad \vdash_2 A : (\{0\}, \emptyset, \rho_2) \triangleright 0 \quad \vdash_2 A : (\emptyset, \{1\}, \rho_1) \triangleright 1}{\vdash_2 T\, A : (\emptyset, \{1\}, \rho_1) \triangleright 2} \quad \vdash_2 \mathsf{e} : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \vdash_2 \mathsf{e} : (\{1\}, \{0\}, \mathsf{o}) \triangleright 0}{\vdash_2 T\, A\, \mathsf{e} : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 3}}{\vdash_2 T\, A\, \mathsf{e} + R(T(TA)) : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 3}$$

where $\rho_3 = (\{1\}, \emptyset, \rho_1) \wedge (\{0\}, \emptyset, \rho_2) \wedge (\emptyset, \{1\}, \rho_1) \to \rho_1$.

Let $\Theta_5 = f : (\emptyset, \{1\}, \rho_1)$. Then we have:

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Theta_5 \vdash_2 f : (\emptyset, \{1\}, \rho_1) \triangleright 0 \quad \vdash_2 \mathsf{e} : (\{0\}, \emptyset, \mathsf{o}) \triangleright 0 \quad \vdash_2 \mathsf{e} : (\emptyset, \{0\}, \mathsf{o}) \triangleright 0}{\Theta_5 \vdash_2 f\, \mathsf{e} : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 1}}{\Theta_5 \vdash_2 f\, \mathsf{e} + R(T\, f) : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 1}}{\vdash_2 \lambda f.f\, \mathsf{e} + R(T\, f) : (\emptyset, \{0\}, (\emptyset, \{1\}, \rho_1) \to \mathsf{o}) \triangleright 1}}{\vdash_2 R : (\emptyset, \{0\}, (\emptyset, \{1\}, \rho_1) \to \mathsf{o}) \triangleright 1} \quad \dfrac{\cdots}{\vdash_2 TA : (\emptyset, \{1\}, \rho_1) \triangleright 2}}{\vdash_2 R(TA) : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 3}}{\vdash_2 A\, \mathsf{e} + R(TA) : (\emptyset, \{0,1\}, \mathsf{o}) \triangleright 3}$$

Let $\Theta_6 = f : (\{1\}, \emptyset, \rho_1), f : (\{0\}, \emptyset, \rho_2), f : (\emptyset, \{1\}, \rho_1)$. Then we have:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\cdots}{\vdash_2 R : (\emptyset, \{0\}, (\emptyset, \{1\}, \rho_1) \to \mathtt{o}) \triangleright 1} \quad \dfrac{\cdots}{\Theta_6 \vdash_2 T\, f : (\emptyset, \{1\}, \rho_1) \triangleright 1}}{\Theta_6 \vdash_2 R(T\, f) : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 2}}{\Theta_6 \vdash_2 f\, \mathtt{e} + R(T\, f) : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 2}}{\vdash_2 \lambda f. f\, \mathtt{e} + R(T\, f) : (\emptyset, \{0\}, \rho_5) \triangleright 2}}{\vdash_2 R : (\emptyset, \{0\}, \rho_5) \triangleright 2} \quad \vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0 \quad \vdash_2 A : (\{0\}, \emptyset, \rho_2) \triangleright 0 \quad \vdash_2 A : (\emptyset, \{1\}, \rho_1) \triangleright 1}{\dfrac{\vdash_2 R\, A : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 3}{\vdash_2 S : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 3}}$$

where $\rho_5 = (\{1\}, \emptyset, \rho_1) \wedge (\{0\}, \emptyset, \rho_2) \wedge (\emptyset, \{1\}, \rho_1) \to \mathtt{o}$. This completes the construction of a type derivation corresponding to the reduction sequence $S \longrightarrow^* \mathtt{a}(\mathtt{a}\,\mathtt{e}\,\mathtt{e})(\mathtt{a}\,\mathtt{e}\,\mathtt{e})$.

The derivation above does not contain a pumpable part. To obtain a pumpable derivation, let us prepare the following derivations:

$$\dfrac{\dfrac{\dfrac{\dfrac{f : (\{0\}, \emptyset, \rho_2) \vdash_2 f : (\{0\}, \emptyset, \rho_2) \triangleright 0 \quad \dfrac{\cdots}{f : (\{0\}, \emptyset, \rho_2), x : (\{0\}, \emptyset, \mathtt{o}) \vdash_2 f\, x : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}}{f : (\{0\}, \emptyset, \rho_2), x : (\{0\}, \emptyset, \mathtt{o}) \vdash_2 f(f\, x) : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}}{f : (\{0\}, \emptyset, \rho_2) \vdash_2 \lambda x. f(f\, x) : (\{0\}, \emptyset, \rho_2) \triangleright 0}}{\vdash_2 \lambda f. \lambda x. f(f\, x) : (\{0\}, \emptyset, (\{0\}, \emptyset, \rho_2) \to \rho_2) \triangleright 0}}{\vdash_2 T : (\{0\}, \emptyset, (\{0\}, \emptyset, \rho_2) \to \rho_2) \triangleright 0}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Theta_7 \vdash_2 f : (\{1\}, \emptyset, \rho_1) \triangleright 0 \quad \dfrac{\cdots}{\Theta_8, x : (\{0\}, \emptyset, \mathtt{o}) \vdash_2 f\, x : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0} \quad \dfrac{\cdots}{\Theta_7, \Theta_3 \vdash_2 f\, x : (\{1\}, \{0\}, \mathtt{o}) \triangleright 0}}{\Theta_9, \Theta_3 \vdash_2 f(f\, x) : (\{1\}, \{0\}, \mathtt{o}) \triangleright 0}}{\Theta_9 \vdash_2 \lambda x. f(f\, x) : (\{1\}, \emptyset, \rho_1) \triangleright 0}}{\vdash_2 \lambda f. \lambda x. f(f\, x) : (\{1\}, \emptyset, (\{1\}, \emptyset, \rho_1) \wedge (\{0\}, \emptyset, \rho_2) \to \rho_1) \triangleright 0}}{\vdash_2 T : (\{1\}, \emptyset, (\{1\}, \emptyset, \rho_1) \wedge (\{0\}, \emptyset, \rho_2) \to \rho_1) \triangleright 0}$$

where $\Theta_9 = f : (\{1\}, \emptyset, \rho_1), f : (\{0\}, \emptyset, \rho_2)$, $\Theta_7 = f : (\{1\}, \emptyset, \rho_1)$, and $\Theta_8 = f : (\{0\}, \emptyset, \rho_2)$.

Thus, we can also construct the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\cdots}{\vdash_2 R : (\emptyset, \{0\}, \rho_5) \triangleright 2} \quad \dfrac{\cdots}{\Theta_9 \vdash_2 T\, f : (\{1\}, \emptyset, \rho_1) \triangleright 0} \quad \dfrac{\cdots}{\Theta_8 \vdash_2 T\, f : (\emptyset, \{0\}, \rho_2) \triangleright 0} \quad \dfrac{\cdots}{\Theta_6 \vdash_2 T\, f : (\emptyset, \{1\}, \rho_1) \triangleright 1}}{\Theta_6 \vdash_2 R(T\, f) : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 3}}{\Theta_6 \vdash_2 f\, \mathtt{e} + R(T\, f) : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 3}}{\vdash_2 \lambda f. f\, \mathtt{e} + R(T\, f) : (\emptyset, \{0\}, \rho_5) \triangleright 3}}{\vdash_2 R : (\emptyset, \{0\}, \rho_5) \triangleright 3} \quad \vdash_2 A : (\{1\}, \emptyset, \rho_1) \triangleright 0 \quad \vdash_2 A : (\{0\}, \emptyset, \rho_2) \triangleright 0 \quad \vdash_2 A : (\emptyset, \{1\}, \rho_1) \triangleright 1}{\dfrac{\vdash_2 R\, A : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 4}{\vdash_2 S : (\emptyset, \{0, 1\}, \mathtt{o}) \triangleright 4}}$$

Thus, the subderivation of $\vdash_2 R : (\emptyset, \{0\}, \rho_5) \triangleright 3$ from $\vdash_2 R : (\emptyset, \{0\}, \rho_5) \triangleright 2$ is pumpable.

From the derivation, we obtain the following triples (C, D, t):

$$
\begin{aligned}
&C = [\,]\, t_{A,(\{1\},\emptyset,\rho_1)}\, t_{A,(\{0\},\emptyset,\rho_2)}\, t_{A,(\emptyset,\{1\},\rho_1)}\\
&t_{A,(\{1\},\emptyset,\rho_1)} = \lambda x_{(\{0\},\emptyset,\mathrm{o})}.\lambda x_{(\emptyset,\{0\},\mathrm{o})}.\mathtt{a}\, x_{(\{0\},\emptyset,\mathrm{o})} x_{(\emptyset,\{0\},\mathrm{o})}\\
&t_{A,(\{0\},\emptyset,\rho_2)} = \lambda x_{(\{0\},\emptyset,\mathrm{o})}.\mathtt{a}\, x_{(\{0\},\emptyset,\mathrm{o})} x_{(\{0\},\emptyset,\mathrm{o})}\\
&t_{A,(\emptyset,\{1\},\rho_1)} = t_{A,(\{1\},\emptyset,\rho_1)}\\
&D = \lambda f_{(\{1\},\emptyset,\rho_1)}.\lambda f_{(\{0\},\emptyset,\rho_2)}.\lambda f_{(\emptyset,\{1\},\rho_1)}.[\,]\, t_{Tf,(\{1\},\emptyset,\rho_1)}\, t_{Tf,(\{0\},\emptyset,\rho_2)}\, t_{Tf,(\emptyset,\{1\},\rho_1)}\\
&t_{Tf,(\{1\},\emptyset,\rho_1)} = \lambda x_{(\{0\},\emptyset,\mathrm{o})}.\lambda x_{(\emptyset,\{0\},\mathrm{o})}.f_{(\{1\},\emptyset,\rho_1)}(f_{(\{0\},\emptyset,\rho_2)} x_{(\{0\},\emptyset,\mathrm{o})})(f_{(\{1\},\emptyset,\rho_1)}\, x_{(\{0\},\emptyset,\mathrm{o})} x_{(\emptyset,\{0\},\mathrm{o})})\\
&t_{Tf,(\{0\},\emptyset,\rho_2)} = \lambda x_{(\{0\},\emptyset,\mathrm{o})}.f_{(\{0\},\emptyset,\rho_2)}(f_{(\{0\},\emptyset,\rho_2)} x_{(\{0\},\emptyset,\mathrm{o})})\\
&t_{Tf,(\emptyset,\{1\},\rho_1)} = \lambda x_{(\{0\},\emptyset,\mathrm{o})}.\lambda x_{(\emptyset,\{0\},\mathrm{o})}.f_{(\{1\},\emptyset,\rho_1)}(f_{(\{0\},\emptyset,\rho_2)} x_{(\{0\},\emptyset,\mathrm{o})})(f_{(\emptyset,\{1\},\rho_1)}\, x_{(\{0\},\emptyset,\mathrm{o})} x_{(\emptyset,\{0\},\mathrm{o})})\\
&t = \lambda f_{(\{1\},\emptyset,\rho_1)}.\lambda f_{(\{0\},\emptyset,\rho_2)}.\lambda f_{(\emptyset,\{1\},\rho_1)}.t_{R,(\emptyset,\{0\},(\emptyset,\{1\},\rho_1)\to\mathrm{o})}(t_{Tf,(\emptyset,\{1\},\rho_1)})\\
&t_{R,(\emptyset,\{0\},(\emptyset,\{1\},\rho_1)\to\mathrm{o})} = \lambda f_{(\emptyset,\{1\},\rho_1)}.f_{(\emptyset,\{1\},\rho_1)}\,\mathtt{e}\,\mathtt{e}
\end{aligned}
$$

For the readability, let us rename variables and terms.

$$
\begin{aligned}
&C = [\,]\, t_{A,0}\, t_{A,1}\, t_{A,2}\\
&t_{A,0} = t_{A,2} = \lambda x_0.\lambda x_1.\mathtt{a}\, x_0\, x_1\\
&t_{A,1} = \lambda x_0.\mathtt{a}\, x_0\, x_0\\
&D = \lambda f_0.\lambda f_1.\lambda f_2.[\,]\, t_{Tf,0}\, t_{Tf,1}\, t_{Tf,2}\\
&t_{Tf,0} = \lambda x_0.\lambda x_1.f_0(f_1 x_0)(f_0\, x_0 x_1)\\
&t_{Tf,1} = \lambda x_0.f_1(f_1\, x_0)\\
&t_{Tf,2} = \lambda x_0.\lambda x_1.f_0(f_1\, x_0)(f_2\, x_0\, x_1)\\
&t = \lambda f_0.\lambda f_1.\lambda f_2.t_R(t_{Tf,2})\\
&t_R = \lambda f_2.f_2\,\mathtt{e}\,\mathtt{e}
\end{aligned}
$$

Note that C and D are linear.

The corresponding OCaml code, and sample runs are given below.

```
(* f0 = f_{({1},{},ro_1)}, f1 = f_{({0},{},ro_1)}, f2 = f_{({},{1},ro_1)} *)
(* x0 = x_{({0},{},T)}, x1 = x_{({},{0},T)} *)

(* tree constructors *)
type tree = A of tree * tree | E

(* context C *)
let tA0 = fun x0 x1 -> A(x0, x1)
let tA1 = fun x0 -> A(x0, x0)
let tA2 = tA0
let c g = g tA0 tA1 tA2

(* context D *)
let tT0 f0 f1 = fun x0 x1 -> f0 (f1 x0) (f0 x0 x1)
let tT1 f1 =  fun x0 -> f1 (f1 x0)
let tT2 f0 f1 f2 =  fun x0 x1 -> f0 (f1 x0) (f2 x0 x1)
let d g = fun f0 f1 f2 -> g (tT0 f0 f1) (tT1 f1) (tT2 f0 f1 f2)

(* term t *)
let tR = fun f2 -> f2 E E
let t = fun f0 f1 f2 -> tR(tT2 f0 f1 f2)

(*  sample execution
# let t0 = c t;;
val t0 : tree = A (A (E, E), A (E, E))
# let t1 = c (d t);;
```

```
val t1 : tree =
  A (A (A (A (E, E), A (E, E)), A (A (E, E), A (E, E))),
   A (A (A (E, E), A (E, E)), A (A (E, E), A (E, E))))
# let t2 = c (d (d t));;
val t2 : tree =
  A
   (A
     (A
       (A
         (A (A (A (A (E, E), A (E, E)), A (A (E, E), A (E, E))),
  ...
*)
```

The following is a direction-annotated version of (C, D, t):

$$
\begin{array}{l}
C' = [\,]\, t'_{A,0}\, t'_{A,1}\, t'_{A,2} \\
t'_{A,0} = t'_{A,2} = \lambda x_0.\lambda x_1.\mathtt{a}^{\langle 1\rangle}\, x_0\, x_1 + \mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 \\
t'_{A,1} = \lambda x_0.\mathtt{a}^{\langle 1\rangle}\, x_0 x_0 + \mathtt{a}^{\langle 2\rangle}\, x_0\, x_0 \\
D' = \lambda f_0.\lambda f_1.\lambda f_2.[\,]\, t'_{Tf,0}\, t'_{Tf,1}\, t'_{Tf,2} \\
t'_{Tf,0} = \lambda x_0.\lambda x_1.f_0(f_1 x_0)(f_0\, x_0 x_1) \\
t'_{Tf,1} = \lambda x_0.f_1(f_1\, x_0) \\
t'_{Tf,2} = \lambda x_0.\lambda x_1.f_0(f_1\, x_0)(f_2\, x_0\, x_1) \\
t' = \lambda f_0.\lambda f_1.\lambda f_2.t'_R(t'_{Tf,2}) \\
t'_R = \lambda f_2.f_2\, \mathtt{e}^{\langle 0\rangle}\, \mathtt{e}^{\langle 0\rangle}
\end{array}
$$

Let us construct a type derivation for C', D', and t'. A derivation for t' is:

$$
\dfrac{\dfrac{\vdash_2 t'_R : (\emptyset,\{0\},(\emptyset,\{1\},\rho'_1)\to \mathtt{o})\rhd 1 \qquad \Theta'_6 \vdash_2 t'_{Tf,2} : (\emptyset,\{1\},\rho'_1)\rhd 1}{\Theta'_6 \vdash_2 t'_R(t'_{Tf,2}) : (\emptyset,\{0,1\},\mathtt{o})\rhd 2}}{\vdash_2 t' : (\emptyset,\{0\},\rho'_5)\rhd 2}
$$

$$
\dfrac{\dfrac{\dfrac{f_2 : (\emptyset,\{1\},\rho'_1) \vdash_2 f_2 : (\emptyset,\{1\},\rho'_1)\rhd 0 \qquad \vdash_2 \mathtt{e}^{\langle 0\rangle} : (\{0\},\emptyset,\mathtt{o})\rhd 0}{f_2 : (\emptyset,\{1\},\rho'_1) \vdash_2 f_2\, \mathtt{e}^{\langle 0\rangle} : (\{0\},\{1\},(\emptyset,\{0\},\mathtt{o})\to\mathtt{o})\rhd 0} \qquad \dfrac{\vdash_2 \mathtt{e}^{\langle 0\rangle} : (\{0\},\emptyset,\mathtt{o})\rhd 0}{\vdash_2 \mathtt{e}^{\langle 0\rangle} : (\emptyset,\{0\},\mathtt{o})\rhd 0}}{f_2 : (\emptyset,\{1\},\rho'_1) \vdash_2 f_2\, \mathtt{e}^{\langle 0\rangle}\, \mathtt{e}^{\langle 0\rangle} : (\emptyset,\{0,1\},\mathtt{o})\rhd 1}}{\vdash_2 t'_R : (\emptyset,\{0\},(\emptyset,\{1\},\rho'_1)\to\mathtt{o})\rhd 1}
$$

$$
\dfrac{\dfrac{\Theta'_9,\Theta'_1 \vdash_2 f_0(f_1 x_0) : (\{0,1\},\emptyset,(\emptyset,\{0\},\mathtt{o})\to\mathtt{o})\rhd 0 \qquad \dfrac{\dfrac{\Theta'_{10} \vdash_2 f_2 : (\emptyset,\{1\},\rho'_1)\rhd 0 \quad \Theta'_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathtt{o})\rhd 0}{\Theta'_{10},\Theta'_1 \vdash_2 f_2 x_0 : (\emptyset,\{1\},(\emptyset,\{0\},\mathtt{o})\to\mathtt{o})\rhd 0} \qquad \Theta'_2 \vdash_2 x_1 : (\emptyset,\{0\},\mathtt{o})\rhd 0}{\Theta'_{10},\Theta'_3 \vdash_2 f_2 x_0 x_1 : (\emptyset,\{0,1\},\mathtt{o})\rhd 0}}{\Theta'_6,\Theta'_3 \vdash_2 f_0(f_1 x_0)(f_2 x_0 x_1) : (\emptyset,\{0,1\},\mathtt{o})\rhd 1}}{\Theta'_6 \vdash_2 t'_{Tf,2} : (\emptyset,\{1\},\rho'_1)\rhd 1}
$$

$$
\dfrac{\Theta'_7 \vdash_2 f_0 : (\{1\},\emptyset,\rho'_1)\rhd 0 \qquad \dfrac{\Theta'_8 \vdash_2 f_1 : (\{0\},\emptyset,\rho'_2\rhd 0) \quad \Theta_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathtt{o})\rhd 0}{\Theta'_8,\Theta'_1 \vdash_2 f_1 x_0 : (\{0\},\emptyset,\mathtt{o})\rhd 0}}{\Theta'_9,\Theta'_1 \vdash_2 f_0(f_1 x_0) : (\{0,1\},\emptyset,(\emptyset,\{0\},\mathtt{o})\to\mathtt{o})\rhd 0}
$$

where

$$\begin{aligned}
\Theta'_1 &= x_0 : (\{0\},\emptyset,\mathsf{o})\\
\Theta'_2 &= x_1 : (\emptyset,\{0\},\mathsf{o})\\
\Theta'_3 &= x_0 : (\{0\},\emptyset,\mathsf{o}), x_1 : (\emptyset,\{0\},\mathsf{o})\\
\Theta'_6 &= f_0 : (\{1\},\emptyset,\rho'_1), f_1 : (\{0\},\emptyset,\rho'_2), f_2 : (\emptyset,\{1\},\rho'_1)\\
\Theta'_7 &= f_0 : (\{1\},\emptyset,\rho'_1)\\
\Theta'_8 &= f_1 : (\{0\},\emptyset,\rho'_2)\\
\Theta'_9 &= f_0 : (\{1\},\emptyset,\rho'_1), f_1 : (\{0\},\emptyset,\rho'_2)\\
\Theta'_{10} &= f_2 : (\emptyset,\{1\},\rho'_1)\\
\rho'_1 &= (\{0\},\emptyset,\mathsf{o}) \to (\emptyset,\{0\},\mathsf{o}) \to \mathsf{o}\\
\rho'_2 &= (\{0\},\emptyset,\mathsf{o}) \to \mathsf{o}\\
\rho'_5 &= (\{1\},\emptyset,\rho'_1) \to (\{0\},\emptyset,\rho'_2) \to (\emptyset,\{1\},\rho'_1) \to \mathsf{o}
\end{aligned}$$

A derivation for C' is:

$$\frac{\vdash_2 [\,] : (\emptyset,\{0\},\rho_5) \triangleright 3 \quad \vdash_2 t'_{A,0} : (\{1\},\emptyset,\rho'_1) \triangleright 0 \quad \vdash_2 t'_{A,1} : (\{0\},\emptyset,\rho'_2) \triangleright 0 \quad \vdash_2 t'_{A,2} : (\emptyset,\{1\},\rho'_1) \triangleright 1}{\vdash_2 C' : (\emptyset,\{0,1\},\mathsf{o}) \triangleright 4}$$

where

$$\frac{\dfrac{\dfrac{\Theta'_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathsf{o}) \quad \Theta'_2 \vdash_2 x_1 : (\emptyset,\{0\},\mathsf{o})}{\Theta'_3 \vdash_2 \mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 : (\{1\},\{0\},\mathsf{o}) \triangleright 0}}{\Theta'_3 \vdash_2 \mathtt{a}^{\langle 1\rangle}\, x_0\, x_1 + \mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 : (\{1\},\{0\},\mathsf{o}) \triangleright 0}}{\vdash_2 t'_{A,0} : (\{1\},\emptyset,\rho'_1) \triangleright 0}$$

$$\frac{\dfrac{\dfrac{\Theta'_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathsf{o})}{\Theta'_1 \vdash_2 \mathtt{a}^{\langle 2\rangle}\, x_0\, x_0 : (\{0\},\emptyset,\mathsf{o}) \triangleright 0}}{\Theta'_1 \vdash_2 \mathtt{a}^{\langle 1\rangle}\, x_0\, x_0 + \mathtt{a}^{\langle 2\rangle}\, x_0\, x_0 : (\{0\},\emptyset,\mathsf{o}) \triangleright 0}}{\vdash_2 t'_{A,1} : (\{0\},\emptyset,\rho'_2) \triangleright 0}$$

$$\frac{\dfrac{\dfrac{\Theta'_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathsf{o}) \quad \dfrac{\Theta'_2 \vdash_2 x_1 : (\emptyset,\{0\},\mathsf{o}) \triangleright 0}{\Theta'_2 \vdash_2 x_1 : (\emptyset,\{0,1\},\mathsf{o}) \triangleright 0}}{\Theta'_3 \vdash_2 \mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 : (\emptyset,\{0,1\},\mathsf{o}) \triangleright 1}}{\Theta'_3 \vdash_2 \mathtt{a}^{\langle 1\rangle}\, x_0\, x_1 + \mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 : (\emptyset,\{0,1\},\mathsf{o}) \triangleright 1}}{\vdash_2 t'_{A,2} : (\emptyset,\{1\},\rho'_1) \triangleright 1}$$

A derivation for D' is:

$$\frac{\dfrac{\vdash_2 [\,] : (\emptyset,\{0\},\rho_5) \triangleright 2 \quad \Theta'_9 \vdash_2 t'_{Tf,0} : (\{1\},\emptyset,\rho'_1) \triangleright 0 \quad \Theta'_8 \vdash_2 t'_{Tf,1} : (\{0\},\emptyset,\rho'_2) \triangleright 0 \quad \dfrac{\dots}{\Theta'_6 \vdash_2 t'_{Tf,2} : (\emptyset,\{1\},\rho'_1) \triangleright 1}}{\Theta'_6 \vdash_2 [\,]\, t'_{Tf,0} t'_{Tf,1} t'_{Tf,2} : (\emptyset,\{0,1\},\mathsf{o}) \triangleright 3}}{\vdash_2 D' : (\emptyset,\{0\},\rho_5) \triangleright 3}$$

where:

$$\frac{\dfrac{\Theta'_9, \Theta'_1 \vdash_2 f_0(f_1 x_0) : (\{0,1\},\emptyset,(\emptyset,\{0\},\mathsf{o}) \to \mathsf{o}) \triangleright 0 \quad \dfrac{\dfrac{\Theta'_7 \vdash_2 f_0 : (\{1\},\emptyset,\rho'_1) \triangleright 0 \quad \Theta'_1 \vdash_2 x_0 : (\{0\},\emptyset,\mathsf{o}) \triangleright 0}{\Theta'_7, \Theta'_1 \vdash_2 f_0 x_0 : (\{0,1\},\emptyset,(\emptyset,\{0\},\mathsf{o}) \to \mathsf{o}) \triangleright 0} \quad \Theta'_2 \vdash_2 x_1 : (\emptyset,\{0\},\mathsf{o}) \triangleright 0}{\Theta'_7, \Theta'_3 \vdash_2 f_0 x_0 x_1 : (\{1\},\{0\},\mathsf{o}) \triangleright 0}}{\Theta'_9, \Theta'_3 \vdash_2 f_0(f_1 x_0)(f_0 x_0 x_1) : (\{1\},\{0\},\mathsf{o}) \triangleright 0}}{\Theta'_9 \vdash_2 t'_{Tf,0} : (\{1\},\emptyset,\rho'_1) \triangleright 0}$$

$$\dfrac{\Theta'_8 \vdash_2 f_1 : (\{0\}, \emptyset, \rho'_2) \triangleright 0 \qquad \dfrac{\Theta'_8 \vdash_2 f_1 : (\{0\}, \emptyset, \rho'_2) \triangleright 0 \quad \Theta'_1 \vdash_2 x_0 : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}{\Theta'_8, \Theta'_1 \vdash_2 f_1 x_0 : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}}{\dfrac{\Theta'_8, \Theta'_1 \vdash_2 f_1(f_1 x_0) : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}{\Theta'_8 \vdash_2 t'_{Tf,1} : (\{0\}, \emptyset, \rho'_2) \triangleright 0}}$$

It turns out that the derivation for $C'[D'[t']]$ is pumpable. We obtain the following triple (G, H, u) that satisfies the condition of Lemma 12.

$$\begin{array}{l}
G = [\,]\, u_{A,0}\, u_{A,1}\, u_{A,2} \\
u_{A,0} = u_{A,2} = \lambda x_0.\lambda x_1.\mathtt{a}^{\langle 2\rangle}\, x_0\, x_1 \\
u_{A,1} = \lambda x_0.\mathtt{a}^{\langle 2\rangle}\, x_0\, x_0 \\
H = \lambda f_0.\lambda f_1.\lambda f_2.[\,]\, u_{Tf,0}\, u_{Tf,1}\, u_{Tf,2} \\
u_{Tf,0} = \lambda x_0.\lambda x_1.f_0(f_1 x_0)(f_0\, x_0 x_1) \\
u_{Tf,1} = \lambda x_0.f_1(f_1\, x_0) \\
u_{Tf,2} = \lambda x_0.\lambda x_1.f_0(f_1\, x_0)(f_2\, x_0\, x_1) \\
u = \lambda f_0.\lambda f_1.\lambda f_2.u_R(u_{Tf,2}) \\
u_R = \lambda f_2.f_2\, \mathtt{e}^{\langle 0\rangle}\, \mathtt{e}^{\langle 0\rangle}
\end{array}$$

In this case, (G, H, u) is the same as (C, D, t) except that $\mathtt{a}$ and $\mathtt{e}$ are annotated with the directions 2 and 0 respectively.

Now, let us apply the induction in the proof of Lemma 16. The triple (G_p, H_p, u_p) is given by:

$$\begin{array}{l}
G_p = [\,]\, u'_{A,0}\, u'_{A,1}\, u'_{A,2} \\
u'_{A,0} = u'_{A,2} = \lambda x_0.\lambda x_1.\mathtt{a}^{\langle 2\rangle} x_1 \\
u'_{A,1} = \lambda x_0.\mathtt{a}^{\langle 2\rangle}\, x_0 \\
H_p = \lambda f_0.\lambda f_1.\lambda f_2.[\,]\, u'_{Tf,0}\, u'_{Tf,1}\, u'_{Tf,2} \\
u'_{Tf,0} = \lambda x_0.\lambda x_1.f_0(f_1 x_0)(f_0\, x_0 x_1) \\
u'_{Tf,1} = \lambda x_0.f_1(f_1\, x_0) \\
u'_{Tf,2} = \lambda x_0.\lambda x_1.f_0(f_1\, x_0)(f_2\, x_0\, x_1) \\
u_p = \lambda f_0.\lambda f_1.\lambda f_2.u'_R(u'_{Tf,2}) \\
u'_R = \lambda f_2.f_2\, \mathtt{e}^{\langle 0\rangle}\, \mathtt{e}^{\langle 0\rangle}
\end{array}$$

By applying the first of the word-to-leaves transformation (in Section B.1), we obtain the following order-1 contexts and terms.

$$\begin{array}{l}
G'_p = [\,]\, u''_{A,0}\, u''_{A,2} \\
u''_{A,0} = u''_{A,2} = \mathtt{br}\, \mathtt{a}^{\langle 2\rangle}\, \mathtt{e} \\
H'_p = \lambda f_0.\lambda f_2.[\,]\, u''_{Tf,0}\, u''_{Tf,2} \\
u''_{Tf,0} = \mathtt{br}\, f_0\, (\mathtt{br}\, f_0\, \mathtt{e}) \\
u''_{Tf,2} = \mathtt{br}\, f_0\, (\mathtt{br}\, f_2\, \mathtt{e}) \\
u'_p = \lambda f_0.\lambda f_2.u''_R(u''_{Tf,2}) \\
u''_R = \lambda f_2.\mathtt{br}\, f_2\, \mathtt{e}^{\langle 0\rangle}
\end{array}$$

By applying the second transformation (in Section B.2) to eliminate redundant occurrences of e, we obtain:

$$G_l = [\,]\, v_{A,0}\, v_{A,2}$$
$$v_{A,0} = v_{A,2} = \mathtt{a}^{\langle 2\rangle}$$
$$H_l = \lambda f_0.\lambda f_2.[\,]\, v_{Tf,0}\, v_{Tf,2}$$
$$v_{Tf,0} = \mathtt{br}\ f_0\ f_0$$
$$v_{Tf,2} = \mathtt{br}\ f_0\ f_2$$
$$u_l = \lambda f_0.\lambda f_2.v_R(v_{Tf,2})$$
$$v_R = \lambda f_2.f_2$$

The corresponding OCaml code, and sample runs are as follows.

```
type tree1 = Br of tree1*tree1 | LeafE | LeafA
(* G_l *)
let vA0 = LeafA
let vA2 = vA0
let gl g = g vA0 vA2
(* H_l *)
let vTf0 f0 = Br(f0,f0)
let vTf2 f0 f2 = Br(f0,f2)
let hl g = fun f0 f2 -> g (vTf0 f0) (vTf2 f0 f2)
(* u_l *)
let vR = fun f2 -> f2
let ul = fun f0 f2 -> vR (vTf2 f0 f2)

(* sample runs
# gl ul;;
- : tree1 = Br (LeafA, LeafA)
# gl(hl ul);;
- : tree1 = Br (Br (LeafA, LeafA), Br (LeafA, LeafA))
# gl(hl (hl ul));;
- : tree1 =
Br (Br (Br (LeafA, LeafA), Br (LeafA, LeafA)),
 Br (Br (LeafA, LeafA), Br (LeafA, LeafA)))
*)
```

By applying the induction hypothesis of Lemma 16, we obtain a pumping sequence (where we can choose $j = 0$ and $k = 1$ in this case):

$$\mathbf{leaves}(\mathcal{T}(G_l[{H_l}^0[u_l]])) \prec \mathbf{leaves}(\mathcal{T}(G_l[{H_l}^1[u_l]])) \prec \mathbf{leaves}(\mathcal{T}(G_l[{H_l}^2[u_l]])) \prec \cdots,$$

and hence we also obtain:

$$\mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^0[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^1[u_{\mathrm{p}}]]) \prec \mathcal{T}(G_{\mathrm{p}}[{H_{\mathrm{p}}}^2[u_{\mathrm{p}}]]) \prec \cdots.$$

We also happen to have a trivial periodic sequence:

$$H^0[u] \preceq_\kappa H^1[u] \preceq_\kappa H^2[u] \preceq_\kappa \cdots$$

for $\kappa = (\mathtt{o} \to \mathtt{o} \to \mathtt{o}) \to (\mathtt{o} \to \mathtt{o}) \to (\mathtt{o} \to \mathtt{o} \to \mathtt{o}) \to \mathtt{o}$. Thus, we also have

$$\mathcal{T}(G[H^0[u]]) \prec \mathcal{T}(G[H^1[u]]) \prec \mathcal{T}(G[H^2[u]]) \prec \cdots.$$

Therefore, we finally obtain:

$$\mathcal{T}(C[t]) \prec \mathcal{T}(C[D[t]]) \prec \mathcal{T}(C[D^2[t]]) \prec \cdots.$$

A.6.2 Other examples

One may expect that the triple (C, D, t) obtained from a pumpable derivation always satisfies

$$|\mathcal{T}(C[t])| < |\mathcal{T}(C[D[t]])| < |\mathcal{T}(C[D[D[t]]])| < \cdots$$

because the counter values of the derivations for $C[D^i[t]]$ strictly increase with respect to i. Below we give examples where that is not the case (although there do exist j, k such that $|\mathcal{T}(C[D^j[t]]) < |\mathcal{T}(C[D^{j+k}[t]])|$ as guaranteed by Lemma 16).

Consider the following order-1 grammar:

$$\begin{array}{l} S \to A(a(a(e))). \\ A\ x \to x. \\ A\ x \to a(A(e)). \end{array}$$

We have the following pumpable derivation (where we omit the target of the transformation):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : (\{0\}, \emptyset, \circ) \vdash_1 x : (\{0\}, \emptyset, \circ) \triangleright 0}}{x : (\{0\}, \emptyset, \circ) \vdash_1 x : (\emptyset, \{0\}, \circ) \triangleright 1}}{\vdash_1 \lambda x.x : (\emptyset, \{0\}, (\{0\}, \emptyset, \circ) \to \circ) \triangleright 1}}{\vdash_1 A : (\emptyset, \{0\}, (\{0\}, \emptyset, \circ) \to \circ) \triangleright 1} \quad \dfrac{}{\vdash_1 e : (\{0\}, \emptyset, \circ) \triangleright 0}}{\vdash_1 A(e) : (\emptyset, \{0\}, \circ) \triangleright 1}}{\vdash_1 a(A(e)) : (\emptyset, \{0\}, \circ) \triangleright 2}}{x : (\{0\}, \emptyset, \circ) \vdash_1 a(A(e)) : (\emptyset, \{0\}, \circ) \triangleright 2}}{\vdash_1 \lambda x.a(A(e)) : (\emptyset, \{0\}, (\{0\}, \emptyset, \circ) \to \circ) \triangleright 2}}{\vdash_1 A : (\emptyset, \{0\}, (\{0\}, \emptyset, \circ) \to \circ) \triangleright 2} \quad \dfrac{\cdots}{\vdash_1 a(a(e)) : (\{0\}, \emptyset, \circ) \triangleright 0}}{\dfrac{\vdash_1 A(a(a(e))) : (\emptyset, \{0\}, \circ) \triangleright 2}{\vdash_1 S : (\emptyset, \{0\}, \circ) \triangleright 2}}$$

The triple (C, D, t) obtained from the above derivation is:

$$C = [\,]\,(a(a(e))) \qquad D = \lambda x.a([\,]\,e) \qquad t = \lambda x.x.$$

Notice that $C[t] \longrightarrow^* a(a(e))$ and $C[D[t]] \longrightarrow^* a((\lambda x.x)e) \longrightarrow^* a(e)$; thus, $|\mathcal{T}(C[t])| \not< |\mathcal{T}(C[D[t]])|$. Note, however, that since $\mathcal{T}(C[D^k[t]]) = a^k(e)$ for $k > 0$, we have:

$$\mathcal{T}(C[D[t]]) \prec \mathcal{T}(C[D^2[t]]) \prec \mathcal{T}(C[D^3[t]]) \prec \cdots.$$

Thus, for the triple above, Lemma 16 holds for $j = k = 1$.

Consider the following order-2 grammar:

$$\begin{array}{l} S \to A(\lambda xy.y). \\ A\ f \to f\ e\ (a^2\,e). \\ A\ f \to a(A(\lambda xy.f\,y\,x)). \end{array}$$

The following is a pumpable derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots}{f : (\{0\}, \emptyset, \rho_1) \vdash_2 f\, e\, (a^2\, e) : (\{0\}, \emptyset, \mathtt{o}) \triangleright 0}}{f : (\{0\}, \emptyset, \rho_1) \vdash_2 f\, e\, (a^2\, e) : (\emptyset, \{0,1\}, \mathtt{o}) \triangleright 1}}{\vdash_2 \lambda f.f\, e\, (a^2\, e) : (\emptyset, \{0,1\}, (\{0\}, \emptyset, \rho_1) \to \mathtt{o}) \triangleright 1}}{\vdash_2 A : (\emptyset, \{0,1\}, (\{0\}, \emptyset, \rho_1) \to \mathtt{o}) \triangleright 1} \quad \dfrac{\dots}{f : (\{0, \emptyset, \rho_1\}) \vdash_2 \lambda xy.f\, y\, x : (\{0\}, \emptyset, \rho_1) \triangleright 0}}{f : (\{0, \emptyset, \rho_1\}) \vdash_2 A(\lambda xy.f\, y\, x) : (\emptyset, \{0,1\}, \mathtt{o}) \triangleright 1}}{f : (\{0, \emptyset, \rho_1\}) \vdash_2 a(A(\lambda xy.f\, y\, x)) : (\emptyset, \{0,1\}, \mathtt{o}) \triangleright 2}}{\vdash_2 \lambda f.a(A(\lambda xy.f\, y\, x)) : (\emptyset, \{0,1\}, (\{0\}, \emptyset, \rho_1) \to \mathtt{o}) \triangleright 2}}{\vdash_2 A : (\emptyset, \{0,1\}, (\{0\}, \emptyset, \rho_1) \to \mathtt{o}) \triangleright 2} \quad \dfrac{\dots}{\vdash_2 \lambda xy.y : (\{0\}, \emptyset, \rho_1) \triangleright 0}}{\vdash_2 A(\lambda xy.y) : (\emptyset, \{0,1\}, \mathtt{o}) \triangleright 2}}{\vdash_2 S : (\emptyset, \{0,1\}, \mathtt{o}) \triangleright 2}$$

where $\rho_1 = (\{0\}, \emptyset, \mathtt{o}) \to (\{0\}, \emptyset, \mathtt{o}) \to \mathtt{o})$. The triple obtained from the above derivation is (C, D, t) where:

$$C = [\,]\lambda xy.y \qquad D = \lambda f.a([\,](\lambda xy.f\, y\, x)) \qquad t = \lambda f.f\, e\, (a^2\, e).$$

We have:

$$\mathcal{T}(C[D^i[t]]) = \begin{cases} a^{i+2}(e) & \text{if } i \text{ is even} \\ a^i(e) & \text{if } i \text{ is odd} \end{cases}$$

Thus, $|\mathcal{T}(C[D^{2i}[t]])| \not< |\mathcal{T}(C[D^{2i+1}[t]])|$. We, however, have:

$$C[D^j[t]] \prec C[D^{j+k}[t]] \prec C[D^{j+2k}[t]] \prec \cdots$$

for $j = 0$, $k = 2$.

B Word-to-leaves Transformations

Here we prove Lemma 13. As explained in Section 4, we use a modified version of the transformation given in [1]. The transformation in [1] consists of two steps, and we restrict each of the two transformations to $\lambda^{\to}$-terms so that the two restricted transformations return again $\lambda^{\to}$-terms. We call the two steps *first* and *second* transformations, and the composite the *whole* transformation. To distinguish the transformations in this paper and those in [1], we call the latter *original first/second/whole transformations*.

In Sections B.1 and B.2 we give the definitions of the first and second transformations, respectively; all the definitions in Sections B.1 and B.2 except for Figures 1 and 2 are taken from [1]. Then, we show that these transformations preserve meaning (in Section B.3) and linearity (in Section B.4).

B.1 First Transformation

The first transformation is applied to order-$(n+1)$ $\lambda^{\to}$-terms of a word alphabet and outputs order-n $\lambda^{\to}$-terms of a $\mathtt{br}$-alphabet. Constants of type $\mathtt{o} \to \mathtt{o}$ before the transformation have type $\mathtt{o}$ after the transformation. This first transformation achieves the purpose of the whole transformation except that an output term might not be $\mathtt{e}$-free $\mathtt{br}$-tree: e.g., $(\lambda x.x)(\mathtt{a}\,\mathtt{e})$ is transformed to $\mathtt{br}\,\mathtt{e}\,(\mathtt{br}\,\mathtt{a}\,\mathtt{e})$, whose leaves, $\mathtt{e}\,\mathtt{a}\,\mathtt{e}$, have extra $\mathtt{e}$. Such extra $\mathtt{e}$'s will be removed by the second transformation. We write $\mathtt{br}\, t\, s$ also as $t * s$ and write $((t_1 * t_2) \cdots * t_m)$ as $t_1 * \cdots * t_m$.

Same as the original first transformation, for technical convenience, we assume below that, for every type κ occurring in the simple type derivation of an input $\lambda^{\rightarrow}$-term, if κ is of the form $\mathtt{o} \to \kappa'$, then $\mathtt{order}(\kappa') \leq 1$. This does not lose generality, since any function $\lambda x : \mathtt{o}.t$ of type $\mathtt{o} \to \kappa'$ with $\mathtt{order}(\kappa') > 1$ can be replaced by the term $\lambda x' : \mathtt{o} \to \mathtt{o}.[x'\mathtt{e}/x]t$ of type $(\mathtt{o} \to \mathtt{o}) \to \kappa'$ (without changing the order of the term), and any term t of type $\mathtt{o}$ can be replaced by the term $(\lambda x, y.x)\, t$ of type $\mathtt{o} \to \mathtt{o}$ (see [1, Appendix D] for the detail). This preprocessing transformation preserves order, meaning, and linearity.

For the first transformation, we use the following intersection types:

$$\delta ::= \mathtt{o} \mid \sigma \to \delta \qquad \sigma ::= \delta_1 \wedge \cdots \wedge \delta_k \quad (k \geq 0)$$

We write $\top$ for $\delta_1 \wedge \cdots \wedge \delta_k$ when $k = 0$. We assume some total order $<$ on intersection types, and require that $\delta_1 < \cdots < \delta_k$ whenever $\delta_1 \wedge \cdots \wedge \delta_k$ occurs in an intersection type. Intuitively, if a function f has type $\delta_1 \wedge \cdots \wedge \delta_k \to \delta$, then f uses an argument (in k-number of different ways), and if f has $\top \to \delta$, then f does not use an argument.

We introduce two refinement relations $\delta ::_{\mathrm{b}} \kappa$ and $\delta ::_{\mathrm{u}} \kappa$. The relations are defined as follows, by mutual induction; k may be 0.

$$\frac{\delta_j ::_{\mathrm{u}} \kappa \qquad j \in \{1, \dots, k\} \qquad \delta_i ::_{\mathrm{b}} \kappa \text{ (for each } i \in \{1, \dots, k\} \setminus \{j\})}{\delta_1 \wedge \cdots \wedge \delta_k ::_{\mathrm{u}} \kappa} \qquad \frac{\delta_i ::_{\mathrm{b}} \kappa \text{ (for each } i \in \{1, \dots, k\})}{\delta_1 \wedge \cdots \wedge \delta_k ::_{\mathrm{b}} \kappa}$$

$$\frac{}{\mathtt{o} ::_{\mathrm{u}} \mathtt{o}} \qquad \frac{\sigma ::_{\mathrm{b}} \kappa \qquad \delta ::_{\mathrm{u}} \kappa'}{\sigma \to \delta ::_{\mathrm{u}} \kappa \to \kappa'} \qquad \frac{\sigma ::_{\mathrm{u}} \kappa \qquad \delta ::_{\mathrm{u}} \kappa'}{\sigma \to \delta ::_{\mathrm{b}} \kappa \to \kappa'} \qquad \frac{\sigma ::_{\mathrm{b}} \kappa \qquad \delta ::_{\mathrm{b}} \kappa'}{\sigma \to \delta ::_{\mathrm{b}} \kappa \to \kappa'}$$

A type δ is called *balanced* if $\delta ::_{\mathrm{b}} \kappa$ for some κ, and called *unbalanced* if $\delta ::_{\mathrm{u}} \kappa$ for some κ. Intuitively, unbalanced types describe trees or closures that contain the end of a word (i.e., symbol $\mathtt{e}$). Intersection types that are neither balanced nor unbalanced are considered ill-formed, and excluded out. For example, the type $\mathtt{o} \to \mathtt{o} \to \mathtt{o}$ (as an intersection type) is ill-formed; since $\mathtt{o}$ is unbalanced, $\mathtt{o} \to \mathtt{o}$ must also be unbalanced according to the rules for arrow types, but it is actually balanced. In fact, no term can have the intersection type $\mathtt{o} \to \mathtt{o} \to \mathtt{o}$ in a word grammar. We write $\delta :: \kappa$ if $\delta ::_{\mathrm{b}} \kappa$ or $\delta ::_{\mathrm{u}} \kappa$.

We introduce a type-directed transformation relation $\Gamma \vdash_{\mathrm{fst}} t : \delta \Rightarrow u$ for terms, where Γ is a set of type bindings of the form $x : \delta$, called a *type environment*, t is a source term, and u is the image of the transformation. We write $\Gamma_1 \cup \Gamma_2$ for the union of Γ_1 and Γ_2; it is defined only if, whenever $x : \delta \in \Gamma_1 \cap \Gamma_2$, δ is balanced. In other words, unbalanced types are treated as linear types, whereas balanced ones as non-linear (or idempotent) types. Intuitively, if a type is unbalanced then an argument of the type is used linearly, i.e., used exactly once in the reduction; while if a type is balanced then an argument of the type may be copied and used in many places. We write $\mathbf{bal}(\Gamma)$ if δ is balanced for every $x : \delta \in \Gamma$.

The relation $\Gamma \vdash_{\mathrm{fst}} t : \delta \Rightarrow u$ is defined inductively by the rules in Figure 1. Note that, for a given ground closed term t, a term u such that $\vdash_{\mathrm{fst}} t : \mathtt{o} \Rightarrow u$ is not necessarily unique, while the original transformation return unique output (gathering all by using non-deterministic choice). However this does not matter since the result is semantically the same as shown in Section B.3. By dropping the transformation part "$\Rightarrow u$" from the rules in Figure 1, we obtain a standard form of intersection type system. Though the transformation is not deterministic, it is deterministic as a transformation of a type derivation tree of the (simplified) intersection type system to a term. The following is a standard fact on intersection type systems.

▶ **Lemma 59** (subject reduction/expansion). *For $t \longrightarrow t'$, $\Gamma \vdash_{\mathrm{fst}} t : \delta \Rightarrow u$ for some u iff $\Gamma \vdash_{\mathrm{fst}} t' : \delta \Rightarrow u$ for some u.*

For a word $a_1 \cdots a_n$, we define term $(a_1 \cdots a_n)^\star$ inductively by: $\epsilon^\star = \mathtt{e}$ and $(as)^\star = \mathtt{br}\, a\, s^\star$.

$$\frac{}{x : \delta \vdash_{\text{fst}} x : \delta \Rightarrow x_\delta} \text{(FTr-Var)} \qquad \frac{}{\vdash_{\text{fst}} \mathtt{e} : \mathtt{o} \Rightarrow \mathtt{e}} \text{(FTr-Const0)} \qquad \frac{\Sigma(a) = 1}{\vdash_{\text{fst}} a : \mathtt{o} \to \mathtt{o} \Rightarrow a} \text{(FTr-Const1)}$$

$$\frac{\Gamma_0 \vdash_{\text{fst}} s : \delta_1 \wedge \cdots \wedge \delta_k \to \delta \Rightarrow v \quad \Gamma_i \vdash_{\text{fst}} t : \delta_i \Rightarrow u_i \text{ and } \delta_i \neq \mathtt{o} \text{ (for each } i \in \{1, \ldots, k\})}{\Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_k \vdash_{\text{fst}} s\, t : \delta \Rightarrow v u_1 \cdots u_k} \text{(FTr-App1)}$$

$$\frac{\Gamma_0 \vdash_{\text{fst}} s : \mathtt{o} \to \delta \Rightarrow v \qquad \Gamma_1 \vdash_{\text{fst}} t : \mathtt{o} \Rightarrow u}{\Gamma_0 \cup \Gamma_1 \vdash_{\text{fst}} s\, t : \delta \Rightarrow \mathtt{br}\, v\, u} \text{(FTr-App2)}$$

$$\frac{\Gamma, x : \delta_1, \ldots, x : \delta_k \vdash_{\text{fst}} t : \delta \Rightarrow u \qquad x \notin dom(\Gamma) \quad \delta_i \neq \mathtt{o} \text{ for each } i \in \{1, \ldots, k\}}{\Gamma \vdash_{\text{fst}} \lambda x.t : \delta_1 \wedge \cdots \wedge \delta_k \to \delta \Rightarrow \lambda x_{\delta_1} \cdots \lambda x_{\delta_k}.u} \text{(FTr-Abs1)}$$

$$\frac{\Gamma, x : \mathtt{o} \vdash_{\text{fst}} t : \delta \Rightarrow u}{\Gamma \vdash_{\text{fst}} \lambda x.t : \mathtt{o} \to \delta \Rightarrow [\mathtt{e}/x_{\mathtt{o}}]u} \text{(FTr-Abs2)}$$

Figure 1 First transformation

▶ **Lemma 60** ([1, Lemma 10]). $\vdash_{\text{fst}} a_1(\cdots(a_n\, \mathtt{e})\cdots) : \mathtt{o} \Rightarrow (a_1 \cdots a_n)^\star$.

By Lemmas 59 and 60, for any closed ground term t, we have $\vdash_{\text{fst}} t : \mathtt{o} \Rightarrow u$ for some u.

We define $[\![\delta :: \kappa]\!]$ by:

$$[\![\delta :: \kappa]\!] = \mathtt{o} \qquad \text{(if } \mathtt{order}(\kappa) \leq 1)$$
$$[\![(\delta_1 \wedge \cdots \wedge \delta_k \to \delta) :: (\kappa_0 \to \kappa)]\!] = [\![\delta_1 :: \kappa_0]\!] \to \ldots \to [\![\delta_k :: \kappa_0]\!] \to [\![\delta :: \kappa]\!] \qquad \text{(if } \mathtt{order}(\kappa_0 \to \kappa) > 1)$$

▶ **Lemma 61.** *For an order-$(n+1)$ term t with $x_1 : \kappa_1, \ldots, x_m : \kappa_m \vdash_{\mathtt{ST}} t : \kappa$, if*

$$x_1 : \delta^1_1, \ldots, x_1 : \delta^1_{k_1}, \ldots, x_m : \delta^m_1, \ldots, x_m : \delta^m_{k_m} \vdash_{\text{fst}} t : \delta \Rightarrow u$$
$$(\wedge_{i \leq k_1} \delta^1_i \to \cdots \to \wedge_{i \leq k_m} \delta^m_i \to \delta) :: (\kappa_1 \to \cdots \to \kappa_m \to \kappa)$$

then u is an order-n term with

$$\begin{pmatrix} (x_1)_{\delta^1_1} : [\![\delta^1_1 :: \kappa_1]\!], & \ldots, & (x_1)_{\delta^1_{k_1}} : [\![\delta^1_{k_1} :: \kappa_1]\!], & \ldots, \\ (x_m)_{\delta^m_1} : [\![\delta^m_1 :: \kappa_m]\!], & \ldots, & (x_m)_{\delta^m_{k_m}} : [\![\delta^m_{k_m} :: \kappa_m]\!] & \end{pmatrix} \vdash_{\mathtt{ST}} u : [\![\delta :: \kappa]\!]$$

Proof. By straightforward induction on $x_1 : \delta^1_1, \ldots, x_1 : \delta^1_{k_1}, \ldots, x_m : \delta^m_1, \ldots, x_m : \delta^m_{k_m} \vdash_{\text{fst}} t : \delta \Rightarrow u$. ◻

B.2 Second Transformation

As explained above, the purpose of the second transformation is to remove extra $\mathtt{e}$. Inputs and outputs of the second transformation is $\lambda^{\to}$-terms of a $\mathtt{br}$-alphabet, and output terms generate $\mathtt{e}$-free $\mathtt{br}$-trees or $\mathtt{e}$.

For the second transformation, we use the following intersection types:

$$\xi ::= \mathtt{o}_\epsilon \mid \mathtt{o}_+ \mid \xi_1 \wedge \cdots \wedge \xi_k \to \xi$$

Intuitively, $\mathtt{o}_\epsilon$ describes trees consisting of only `br` and `e`, and $\mathtt{o}_+$ describes trees that have at least one non-`e` leaf. We again assume some total order $<$ on intersection types, and require that whenever we write $\xi_1 \wedge \cdots \wedge \xi_k$, $\xi_1 < \cdots < \xi_k$ holds. We define the refinement relation $\xi :: \kappa$ inductively by: (i) $\mathtt{o}_\epsilon :: \mathtt{o}$, (ii) $\mathtt{o}_+ :: \mathtt{o}$, and (iii) $(\xi_1 \wedge \cdots \wedge \xi_k \to \xi) :: (\kappa_1 \to \kappa_2)$ if $\xi :: \kappa_2$ and $\xi_i :: \kappa_1$ for every $i \in \{1, \dots, k\}$. We consider only types ξ such that $\xi :: \kappa$ for some κ. For example, we forbid an ill-formed type like $\mathtt{o}_+ \wedge (\mathtt{o}_+ \to \mathtt{o}_+) \to \mathtt{o}_+$.

We introduce a type-based transformation relation $\Xi \vdash_{\mathrm{snd}} t : \xi \Rightarrow u$, where Ξ is a type environment (i.e., a set of bindings of the form $x : \xi$), t is a source term, ξ is the type of t, and u is the result of transformation. The relation is defined inductively by the rules in Figure 2. As the first transformation, the results of the second transformation are not necessarily unique, but unique semantically as shown in Section B.3. Also, we have the following standard fact for this intersection type system.

▶ **Lemma 62** (subject reduction/expansion). *For $t \longrightarrow t'$, $\Xi \vdash_{\mathrm{snd}} t : \xi \Rightarrow u$ for some u iff $\Xi \vdash_{\mathrm{snd}} t' : \xi \Rightarrow u$ for some u.*

We define $[\![\xi]\!]$ by:

$$[\![\mathtt{o}_\epsilon]\!] = [\![\mathtt{o}_+]\!] = \mathtt{o} \qquad [\![\, \xi_1 \wedge \cdots \wedge \xi_k \to \xi]\!] = [\![\xi_1]\!] \to \cdots \to [\![\xi_k]\!] \to [\![\xi]\!]$$

▶ **Lemma 63.** *For an order-$(n+1)$ term t with $x_1 : \kappa_1, \dots, x_m : \kappa_m \vdash_{\mathtt{ST}} t : \kappa$, if*

$$x_1 : \xi^1_1, \dots, x_1 : \xi^1_{k_1}, \dots, x_m : \xi^m_1, \dots, x_m : \xi^m_{k_m} \vdash_{\mathrm{snd}} t : \xi \Rightarrow u$$
$$(\wedge_{i \le k_1} \xi^1_i \to \cdots \to \wedge_{i \le k_m} \xi^m_i \to \xi) :: (\kappa_1 \to \cdots \to \kappa_m \to \kappa)$$

then u is an order-n term with

$$(x_1)_{\xi^1_1} : [\![\xi^1_1]\!], \dots, (x_1)_{\xi^1_{k_1}} : [\![\xi^1_{k_1}]\!], \dots, (x_m)_{\xi^m_1} : [\![\xi^m_1]\!], \dots, (x_m)_{\xi^m_{k_m}} : [\![\xi^m_{k_m}]\!] \vdash_{\mathtt{ST}} u : [\![\xi]\!]$$

Proof. By straightforward induction on $x_1 : \xi^1_1, \dots, x_1 : \xi^1_{k_1}, \dots, x_m : \xi^m_1, \dots, x_m : \xi^m_{k_m} \vdash_{\mathrm{snd}} t : \xi \Rightarrow u$. ☐

B.3 First and Second Transformations Preserve Meaning

Here we show that the first and second transformations preserve the meaning of a term. Before that, let us review the corresponding theorems in [1]; here for a word w, we write $w{\uparrow}_{\mathtt{e}}$ for the word obtained by removing all the occurrences of `e` in w, and $\mathcal{L}{\uparrow}_{\mathtt{e}}$ for $\{w{\uparrow}_{\mathtt{e}} \mid w \in \mathcal{L}\}$.

▶ **Theorem 64** ([1, Theorem 7]). *If $\vdash_{\mathrm{ofst}} \mathcal{G} \Rightarrow \mathcal{G}'$, then $\mathcal{L}_{\mathtt{w}}(\mathcal{G}) = \mathcal{L}_{\mathtt{leaf}}(\mathcal{G}'){\uparrow}_{\mathtt{e}}$.*

▶ **Theorem 65** ([1, Theorem 9]). *If $\vdash_{\mathrm{osnd}} \mathcal{G} \Rightarrow \mathcal{G}'$, then $\mathcal{L}_{\mathtt{leaf}}(\mathcal{G}){\uparrow}_{\mathtt{e}} = \mathcal{L}^{\varepsilon}_{\mathtt{leaf}}(\mathcal{G}')$.*

▶ **Lemma 66.** *If $\vdash_{\mathrm{fst}} t : \mathtt{o} \Rightarrow u$, then $\mathbf{word}(\mathcal{T}(t)) = \boldsymbol{leaves}(\mathcal{T}(u)){\uparrow}_{\mathtt{e}}$.*

Proof. Figure 4 shows the definition of the original first transformation $\vdash_{\mathrm{ofst}}$ given in [1]. Here, meta variables U and V represent a set $\{u_1, \dots, u_k\}$ of terms, which is nothing but the non-deterministic choice $u_1 + \cdots + u_k$. Figure 3 are obtained from Figure 4 by removing non-deterministic choices; the change is just for (Tr1-App1), (Tr1-App2), (Tr1-Set), and (Tr1-Gram). We write $\vdash^{\mathrm{det}}_{\mathrm{ofst}}$ for the transformation by Figure 3. When applied to

$$\frac{}{x : \xi \vdash_{\mathrm{snd}} x : \xi \Rightarrow x_\xi}\ (\textsc{STr-Var}) \qquad \frac{}{\vdash_{\mathrm{snd}} \mathtt{e} : \mathtt{o}_\epsilon \Rightarrow \mathtt{e}}\ (\textsc{STr-Const0}) \qquad \frac{\Sigma(a) = 0 \qquad a \neq \mathtt{e}}{\vdash_{\mathrm{snd}} a : \mathtt{o}_+ \Rightarrow a}\ (\textsc{STr-Const1})$$

$$\frac{\Xi \vdash_{\mathrm{snd}} t_0 : \xi_0 \Rightarrow u_0 \qquad \Xi \vdash_{\mathrm{snd}} t_1 : \xi_1 \Rightarrow u_1 \qquad (u, \xi) = \begin{cases} (\mathtt{br}\, u_0\, u_1, \mathtt{o}_+) & \text{if } \xi_0 = \xi_1 = \mathtt{o}_+ \\ (u_i, \mathtt{o}_+) & \text{if } \xi_i = \mathtt{o}_+ \text{ and } \xi_{1-i} = \mathtt{o}_\epsilon \\ (\mathtt{e}, \mathtt{o}_\epsilon) & \text{if } \xi_0 = \xi_1 = \mathtt{o}_\epsilon \end{cases}}{\Xi \vdash_{\mathrm{snd}} \mathtt{br}\, t_0\, t_1 : \xi \Rightarrow u}\ (\textsc{STr-Const2})$$

$$\frac{\Xi \vdash_{\mathrm{snd}} s : \xi_1 \wedge \cdots \wedge \xi_k \to \xi \Rightarrow v \qquad \Xi \vdash_{\mathrm{snd}} t : \xi_i \Rightarrow u_i \ (\text{for each } i \in \{1, \ldots, k\})}{\Xi \vdash_{\mathrm{snd}} st : \xi \Rightarrow v u_1 \cdots u_k}\ (\textsc{STr-App})$$

$$\frac{\Xi, x : \xi_1, \ldots, x : \xi_k \vdash_{\mathrm{snd}} t : \xi \Rightarrow u}{\Xi \vdash_{\mathrm{snd}} \lambda x.t : \xi_1 \wedge \cdots \wedge \xi_k \to \xi \Rightarrow \lambda x_{\xi_1} \cdots \lambda x_{\xi_k}.u}\ (\textsc{STr-Abs})$$

Figure 2 Second transformation

recursion-free higher-order grammars, Figure 3 is essentially the same as Figure 1; the condition $(**)$ in (Tr1-Gram-Det) in Figure 3 ensures that $\vdash^{\mathrm{det}}_{\mathrm{ofst}}$ produces a deterministic higher-order grammar, and as in the case of $\vdash_{\mathrm{fst}}$, the fact that $\vdash^{\mathrm{det}}_{\mathrm{ofst}}$ produces at least one deterministic higher-order grammar is shown by subject expansion.

For a ground closed $\lambda^{\to}$-term t, let $\vdash_{\mathrm{fst}} t : \mathtt{o} \Rightarrow u$ and let $\mathcal{G}$ and $\mathcal{G}'$ be the deterministic higher-order grammars obtained from t and u, respectively. Then clearly we have $\vdash^{\mathrm{det}}_{\mathrm{ofst}} \mathcal{G} \Rightarrow \mathcal{G}'$; also let $\vdash_{\mathrm{ofst}} \mathcal{G} \Rightarrow \mathcal{G}''$. Now, since the set of rules in Figure 3 is just a subset of that in Figure 4, and since the rule (Tr1-Gram) in Figure 4 gathers all the derived rewriting rules, $\mathcal{G}'$ is a "syntactical determinization" of the choices in $\mathcal{G}''$; i.e., $\mathcal{G}' \Subset \mathcal{G}''$ where we write $\Subset$ for the least congruence relation such that $s \Subset s + t$ and $t \Subset s + t$ and we regard a grammar as a ground closed $\lambda^{\to,+}$-term with Y-combinator. Hence, we have $(\{\mathcal{T}(\mathcal{G}')\} =)\mathcal{L}(\mathcal{G}') \subseteq \mathcal{L}(\mathcal{G}'')$.

Now by Theorem 7 in [1], $\mathcal{L}_{\mathtt{w}}(\mathcal{G}) = \mathcal{L}_{\mathtt{leaf}}(\mathcal{G}'')\!\uparrow_{\mathtt{e}}$. Hence,

$$\{\mathbf{word}(\mathcal{T}(t))\} = \{\mathbf{word}(\mathcal{T}(\mathcal{G}))\} = \mathcal{L}_{\mathtt{w}}(\mathcal{G}) = \mathcal{L}_{\mathtt{leaf}}(\mathcal{G}'')\!\uparrow_{\mathtt{e}} \supseteq \{\mathbf{leaves}(\mathcal{T}(\mathcal{G}'))\}\!\uparrow_{\mathtt{e}} = \{\mathbf{leaves}(\mathcal{T}(u))\}\!\uparrow_{\mathtt{e}}$$

and therefore $\mathbf{word}(\mathcal{T}(t)) = \mathbf{leaves}(\mathcal{T}(u))\!\uparrow_{\mathtt{e}}$. ☐

▶ **Lemma 67.** *If $\vdash_{\mathrm{snd}} t : \mathtt{o} \Rightarrow u$, then*

$$\boldsymbol{leaves}(\mathcal{T}(t))\!\uparrow_{\mathtt{e}} = \begin{cases} \varepsilon & (\boldsymbol{leaves}(\mathcal{T}(u)) = \mathtt{e}) \\ \boldsymbol{leaves}(\mathcal{T}(u)) & (\boldsymbol{leaves}(\mathcal{T}(u)) \neq \mathtt{e}). \end{cases}$$

Proof. The proof is quite similar to that of Lemma 66: the points are (i) Figure 2 is a subset of the set of rules of the original second transformation (up to the embedding of λ-terms to higher-order grammars), (ii) a result of the second transformation is a deterministic λ-term, and hence its meaning seen as a language is a singleton, (iii) subset relation between singleton sets implies equality between their elements. ☐

$$\frac{}{x:\delta \vdash x:\delta \Rightarrow x_\delta}\quad(\textsc{Tr1-Var}) \qquad \frac{\delta :: \mathcal{N}(A)}{\vdash A:\delta \Rightarrow A_\delta}\quad(\textsc{Tr1-NT})$$

$$\frac{}{\vdash \mathtt{e}:\mathtt{o} \Rightarrow \mathtt{e}}\quad(\textsc{Tr1-Const0}) \qquad \frac{\Sigma(a)=1}{\vdash a:\mathtt{o}\to\mathtt{o} \Rightarrow a}\quad(\textsc{Tr1-Const1})$$

$$\frac{\begin{array}{c}\Gamma_0 \vdash s:\delta_1\wedge\cdots\wedge\delta_k\to\delta \Rightarrow v\\ \Gamma_i \vdash t:\delta_i \Rightarrow u_i \text{ and } \delta_i \neq \mathtt{o} \text{ (for each } i\in\{1,\dots,k\})\end{array}}{\Gamma_0\cup\Gamma_1\cup\cdots\cup\Gamma_k \vdash st:\delta \Rightarrow vu_1\cdots u_k}\quad(\textsc{Tr1-App1-Det})$$

$$\frac{\Gamma_0 \vdash s:\mathtt{o}\to\delta \Rightarrow v \qquad \Gamma_1 \vdash t:\mathtt{o} \Rightarrow u}{\Gamma_0\cup\Gamma_1 \vdash st:\delta \Rightarrow \mathtt{br}\,v\,u}\quad(\textsc{Tr1-App2-Det})$$

$$\frac{\begin{array}{c}\Gamma, x:\delta_1,\dots,x:\delta_k \vdash t:\delta \Rightarrow u \qquad x\notin \mathit{dom}(\Gamma)\\ \delta_i\neq\mathtt{o} \text{ for each } i\in\{1,\dots,k\}\end{array}}{\Gamma \vdash \lambda x.t:\delta_1\wedge\cdots\wedge\delta_k\to\delta \Rightarrow \lambda x_{\delta_1}\cdots\lambda x_{\delta_k}.u}\quad(\textsc{Tr1-Abs1})$$

$$\frac{\Gamma, x:\mathtt{o} \vdash t:\delta \Rightarrow u}{\Gamma \vdash \lambda x.t:\mathtt{o}\to\delta \Rightarrow [\mathtt{e}/x_\mathtt{o}]u}\quad(\textsc{Tr1-Abs2})$$

$$\frac{\emptyset \vdash \lambda x_1.\cdots\lambda x_k.t:\delta \Rightarrow \lambda x'_1.\cdots\lambda x'_\ell.u \qquad \delta :: \mathcal{N}(A)}{(A\,x_1\,\cdots\,x_k \to t) \Rightarrow (A_\delta\,x'_1\,\cdots\,x'_\ell \to u)}\quad(\textsc{Tr1-Rule})$$

$$\frac{\begin{array}{c}\Sigma' = \{\mathtt{br}\mapsto 2, \mathtt{e}\mapsto 0\}\cup\{a\mapsto 0 \mid \Sigma(a)=1\}\\ \mathcal{N}' = \{A_\delta : [\![\delta :: \kappa]\!] \mid \mathcal{N}(A)=\kappa\wedge\delta::\kappa\} \qquad \mathcal{R}'\subseteq\{r' \mid \exists r\in\mathcal{R}.r\Rightarrow r'\}\\ \forall(A\,x_1\,\cdots\,x_k\to t)\in\mathcal{R}.\forall\delta::\mathcal{N}(A).\exists! r'\in\mathcal{R}'.(A\,x_1\,\cdots\,x_k\to t)\Rightarrow r' \quad (**)\end{array}}{\vdash (\Sigma,\mathcal{N},\mathcal{R},S) \Rightarrow (\Sigma',\mathcal{N}',\mathcal{R}',S_\mathtt{o})}$$

(Tr1-Gram-Det)

Figure 3 First transformation for deterministic higher-order grammar

$$\frac{\mathbf{bal}(\Gamma)}{\Gamma, x:\delta \vdash x:\delta \Rightarrow x_\delta} \quad (\textsc{Tr1-Var}) \qquad \frac{\delta :: \mathcal{N}(A) \qquad \mathbf{bal}(\Gamma)}{\Gamma \vdash A:\delta \Rightarrow A_\delta} \quad (\textsc{Tr1-NT})$$

$$\frac{\mathbf{bal}(\Gamma)}{\Gamma \vdash \mathtt{e}:\mathtt{o} \Rightarrow \mathtt{e}} \quad (\textsc{Tr1-Const0}) \qquad \frac{\Sigma(a)=1 \qquad \mathbf{bal}(\Gamma)}{\Gamma \vdash a:\mathtt{o}\to\mathtt{o} \Rightarrow a} \quad (\textsc{Tr1-Const1})$$

$$\frac{\begin{array}{c}\Gamma_0 \vdash s:\delta_1 \wedge \cdots \wedge \delta_k \to \delta \Rightarrow v \\ \Gamma_i \vdash t:\delta_i \Rightarrow U_i \text{ and } \delta_i \neq \mathtt{o} \text{ (for each } i \in \{1,\dots,k\})\end{array}}{\Gamma_0 \cup \Gamma_1 \cup \cdots \cup \Gamma_k \vdash st:\delta \Rightarrow vU_1 \cdots U_k} \quad (\textsc{Tr1-App1})$$

$$\frac{\Gamma_0 \vdash s:\mathtt{o}\to\delta \Rightarrow V \qquad \Gamma_1 \vdash t:\mathtt{o} \Rightarrow U}{\Gamma_0 \cup \Gamma_1 \vdash st:\delta \Rightarrow \mathtt{br}\, V\, U} \quad (\textsc{Tr1-App2})$$

$$\frac{\Gamma \vdash t:\delta \Rightarrow u_i \text{ (for each } i \in \{1,\dots,k\}) \qquad k \geq 1}{\Gamma \vdash t:\delta \Rightarrow \{u_1,\dots,u_k\}} \quad (\textsc{Tr1-Set})$$

$$\frac{\begin{array}{c}\Gamma, x:\delta_1,\dots,x:\delta_k \vdash t:\delta \Rightarrow u \qquad x \notin dom(\Gamma) \\ \delta_i \neq \mathtt{o} \text{ for each } i \in \{1,\dots,k\}\end{array}}{\Gamma \vdash \lambda x.t:\delta_1 \wedge \cdots \wedge \delta_k \to \delta \Rightarrow \lambda x_{\delta_1} \cdots \lambda x_{\delta_k}.u} \quad (\textsc{Tr1-Abs1})$$

$$\frac{\Gamma, x:\mathtt{o} \vdash t:\delta \Rightarrow u}{\Gamma \vdash \lambda x.t:\mathtt{o}\to\delta \Rightarrow [\mathtt{e}/x_{\mathtt{o}}]u} \quad (\textsc{Tr1-Abs2})$$

$$\frac{\emptyset \vdash \lambda x_1.\cdots\lambda x_k.t:\delta \Rightarrow \lambda x'_1.\cdots\lambda x'_\ell.u \qquad \delta :: \mathcal{N}(A)}{(A\, x_1\, \cdots\, x_k \to t) \Rightarrow (A_\delta\, x'_1\, \cdots\, x'_\ell \to u)} \quad (\textsc{Tr1-Rule})$$

$$\frac{\begin{array}{c}\Sigma' = \{\mathtt{br} \mapsto 2, \mathtt{e} \mapsto 0\} \cup \{a \mapsto 0 \mid \Sigma(a)=1\} \\ \mathcal{N}' = \{A_\delta : [\![\delta :: \kappa]\!] \mid \mathcal{N}(A) = \kappa \wedge \delta :: \kappa\} \qquad \mathcal{R}' = \{r' \mid \exists r \in \mathcal{R}.r \Rightarrow r'\}\end{array}}{\vdash (\Sigma, \mathcal{N}, \mathcal{R}, S) \Rightarrow (\Sigma', \mathcal{N}', \mathcal{R}', S_{\mathtt{o}})} \quad (\textsc{Tr1-Gram})$$

Figure 4 Original first transformation for higher-order grammar

B.4 First and Second Transformations Preserve Linearity

Here we prove Lemma 13, by showing that the first and second transformations preserve linearity. In a nutshell, this consists of two parts: (i) simulation of reduction before a (first or second) transformation by reduction after that, and (ii) any linear normal form is transformed to a linear term. For $u \longrightarrow u'$, if u' is linear then so is u, and hence (i) and (ii) imply the preservation of linearity. However, for the first transformation, actually (i) does not hold due to the complicated rules (FTr-App2) and (FTr-Abs2); so we relax (i) to (i)': if $t \longrightarrow t'$ and t is transformed to u, then t' is transformed to some u' such that if u' is linear so is u. For the second transformation, (i) holds, and there is no difficulty. Here we explain the first transformation; for the second transformation the proof is analogous and easier. In the rest of this section, we call a call-by-name normal form a *normal form*.

▶ **Lemma 68.** *For $\Gamma \vdash_{\mathrm{fst}} t : \delta' \Rightarrow u$, x_δ occurs in u iff $(x : \delta) \in \Gamma$. If δ is unbalanced and $(x : \delta) \in \Gamma$, then x_δ occurs in u exactly once.*

Proof. By straightforward induction on $\Gamma \vdash_{\mathrm{fst}} t : \delta' \Rightarrow u$. ☐

▶ **Lemma 69** (substitution)**.** *Given $\Gamma_0, x : \delta_1, \ldots, x : \delta_k \vdash_{\mathrm{fst}} s : \delta \Rightarrow v$ where $x \notin dom(\Gamma_0)$ and $k \geq 0$, and given $\Gamma_i \vdash_{\mathrm{fst}} t : \delta_i \Rightarrow u_i$ for each $i \leq k$, if $\Gamma_0 \cup \cdots \cup \Gamma_k$ is well-defined, then we have*

$$\Gamma_0 \cup \cdots \cup \Gamma_k \vdash_{\mathrm{fst}} [t/x]s : \delta \Rightarrow [u_i/x_{\delta_i}]_{i \leq k} v\,.$$

Proof. By straightforward induction on $\Gamma_0, x : \delta_1, \ldots, x : \delta_k \vdash_{\mathrm{fst}} s : \delta \Rightarrow v$. ☐

For intersection type systems, also de-substitution is standard:

▶ **Lemma 70** (de-substitution)**.** *Given $\Gamma \vdash_{\mathrm{fst}} [t/x]s : \delta \Rightarrow v'$, there exist $k \geq 0$, $(\Gamma_i)_{0 \leq i \leq k}$, $(\delta_i)_{1 \leq i \leq k}$, $(u_i)_{1 \leq i \leq k}$, and v such that*

1. $\Gamma_0, x : \delta_1, \ldots, x : \delta_k \vdash_{\mathrm{fst}} s : \delta \Rightarrow v$
2. $\Gamma_i \vdash_{\mathrm{fst}} t : \delta_i \Rightarrow u_i \quad (1 \leq i \leq k)$
3. $\Gamma = \Gamma_0 \cup \cdots \cup \Gamma_k$
4. $v' = [u_i/x_{\delta_i}]_{i \leq k} v$.

Proof. By induction on s and case analysis on the last rule used for deriving $\Gamma \vdash_{\mathrm{fst}} [t/x]s{:}\delta \Rightarrow v'$. ☐

We call a type δ *inhabited* if there exist s and v such that $\vdash_{\mathrm{fst}} s : \delta \Rightarrow v$.

▶ **Lemma 71.** *If $z : \delta, x : \mathsf{o} \vdash_{\mathrm{fst}} t : \mathsf{o} \Rightarrow u$, $u \longrightarrow^* u'$, and δ is inhabited, then x_{o} occurs in u' exactly once.*

Proof. Since there exist s and v such that $\vdash_{\mathrm{fst}} s : \delta \Rightarrow v$, we have $x : \mathsf{o} \vdash_{\mathrm{fst}} [s/z]t : \mathsf{o} \Rightarrow [v/z_\delta]u$ by Lemma 69 above. Then we use the results in [1]: Lemmas 22, 24-(1), and the linearity of the type `oR`. ☐

The following lemma is a kind of subject reduction as well as a kind of left-to-right simulation.

▶ **Lemma 72.** *If $z : \delta \vdash_{\mathrm{fst}} t : \mathsf{o} \Rightarrow u$, $t \longrightarrow t'$, and δ is inhabited, then there exists u' such that $z : \delta \vdash_{\mathrm{fst}} t' : \mathsf{o} \Rightarrow u'$ and if u' is linear then so is u.*

Proof. The proof proceeds by induction on term t and by case analysis on the head of t. The case $t = as$ is clear by induction hypothesis. We consider the other case that $t = (\lambda x.t_0)t_1\, t_2 \dots, t_m$ $(m \geq 1)$. We further perform case analysis on the transformation rule used for the application $(\lambda x.t_0)t_1$: (FTr-App1) or (FTr-App2).

- Case of (FTr-App1): Let the simple type of $\lambda x.t_0$ be

  $$\kappa_1 \to \cdots \to \kappa_m \to \mathtt{o}$$

  and let κ_ℓ be the first type of order-0 among κ_i. Because of the assumption on simple types, $\kappa_i = \mathtt{o}$ for $i \geq \ell$. We have

  $$z : \delta \vdash_{\mathrm{fst}} (\lambda x.t_0)t_1\, t_2 \dots t_m : \mathtt{o} \Rightarrow \big((\lambda x_{\delta_1} \cdots x_{\delta_{k_1}}.u_0)\overrightarrow{u_i^1}^{i\leq k_1} \cdots \overrightarrow{u_i^{\ell-1}}^{i\leq k_{\ell-1}}\big) * u^\ell * \cdots * u^m$$
  $$(\lambda x.t_0)t_1\, t_2 \cdots t_m \longrightarrow ([t_1/x]t_0)t_2 \cdots t_m$$
  $$z : \delta \vdash_{\mathrm{fst}} ([t_1/x]t_0)t_2 \cdots t_m : \mathtt{o} \Rightarrow \big(([u_i^1/x_{\delta_{k_i}}]_{i\leq k_1} u_0)\overrightarrow{u_i^2}^{i\leq k_2} \cdots \overrightarrow{u_i^{\ell-1}}^{i\leq k_{\ell-1}}\big) * u^\ell * \cdots * u^m$$

  Then we have

  $$\begin{aligned} &\big((\lambda x_{\delta_1} \cdots x_{\delta_{k_1}}.u_0)\overrightarrow{u_i^1}^{i\leq k_1} \cdots \overrightarrow{u_i^{\ell-1}}^{i\leq k_{\ell-1}}\big) * u^\ell * \cdots * u^m \\ \longrightarrow^*\ &\big(([u_i^1/x_{\delta_{k_i}}]_{i\leq k_1} u_0)\overrightarrow{u_i^2}^{i\leq k_2} \cdots \overrightarrow{u_i^{\ell-1}}^{i\leq k_{\ell-1}}\big) * u^\ell * \cdots * u^m \end{aligned}$$

  where if the latter is linear then so is the former.
- Case of (FTr-App2): In this case the type of $(\lambda x.t_0)$ is of the form $\mathtt{o} \to \top \to \cdots \to \top \to \mathtt{o}$, and we have

  $$z : \delta \vdash_{\mathrm{fst}} (\lambda x.t_0)t_1\, t_2 \dots, t_m : \mathtt{o} \Rightarrow \mathtt{br}\, ([\mathtt{e}/x_{\mathtt{o}}]u_0)\, u_1$$
  $$(\lambda x.t_0)t_1\, t_2 \dots, t_m \longrightarrow ([t_1/x]t_0)t_2 \dots, t_m$$
  $$z : \delta \vdash_{\mathrm{fst}} ([t_1/x]t_0)t_2 \dots, t_m : \mathtt{o} \Rightarrow [u_1/x_{\mathtt{o}}]u_0$$

  By Lemma 68, x_o occurs in u_0 exactly once. From now we assume that $[u_1/x_{\mathtt{o}}]u_0$ is linear, and show that $\mathtt{br}\, ([\mathtt{e}/x_{\mathtt{o}}]u_0)\, u_1$ is also linear. We further perform case analysis:
  - Case where z_δ occurs in u_1: We have

    $$z : \delta \vdash_{\mathrm{fst}} t_1 : \mathtt{o} \Rightarrow u_1$$

    and the normal form of u_1 is of the form $E_1[z_\delta]$.

    The normal form of u_0 is of the form either $E_0[z_\delta]$ or $E_0[x_{\mathtt{o}}]$. If the former, then

    $$[u_1/x_{\mathtt{o}}]u_0 \longrightarrow^* [u_1/x_{\mathtt{o}}](E_0[z_\delta]) = ([u_1/x_{\mathtt{o}}]E_0)[z_\delta]$$

    where the last term is normal form since $[u_1/x_{\mathtt{o}}]E_0$ is an evaluation context. Since $[u_1/x_{\mathtt{o}}]u_0$ is linear, z_δ occurs in $([u_1/x_{\mathtt{o}}]E_0)[z_\delta]$ exactly once. Since z_δ occurs in u_1, $x_{\mathtt{o}}$ must not occur in E_0. Thus $x_{\mathtt{o}}$ does not occur in $E_0[z_\delta]$ and hence by Lemma 71, neither in u_0. This is a contradiction since $x_{\mathtt{o}}$ occurs in u_0 by Lemma 68. Thus, the normal form of u_0 is of the form $E_0[x_{\mathtt{o}}]$.

    Then, we have

    $$[u_1/x_{\mathtt{o}}]u_0 \longrightarrow^* [u_1/x_{\mathtt{o}}](E_0[x_{\mathtt{o}}]) = ([u_1/x_{\mathtt{o}}]E_0)[u_1] \longrightarrow^* ([u_1/x_{\mathtt{o}}]E_0)[E_1[z_\delta]]$$

    where note that $[u_1/x_{\mathtt{o}}]E_0$ is an evaluation context and the last term is a normal form. Since $[u_1/x_{\mathtt{o}}]u_0 \longrightarrow^* [u_1/x_{\mathtt{o}}](E_0[x_{\mathtt{o}}])$ is linear, z_δ does not occur in $[u_1/x_{\mathtt{o}}]E_0$

nor in E_1. Since z_δ occurs in u_1, $x_\mathtt{o}$ does not occur in E_0. Hence $E_0[x_\mathtt{o}]$ is closed; let $E_0[x_\mathtt{o}] \longrightarrow^* \pi$.

Now we have

$$\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1 \longrightarrow^* \mathtt{br}\,([\mathtt{e}/x_\mathtt{o}](E_0[x_\mathtt{o}]))\,u_1 = \mathtt{br}\,(E_0[\mathtt{e}])\,u_1 \longrightarrow^* \mathtt{br}\,\pi\,E_1[z_\delta]$$

where the last term is a normal form and contains exactly one z_δ. Thus, $\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1$ is linear.

- Case where z_δ does not occur in u_1: In this case we have

$$z : \delta, x : \mathtt{o} \vdash_{\mathrm{fst}} t_0 : \top \to \cdots \to \top \to \mathtt{o} \Rightarrow u_0$$
$$\vdash_{\mathrm{fst}} t_1 : \mathtt{o} \Rightarrow u_1$$

and hence u_1 is closed; let $u_1 \longrightarrow^* \pi$. The normal form of u_0 is of the form either $E_0[x_\mathtt{o}]$ or $E_0[z_\delta]$.

  * Case of $u_0 \longrightarrow^* E_0[x_\mathtt{o}]$: By Lemma 71, E_0 does not contain $x_\mathtt{o}$, and we have

$$[u_1/x_\mathtt{o}]u_0 \longrightarrow^* [u_1/x_\mathtt{o}](E_0[x_\mathtt{o}]) = E_0[u_1] \longrightarrow^* E_0[\pi]$$

  Here we put a label on π and then reduce further to a normal form N:

$$[u_1/x_\mathtt{o}]u_0 \longrightarrow^* E_0[\pi^\dagger] \longrightarrow^* N$$

  Let N' be the term obtained by replacing $\pi^\dagger$ in N with $\mathtt{e}$. Then we have

$$E_0[\mathtt{e}] \longrightarrow^* N'$$

  and N' is also a normal form. Further, by the linearity of $[u_1/x_\mathtt{o}]u_0$, z_δ occurs in N exactly once, and so does in N'. Now we have

$$\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1 \longrightarrow^* \mathtt{br}\,([\mathtt{e}/x_\mathtt{o}](E_0[x_\mathtt{o}]))\,u_1 = \mathtt{br}\,(E_0[\mathtt{e}])\,u_1 \longrightarrow^* \mathtt{br}\,N'\,u_1$$

  where the last term is a normal form and contains exactly one z_δ. Thus $\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1$ is linear.

  * Case of $u_0 \longrightarrow^* E_0[z_\delta]$: We have

$$[u_1/x_\mathtt{o}]u_0 \longrightarrow^* [u_1/x_\mathtt{o}](E_0[z_\delta]) = ([u_1/x_\mathtt{o}]E_0)[z_\delta]$$

  where the last term is a normal form, and by the linearity of $[u_1/x_\mathtt{o}]u_0$, z_δ does not occur in E_0. Now we have

$$\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1 \longrightarrow^* \mathtt{br}\,([\mathtt{e}/x_\mathtt{o}](E_0[z_\delta]))\,u_1 = \mathtt{br}\,(([\mathtt{e}/x_\mathtt{o}]E_0)[z_\delta])\,u_1$$

  where the last term is a normal form and contains exactly one z_δ. Thus $\mathtt{br}\,([\mathtt{e}/x_\mathtt{o}]u_0)\,u_1$ is linear.

☐

▶ **Lemma 73.** *For $\Gamma \vdash_{\mathrm{fst}} t : \delta \Rightarrow u$, if x does not occur in t, then $x \notin dom(\Gamma)$.*

Proof. By straightforward induction on term t. ☐

▶ **Lemma 74.** *For a linear normal form N, if $x : \delta_1, \ldots, x : \delta_k \vdash_{\mathrm{fst}} N : \mathtt{o} \Rightarrow u$, then $k = 1$ and u is a linear normal form.*

Proof. By straightforward induction on open normal form N and by Lemma 73. ☐

▶ **Lemma 75.** *If $x : \delta_1, \dots, x : \delta_k \vdash_{\text{fst}} t : \mathsf{o} \Rightarrow u$ and t is linear, then $k = 1$. Further if δ_1 is inhabited, then u is linear.*

Proof. Let N be the normal form of t. By Lemma 59, we have $x : \delta_1, \dots, x : \delta_k \vdash_{\text{fst}} N : \mathsf{o} \Rightarrow u'$ for some u'. By Lemma 74, $k = 1$.

When δ_1 is inhabited, for the reduction sequence

$$t = t_0 \longrightarrow t_1 \longrightarrow \dots \longrightarrow t_\ell = N$$

by Lemma 72, there exist $u_0, \dots, u_\ell$ such that

$$u_0 = u$$
$$x : \delta_1 \vdash_{\text{fst}} t_i : \mathsf{o} \Rightarrow u_i \quad (i = 0, \dots, \ell)$$
if u_{i+1} is linear then so is u_i $\quad (i = 0, \dots, \ell - 1)$

By Lemma 74 u_ℓ is linear, and hence $u_0 = u$ is linear. ☐

As a corollary of Lemmas 75 and 69, we have:

▶ **Lemma 76.** *For $z : \delta \vdash_{\text{fst}} t : \mathsf{o} \Rightarrow u$ and $y : \delta_1, \dots, y : \delta_k \vdash_{\text{fst}} s : \delta \Rightarrow v$, if $[s/z]t$ is linear, $k = 1$.*

A $\lambda^{\rightarrow}$-term t is *relevant* if, for any subterm $\lambda x.s$ in t, x occurs in s.

▶ **Lemma 77.** *For $\Gamma \vdash_{\text{fst}} t : \delta \Rightarrow u$, u is relevant, and if u is linear on some variable z, then z occurs in u exactly once.*

Proof. The former part is shown by straightforward induction on $\Gamma \vdash_{\text{fst}} t : \delta \Rightarrow u$, with using Lemma 68. The latter part is clear from the former part since, during reduction of a relevant term toward the normal form, the number of free occurrences of each variable does not decrease. ☐

▶ **Lemma 78.** *Given order-n $\lambda^{\rightarrow}$-contexts C, D, and order-n $\lambda^{\rightarrow}$-term t such that*

- *the constants in C, D, t are in a word alphabet*
- *$\{\mathcal{T}(C[D^{\ell_i}[t]]) \mid i \geq 0\}$ is infinite for any strictly increasing sequence $(\ell_i)_i$.*
- *C and D are linear*

there exist order-$(n-1)$ $\lambda^{\rightarrow}$-contexts G, H, order-$(n-1)$ $\lambda^{\rightarrow}$-term u, and some constant numbers c, $d \geq 1$ such that

- *the constants in G, H, u are in a* `br`*-alphabet*
- $\mathbf{word}(\mathcal{T}(C[D^{ci+d}[t]])) = \boldsymbol{leaves}(\mathcal{T}(G[H^i[u]]))\!\uparrow_{\mathsf{e}} \quad (i \geq 0)$
- *G and H are linear.*

Proof. Since $C[D^j[t]]$ is a ground closed term, we have $\vdash_{\text{fst}} C[D^j[t]] : \mathsf{o} \Rightarrow u_0$ for some u_0. By Lemmas 70, 75, 76, and the linearity, we have:

$$z : \delta_0 \vdash_{\text{fst}} C[z] : \mathsf{o} \Rightarrow v_0$$
$$z : \delta_j \vdash_{\text{fst}} D[z] : \delta_{j-1} \Rightarrow v_j \quad (j = 1, \dots, j)$$
$$\vdash_{\text{fst}} t : \delta_j \Rightarrow u'$$
$$u_0 = v_0[v_1[\dots v_j[u'/z_{\delta_j}] \dots /z_{\delta_1}]/z_{\delta_0}]$$

Now we use a "pumping" argument: Let κ be the simple type of t and let j_0 be the number of intersection types δ such that $\delta :: \kappa$. When $j = j_0 + 1$, among δ_j above, we have $\delta_{j_1} = \delta_{j_2}$ for some $1 \leq j_1 < j_2 \leq j$. Then we define

$$\begin{aligned} G &:= [[\,]/z_{\delta_{j_1}}](v_0[v_1[\dots v_{j_1-1}[v_{j_1}/z_{\delta_{j_1-1}}]\dots/z_{\delta_1}]/z_{\delta_0}]) \\ H &:= [[\,]/z_{\delta_{j_2}}](v_{j_1+1}[\dots v_{j_2-1}[v_{j_2}/z_{\delta_{j_2-1}}]\dots/z_{\delta_{j_1+1}}]) \\ u &:= [[\,]/z_{\delta_j}](v_{j_2+1}[\dots v_j[u'/z_{\delta_j}]\dots/z_{\delta_{j_2+1}}]) \end{aligned}$$

where $[[\,]/z_{\delta_j}]$ represents the replacement of a unique z_{δ_j} with $[\,]$, and this uniqueness follows from Lemma 77. Since $\delta_{j_1} = \delta_{j_2}$, for any $i \geq 0$, $G[H^i[u]]$ is well-typed.

Let $c = j_2 - j_1$ and $d = (j_0 + 1) - (j_2 - j_1)$. Let $i \geq 0$. By Lemma 69 we have

$$\vdash_{\text{fst}} C[D^{ci+d}[t]] : \mathtt{o} \Rightarrow G[H^i[u]]$$

and hence

$$\mathcal{T}(C[D^{ci+d}[t]]) = \mathbf{leaves}(\mathcal{T}(G[H^i[u]]))\!\uparrow_{\mathtt{e}}$$

by Lemma 66. Also, by Lemma 61, $G[H^i[u]]$ is order-$(n-1)$.

Finally, by Lemma 69 we have

$$z : \delta_{j_2} \vdash_{\text{fst}} C[D^{ci+j_1}[z]] : \mathtt{o} \Rightarrow G[H^i[z_{\delta_{j_2}}]]$$

and hence $G[H^i[z_{\delta_{j_2}}]]$ is linear by Lemma 75; thus, G and H are linear. ☐

For the second transformation, we have the following:

▶ **Lemma 79.** *Given order-n $\lambda^{\to}$-contexts C, D, and order-n $\lambda^{\to}$-term t such that*

- *the constants in C, D, t are in a* `br`*-alphabet*
- *$\{\boldsymbol{leaves}(\mathcal{T}(C[D^{\ell_i}[t]]))\!\uparrow_{\mathtt{e}} \mid i \geq 0\}$ is infinite for any strictly increasing sequence $(\ell_i)_i$.*
- *C and D are linear*

there exist order-n $\lambda^{\to}$-contexts G, H, order-n $\lambda^{\to}$-term u, and some constant numbers c, $d \geq 1$ such that

- *the constants in G, H, u are in a* `br`*-alphabet*
- *for $i \geq 0$,*

$$\boldsymbol{leaves}(\mathcal{T}(C[D^{ci+d}[t]]))\!\uparrow_{\mathtt{e}} = \begin{cases} \varepsilon & (\boldsymbol{leaves}(\mathcal{T}(G[H^i[u]])) = \mathtt{e}) \\ \boldsymbol{leaves}(\mathcal{T}(G[H^i[u]])) & (\boldsymbol{leaves}(\mathcal{T}(G[H^i[u]])) \neq \mathtt{e}) \end{cases}$$

- *G and H are linear.*

Lemma 13 is a corollary of Lemmas 78 and 79.

C Properties of Homeomorphic Embedding on Tree Functions

This section shows the well-definedness of $\preceq_\kappa$ and its reflexivity and transitivity. We write $=_{\beta\eta}$ for the $\beta\eta$-equivalence relation on the simply-typed λ-terms.

▶ **Lemma 80** (well-definedness)**.** *Let t, s, t', s' be terms of type κ. If $t =_{\beta\eta} t'$ and $s =_{\beta\eta} s'$, then $t \preceq_\kappa s$ if and only if $t' \preceq_\kappa s'$.*

Proof. Let $\kappa = \kappa_1 \to \cdots \to \kappa_k \to \mathsf{o}$. Then

$$\begin{aligned}
& t \preceq_\kappa s \\
& \Leftrightarrow t\, s_1 \cdots s_k \preceq_\mathsf{o} s\, s'_1 \cdots s'_k \text{ for every } s_1,\ldots,s_k,s'_1,\ldots,s'_k \text{ such that } s_i \preceq_{\kappa_i} s'_i \\
& \Leftrightarrow \mathcal{T}(t\, s_1 \cdots s_k) \preceq \mathcal{T}(s\, s'_1 \cdots s'_k) \text{ for every } s_1,\ldots,s_k,s'_1,\ldots,s'_k \text{ such that } s_i \preceq_{\kappa_i} s'_i \\
& \Leftrightarrow \mathcal{T}(t'\, s_1 \cdots s_k) \preceq \mathcal{T}(s'\, s'_1 \cdots s'_k) \text{ for every } s_1,\ldots,s_k,s'_1,\ldots,s'_k \text{ such that } s_i \preceq_{\kappa_i} s'_i \\
& \Leftrightarrow t'\, s_1 \cdots s_k \preceq_\mathsf{o} s'\, s'_1 \cdots s'_k \text{ for every } s_1,\ldots,s_k,s'_1,\ldots,s'_k \text{ such that } s_i \preceq_{\kappa_i} s'_i \\
& \Leftrightarrow t' \preceq_\kappa s'
\end{aligned}$$

◻

The following is the abstraction lemma of the logical relation.

▶ **Lemma 81.** *If* $\Gamma \vdash t : \kappa$*, then* $\Gamma \models t \preceq_\kappa t$*.*

Proof. This follows by induction on the derivation of $\Gamma \vdash t : \kappa$. Since the other cases are trivial, we show only the case where t is a λ-abstraction. In this case, we have

$$t = \lambda x_0 : \kappa_0 . t' \qquad \kappa = \kappa_0 \to \kappa' \qquad \Gamma, x_0 : \kappa_0 \vdash t' : \kappa'$$

Let $\Gamma = x_1 : \kappa_1, \ldots, x_k : \kappa_k$. We need to show that $[s_1/x_1,\ldots,s_k/x_k]t \preceq_\kappa [s'_1/x_1,\ldots,s'_k/x_k]t$ holds for every $s_1,\ldots,s_k,s'_1,\ldots,s'_k$ such that $s_i \preceq_{\kappa_i} s'_i$ for each $i \in \{1,\ldots,k\}$. i.e.,

$$([s_1/x_1,\ldots,s_k/x_k]t)s_0 \preceq_{\kappa'} ([s'_1/x_1,\ldots,s'_k/x_k]t)s'_0$$

holds for every $s_0,s_1,\ldots,s_k,s'_0,s'_1,\ldots,s'_k$ such that $s_i \preceq_{\kappa_i} s'_i$ for each $i \in \{0,1,\ldots,k\}$. By Lemma 80, it suffices to show

$$[s_0/x_0,s_1/x_1,\ldots,s_k/x_k]t' \preceq_{\kappa'} [s'_0/x_0,s'_1/x_1,\ldots,s'_k/x_k]t'.$$

This follows immediately from the induction hypothesis and $\Gamma, x_0 : \kappa_0 \vdash t' : \kappa'$. ◻

We are now ready to show that $\preceq_\kappa$ is reflexive and transitive.

▶ **Lemma 82.** $\preceq_\kappa$ *is reflexive and transitive.*

Proof. The reflexivity follows immediately as a special case of Lemma 81, where $\Gamma = \emptyset$. We show the transitivity by induction on κ. The base case follows immediately from the definition. For the induction step, suppose $t_1 \preceq_{\kappa_1 \to \kappa_2} t_2$ and $t_2 \preceq_{\kappa_1 \to \kappa_2} t_3$. Suppose also $s_1 \preceq_{\kappa_1} s_3$. We need to show $t_1 s_1 \preceq_{\kappa_2} t_3 s_3$. Since $s_1 \preceq_{\kappa_1} s_1$, we have $t_1 s_1 \preceq_{\kappa_2} t_2 s_1$ and $t_2 s_1 \preceq_{\kappa_2} t_3 s_3$. By the induction hypothesis, we obtain $t_1 s_1 \preceq_{\kappa_2} t_3 s_3$ as required. ◻

As a corollary, it follows that every tree function represented by the simply-typed λ-calculus is monotonic.

▶ **Corollary 83** (monotonicity)**.** *If* $\emptyset \vdash t : \kappa_1 \to \kappa_2$*, then* $t\, s_1 \preceq_{\kappa_2} t\, s'_1$ *for every* s_1, s'_1 *such that* $s_1 \preceq_{\kappa_1} s'_1$*.*