

Sequence Types for Hereditary Permutators

Pierre Vial 

Inria, Nantes

firstname.name@inria.fr

Abstract

The invertible terms in Scott's model \mathcal{D}_∞ are known as the hereditary permutators. Equivalently, they are terms which are invertible up to $\beta\eta$ -conversion with respect to the composition of the λ -terms. Finding a type-theoretic characterization to the set of hereditary permutators was problem # 20 of TLCA list of problems. In 2008, Tatsuta proved that this was not possible with an inductive type system. Building on previous work, we use an infinitary intersection type system based on sequences (*i.e.*, families of types indexed by integers) to characterize hereditary permutators with a unique type. This gives a positive answer to the problem in the coinductive case.

2012 ACM Subject Classification General and reference → General literature; General and reference

Keywords and phrases hereditary permutators, Böhm trees, intersection types, coinduction, rigidity, sequence types, non-idempotent intersection

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

The study of $\beta\eta$ -invertible terms goes back to Curry and Feys [7], who showed that the only regular *combinators* having an inverse have the form $\lambda x x_1 \dots x_n. x x_{\sigma(1)} \dots x_{\sigma(n)}$ with σ a permutation. Building on this work, Dezani [9] gave a characterization of the normal forms of all the invertible *normalizing* terms. This characterization was extended by Bergstra and Klop [3] for *any* term: $\beta\eta$ -invertible terms were proved to have Böhm trees of a certain form, generalizing that given by Curry and Feys and suggesting to name them *hereditary permutators*.

On another hand, intersection types systems were introduced by Coppo and Dezani [6, 12] around 1980 (see [16] for a survey). They were extensively used to characterize various sets of terms having common semantic properties (including head, weak, strong normalization) in different calculi. Yet, hereditary permutators resisted such a characterization, so that the problem of finding a type system assigning a unique type to all hereditary permutators (and only to them) was inscribed in TLCA list of open problem by Dezani in 2006 (Problem # 20). Two years later, Tatsuta [14] proved that the set HP of hereditary head permutators is not recursively enumerable. This entails that HP cannot be characterized in an *inductive* type system.

However, in [17], using a *coinductive intersection type system* named system **S**, we characterized the so-called set of *hereditary head normalizing (HHN)* terms which is also a set of terms having Böhm trees of certain form (without the constant \perp), whereas this set was also proved not to be recursively enumerable by Tatsuta [13]. As in the finitary case, infinite types bring simpler semantic proofs of well-known theorem, *e.g.*, system **S** helps proving that an asymptotic reduction strategy produces the infinitary normal form of a term when it exists. In this paper, we extend system **S** with a type constant characterizing the set of hereditary permutators and we thus give a positive answer to TLCA Problem # 20 in the coinductive case. This also proves that infinitary type systems may be used to characterize other sets of Böhm trees.

Before properly starting the article, a few words should be said on system **S** and infinitary typing: intersections are represented by families of types indexed by sets K of integers ≥ 2 . These indexes are called *tracks*. Thus, system **S** is close to *non-idempotent* intersection, introduced by Gardner [10] and de Carvalho [5], for which $A \wedge A \neq 0$. In the finite case, non-idempotency gives very simple proofs of normalization (see [4] for a survey). Tracks allow



© P. Vial;

licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

tracing occurrences of a given type in a derivation (*rigidity*) while ensuring syntax-direction, whereas having both is not possible when non-idempotent intersection is represented by lists or multisets. Rigidity is crucial in the infinitary case, because coinductive type grammars give birth to unsound derivations, *e.g.*, the unsolvable term $\Omega := (\lambda x.x x)(\lambda x.x x)$ becomes typable. However, rigidity allows defining a validity criterion, called approximability, which brings back semantic soundness. This is why system \mathbf{S} provides a good framework to characterize hereditary permutators.

Last, Tatsuta defines a type system with a family of type constants \mathbf{ptyp}_d (with $d \in \mathbb{N}$) such that $t : \mathbf{ptyp}_d$ iff t is a hereditary permutator on d levels. Then, a term is a hereditary permutator iff $t : \mathbf{ptyp}_d$ is derivable for all $d \in \mathbb{N}$. However, given a hereditary permutator t , there is no explicit relation between the different typings $t : \mathbf{ptyp}_d$ when d ranges over \mathbb{N} . We reuse this idea here, but the notion of approximability hinted at above allows formally expressing the typing derivations concluding with $t : \mathbf{ptyp}_d$ as *extensions* of those concluding with $t : \mathbf{ptyp}_{d_0}$ with $d_0 < d$. Actually, we define a type constant \mathbf{ptyp} , which can be assigned to hereditary permutators and to them only, which is the “supremum” of all \mathbf{ptyp}_{d_0} *i.e.*, such that a typing $t : \mathbf{ptyp}$ is an extension of typings $t : \mathbf{ptyp}_d$ for all $d \in \mathbb{N}$.

Structure of the paper: We conclude this introduction with some technical background on hereditary permutators. Sec. 1 recalls some basic definitions about Böhm trees and the infinite λ -calculus, but also on system \mathbf{S} and infinite types. In Sec. 2, we give a type-theoretic characterization of hereditary permutations in system \mathbf{S} . In Sec. 3, we introduce system $\mathbf{S}_{\mathbf{hp}}$, an extension of system \mathbf{S} , such that hereditary permutators have a unique type. The technical contributions of this paper are found mainly in Sec. 2 and 3.1.

Hereditary Permutators

Let \mathcal{V} be a set of term variables. For all $n \in \mathbb{N}$, \mathfrak{S}_n denotes the set of permutations of $\{1, \dots, n\}$, $\rightarrow_{\mathbf{h}}$ denotes head reduction and the reflexive-transitive closure of a reduction $\rightarrow_{\mathcal{R}}$ is denoted $\rightarrow_{\mathcal{R}}^*$. To define hereditary permutators, we first consider *headed* hereditary permutators, *i.e.*, hereditary permutators whose head variables have not been bound yet.

► Definition 1.

- For all $x \in \mathcal{V}$, the sets $\mathbf{HP}(x)$ of *x -headed Hereditary Permutators* (*x -HP*) ($x \in \mathcal{V}$) are defined by mutual coinduction:

$$\frac{h_1 \in \mathbf{HP}(x_1) \ \dots \ h_n \in \mathbf{HP}(x_n) \quad (n \geq 0, \sigma \in \mathfrak{S}_n, x_i \neq x, x_i \text{ pairwise distinct}) \quad \text{and } t \rightarrow_{\mathbf{h}}^* \lambda x_1 \dots x_n. x h_{\sigma(1)} \dots h_{\sigma(n)}}{h \in \mathbf{HP}(x)}$$

- A closed hereditary permutator, or simply, a **Hereditary Permutator (HP)** is a term of the form $h = \lambda x.h_0$ with $h_0 \in \mathbf{HP}(x)$ for some x .

A headed hereditary permutator is the head reduct of a hereditary permutator applied to a variable.

- **Theorem 2 ([3]).** A λ -term t is a hereditary permutator iff t is invertible modulo $\beta\eta$ -conversion for the operation \cdot defined by $u \cdot v = \lambda x.u(v x)$, whose neutral element is $I = \lambda x.x$.

An extensive presentation of hereditary permutators and their properties is given in Chapter 21 of [2].

1 Infinite terms and types

In this section, we present Böhm trees (chapter 10 of [2]) and the construction of one of the infinitary calculi introduced in [11]. See also [8, 1] for alternative presentations. We then present system **S**, an infinitary intersection type system with a validity criterion (approximability) discarding unsound coinductive derivations and using sequences to represent intersection. Some more details can also be found in [17].

General notations: The set of finite words on \mathbb{N} is denoted with \mathbb{N}^* , ε is the empty word, $a \cdot a'$ the concatenation of a and a' . The prefix order \preceq is defined on \mathbb{N}^* by $a \preceq a'$ if there is a_0 such that $a' = a \cdot a_0$, e.g., $2 \cdot 3 \preceq 2 \cdot 3 \cdot 0 \cdot 1$.

Intuitively, 0 is dedicated to the constructor λx , 1 is dedicated to the left-hand side of applications and all the $k \geq 2$ to the possibly multiple typings of the arguments of applications. This also explains why 0 and 1 will have a particular status in the definitions to come. For instance, the **applicative depth** $\text{ad}(a)$ of $a \in \mathbb{N}^*$ is the number of nestings inside arguments, i.e., $\text{ad}(a)$ is defined inductively by $\text{ad}(\varepsilon) = 0$, $\text{ad}(a \cdot k) = \text{ad}(a)$ if $k = 0$ or $k = 1$ and $\text{ad}(a \cdot k) = \text{ad}(a) + 1$ if $k \geq 2$. The **collapse** is defined on \mathbb{N} by $\bar{k} = \min(k, 2)$ and on \mathbb{N}^* inductively by $\bar{\varepsilon} = \varepsilon$, $\bar{a \cdot k} = \bar{a} \cdot \bar{k}$, e.g., $\bar{7} = 2$, $\bar{1} = 1$ and $\bar{2 \cdot 3 \cdot 0 \cdot 1} = 2 \cdot 2 \cdot 0 \cdot 1$. These notions are straightforwardly extended to words of infinite length, e.g., 2^ω , which is the infinite repetition of 2.

1.1 Infinite Lambda Terms

The set Λ^∞ of infinitary λ -terms is coinductively defined by:

$$t, u := x \in \mathcal{V} \parallel (\lambda x.t) \parallel (tu)$$

When there is no ambiguity, we usually just write $\lambda x.t$ and $tu_1 \dots u_n$ instead of $(\lambda x.t)$ and $(\dots (tu_1) \dots u_n)$. If t is an infinitary term, then $\text{supp}(t)$, the **support** of t (the set of positions in t) is defined in the usual way, i.e., coinductively, $\text{supp}(x) = \{\varepsilon\}$, $\text{supp}(\lambda x.t) = \{\varepsilon\} \cup 0 \cdot \text{supp}(t)$ and $\text{supp}(tu) = \{\varepsilon\} \cup 1 \cdot \text{supp}(t) \cup 2 \cdot \{\varepsilon\}$. If $\bar{a} \in \text{supp}(t)$, the subterm of t at position \bar{a} is denoted $t|_{\bar{a}}$, e.g., if $t = \lambda x.(xy)z$ and $a = 0 \cdot 1$ (resp. $a = 0 \cdot 4$), then $t|_a = xy$ (resp. $t|_a = z$).

► **Definition 3 (001-Terms).** Let $t \in \Lambda^\infty$. Then t is a **001-term**, if, for all infinite branches γ in $\text{supp}(t)$, $\text{ad}(\gamma) = \infty$.

Once again, the vocable “001-term” comes from [11]. For instance, the 001-term f^ω is formally defined as the tree such that $\text{supp}(f^\omega) = \{2^n \mid n \in \mathbb{N}\} \cup \{2^n \cdot 1 \mid n \in \mathbb{N}\}$, $f^\omega(2^n) = @$ and $f^\omega(2^n \cdot 1) = f$ for all $n \in \mathbb{N}$. Its unique infinite branch is 2^ω (since all the finite prefixes of 2^ω are in $\text{supp}(f^\omega)$), which satisfies $\text{ad}(2^\omega) = \infty$. In contrast, the infinite term t defined by $t = tx$, so that $t = (((\dots)x)x)x$, is *not* a 001-term: indeed, $\text{supp}(t) = \{1^n \mid n \in \mathbb{N}\} \cup \{1^n \cdot 2 \mid n \in \mathbb{N}\}$, so $\text{supp}(t)$ has the infinite branch 1^ω (this indicates a leftward infinite branch), which satisfies $\text{ad}(1^\omega) = 0$ since 2 does not occur in 1^ω .

1.2 The computation of Böhm trees

The notation $t[u/x]$ denotes the term obtained from t by the capture-free substitution of the occurrences of x with u ([11] gives a formal definition in the infinitary calculus). The β -reduction \rightarrow_β is obtained by the contextual closure of $(\lambda x.t)u \rightarrow_\beta t[u/x]$ and $t \xrightarrow[\beta]{b} t'$ denotes the reduction of a redex at position b in t , e.g., $\lambda y.((\lambda x.x)u)v \xrightarrow[\beta]{0 \cdot 1} \lambda y.u.v$. A **001-Normal**

125 **Form (001-NF)** is a 001-term that does not contain a redex. A 001-term is **solvable** if
 126 $t \rightarrow_h^* \lambda x_1 \dots x_p. x t_1 \dots t_q$, which is a head normal form (HNF) of **arity** p .

127 ► **Definition 4** (Böhm tree of a term). *Let t be a 001-term.*

128 *The Böhm tree $\text{BT}(t)$ of t is coinductively defined by:*

129 ■ $\text{BT}(t) = \lambda x_1 \dots x_p. x \text{BT}(t_1) \dots \text{BT}(t_q)$ if $t \rightarrow_h^* \lambda x_1 \dots x_p. x t_1 \dots t_q$.

130 ■ $\text{BT}(t) = \perp$ if t is unsolvable.

131 For instance, $\text{BT}(\Omega) = \perp$ where $\Omega = (\lambda x. x x)(\lambda x. x x)$ and $\text{BT}(t) = t$ if t a 001-normal
 132 form. Definition 1 can be read as the specification as a set of terms whose Böhm trees have
 133 a particular form. Intuitively, the computation of Böhm tree is done by a possibly infinite
 134 series of head reduction at deeper and deeper levels. This corresponds to an asymptotic
 135 reduction strategy known as *hereditary head reduction*.

136 Some reduction paths are of infinite length but asymptotically produce a term.

137 ► **Definition 5** (Productive reduction paths). *Let $t = t_0 \xrightarrow{\beta}_{b_0} t_1 \xrightarrow{\beta}_{b_1} t_2 \dots t_n \xrightarrow{\beta}_{b_n} t_{n+1} \dots$ be a*
 138 *reduction path of length $\ell \leq \omega$.*

139 *Then, this reduction path is said to be **productive** if either it is of finite length ($\ell \in \mathbb{N}$), or*
 140 *$\ell = \infty$ and $\text{ad}(b_n)$ tends to infinity.*

141 A productive reduction path is called a *strongly converging reduction sequence* in [11], in
 142 which numerous examples are found. When $\text{BT}(t)$ does not contain \perp , the hereditary head
 143 reduction strategy on a term t gives a particular case of productive path.

144 ► **Lemma 6** (Limits of productive paths). *Let $t = t_0 \xrightarrow{\beta}_{b_0} t_1 \xrightarrow{\beta}_{b_1} t_2 \dots t_n \xrightarrow{\beta}_{b_n} t_{n+1} \dots$ be a*
 145 *productive reduction path of infinite length.*

146 *Then, there is a 001-term t' such that, for all $d \geq 0$, there is $N \in \mathbb{N}$ such that, for all $n \geq N$,*
 147 *$\text{supp}(t_n) \cap \{b \in \{0, 1, 2\}^* \mid \text{ad}(b) \leq d\} = \text{supp}(t') \cap \{b \in \{0, 1, 2\}^* \mid \text{ad}(b) \leq d\}$.*

148

149 The term t' in the statement of Lemma 6 is called the **limit** of the productive path.
 150 Intuitively, when t' is the limit of $(t_n)_{n \geq 0}$, then t' induces the same tree as t_n at fixed
 151 applicative depth after sufficiently many reduction steps. We then write $t \rightarrow_\beta^\infty t'$ if $t \rightarrow_\beta^* t'$
 152 or t is the limit of a productive path starting at t . For instance, if $\Delta_f = \lambda x. f(x x)$,
 153 $Y_f = \Delta_f \Delta_f$ (with $f \in \mathcal{V}$), then $Y_f \xrightarrow{\beta}_\varepsilon f(Y_f)$, which gives the productive path $Y_f \xrightarrow{\beta}_\varepsilon$
 154 $f(Y_f) \xrightarrow{\beta}_2 f^n(Y_f) \xrightarrow{\beta}_{2^n} f^{n+1}(Y_f) \dots$ since $\text{ad}(2^n) \rightarrow \infty$. The limit of this path—which
 155 implements hereditary head reduction on Y_f —is f^ω , i.e., $Y_f \rightarrow_\beta^\infty f^\omega$ and also $\text{BT}(Y_f) = f^\omega$.

156 A 001-term t is said to be **infinitary weakly normalizing** (WN_∞) if there is a 001-NF
 157 t' such that $t \rightarrow_\beta^\infty t'$. It turns out that t is WN_∞ iff its Böhm tree does not contain \perp . The
 158 result is proved in [11] in a syntactical way, but we give a semantic proof of this fact in [17].

159 1.3 System S (sequential intersection)

160 A **sequence** of elements of a set X is a family $(x_k)_{k \in K}$ with $K \subseteq \mathbb{N} \setminus \{0, 1\}$. In this case, if
 161 $k_0 \in K$, x_{k_0} is the element of $(x_k)_{k \in K}$ **on track** k . We often write $(k \cdot x_k)_{k \in K}$ for $(x_k)_{k \in K}$,
 162 which, for instance, allows us to denote by $(2 \cdot a, 4 \cdot b, 5 \cdot a)$ or $(4 \cdot b, 2 \cdot a, 5 \cdot a)$ the sequence
 163 $(x_k)_{k \in K}$ with $K = \{2, 4, 5\}$, $x_2 = x_5 = a$ and $x_4 = b$. In this sequence, the element on track
 164 4 is b . Sequences come along with a **disjoint union** operator, denoted \uplus . Let $(x_k)_{k \in K}$ and
 165 $(x'_k)_{k \in K'}$ be two sequences:

166 ■ If $K \cap K' = \emptyset$, then $(x_k)_{k \in K} \uplus (x'_k)_{k \in K'}$ is $(x'')_{k \in K''}$ with $K'' = K \cup K'$ and $x''_k = x_k$
 167 when $k \in K$ and $x''_k = x'_k$ when $k \in K'$.

168 ■ If $K \cap K' \neq \emptyset$, $(x_k)_{k \in K} \uplus (x'_k)_{k \in K'}$ is not defined.

169 The operator \uplus is partial, associative and commutative.

170 Let \mathcal{O} a set of type atoms o . The set of **S**-types is coinductively defined by:

$$171 \quad T, S_k ::= o \in \mathcal{O} \parallel (S_k)_{k \in K} \rightarrow T$$

172 A sequence of types $(S_k)_{k \in K}$ is called a **sequence type** and it represents an intersection
 173 of types. The types of system **S** collapse on usual non-idempotent intersection types built
 174 on multisets [4], *e.g.*, the **S**-types $(2 \cdot o, 3 \cdot o', 4 \cdot o) \rightarrow o$ and $(2 \cdot o', 8 \cdot o, 9 \cdot o)$ collapse on
 175 $[o, o, o'] \rightarrow o$. The system is *strict* [12, 15, 16] since intersections occur only on left-hand sides
 176 of arrows. The **domain** and **codomain** of an arrow type are defined by $\text{dom}((S_k)_{k \in K} \rightarrow T) = (S_k)_{k \in K}$ and $\text{codom}((S_k)_{k \in K} \rightarrow T) = T$. The **arity** of a type is coinductively
 177 defined by $\text{ar}(o) = 0$ and $\text{ar}((S_k)_{k \in K} \rightarrow T) = \text{ar}(T) + 1$. For instance, if T is defined by
 178 $T = (2 \cdot o) \rightarrow T = (2 \cdot o) \rightarrow (2 \cdot o) \rightarrow \dots$, then $\text{ar}(T) = \infty$. A **001-type** is a **S**-type T such
 179 that, for all $c \in \text{supp}(T)$, $\text{ar}(T|_c) < \infty$, where $T|_c$ is the subtree rooted at c in T ($T|_c$ is a
 180 type). The **target type** $\text{targ}(T)$ of a 001-type S is *inductively* defined by $\text{targ}(o) = o$ and
 181 $\text{targ}((S_k)_{k \in K} \rightarrow T) = \text{targ}(T)$.

182 The **support** of a type or a sequence type U is coinductively defined by $\text{supp}(o) = \{\varepsilon\}$,
 183 $\text{supp}((S_k)_{k \in K}) = \cup_{k \in K} k \cdot \text{supp}(S_k)$ and $\text{supp}((S_k)_{k \in K} \rightarrow T) = \{\varepsilon\} \cup \text{supp}((S_k)_{k \in K}) \cup 1 \cdot$
 184 $\text{supp}(T)$. Since $1 \notin K$ by convention, this definition is correct. If $c \in \text{supp}(U)$, then $U|_c$
 185 denotes the type or sequence type rooted at position c in U ($U|_c$ is a type when U is a type or
 186 $c \neq \varepsilon$). For instance, if $U = (2 \cdot o) \rightarrow (2 \cdot o, 3 \cdot o) \rightarrow o$ and $c = 1$, then $U|_c = (2 \cdot o, 3 \cdot o) \rightarrow o$.

187 An **S**-context C (or D) is a total function from \mathcal{V} to the set of **S**-types. The operator \uplus is
 188 extended point-wise. An **S**-judgment is a triple $C \vdash t : T$, where C , t and T are respectively
 189 an **S**-context, a 001-term and an **S**-type. A **sequence judgment** is a sequence of judgments
 190 $(C_k \vdash t : T_k)_{k \in K}$ with $K \subseteq \mathbb{N} \setminus \{0, 1\}$. For instance, if $8 \in K$, the judgment $C_8 \vdash t : T_8$ is
 191 specified on track 8. The set of **S**-derivations is defined coinductively by:

$$192 \quad \frac{}{x : (k \cdot T) \vdash x : T} \text{ax} \quad \frac{C; x : (S_k)_{k \in K} \vdash t : T}{C \vdash \lambda x. t : (S_k)_{k \in K} \rightarrow T} \text{abs}$$

$$193 \quad \frac{C \vdash t : (S_k)_{k \in K} \rightarrow T \quad (D_k \vdash u : S_k)_{k \in K}}{C \uplus (\uplus_{k \in K} D_k) \vdash t u : T} \text{app}$$

194 We call **S**₀, the restriction of system **S** to finite types and contexts, but allowing infinite
 195 terms. The derivation P_{ex} below is in **S**₀.

196 Let P be a **S**-derivation typing a term t . The **support** of P is the set of positions of
 197 judgments inside P defined in the expected way: 0 to visit the premise of an **abs**-rule, 1 to
 198 visit the left-hand side of an **app**-rule and $k \geq 2$ to visit an argument judgment on track k on
 199 the right-hand side of the **app**-rule. Thus, if $a \in \text{supp}(P)$, $P(a)$, which denotes the judgment
 200 at position a in P , types the subterm $t|_a$. We denote the type and the context of $P(a)$ by
 201 $\mathbf{T}^P(a)$ and $\mathbf{C}^P(a)$, so that $P(a) = \mathbf{C}^P(a) \vdash t|_a : \mathbf{T}^P(a)$. Moreover:

202 ■ If $a \in \text{supp}(P)$ and $c \in \text{supp}(\mathbf{T}^P(a))$, then the pair (a, c) is a **right biposition** of P and
 203 $P(a, c)$ denotes $\mathbf{T}^P(a)(c)$.

204 ■ If $a \in \text{supp}(P)$, $x \in \mathcal{V}$ and $k \cdot c \in \text{supp}(\mathbf{C}^P(a)(x))$, then the triple $(a, x, k \cdot c)$ is a **left**
 205 **biposition** in P and $P(a, x, c)$ denotes $\mathbf{C}^P(a)(x)(c)$.

206 The set of bipositions of P is called the **bisupport** of P and is denoted by $\text{bisupp}(P)$. An
 207 **S**-derivation P is **finite**, *i.e.*, is a derivation of system **S**₀, iff $\text{bisupp}(P)$ is a finite set. If
 208 $a \in \text{supp}(P)$ and $t(a) = x$, $P(a)$ is an **ax**-rule and $\mathbf{C}^P(a) = x : (k \cdot \mathbf{T}^P(a))$.

209 ► **Example 7.** In the derivation, [7] indicates that the judgment $f : 2 \cdot () \rightarrow o \vdash f^\omega : o$ is on
 210 track 7.

$$\begin{array}{c}
212 \quad P_{\text{ex}} = \frac{\frac{\frac{}{x : (2 \cdot (7 \cdot o) \rightarrow o') \vdash x : (7 \cdot o) \rightarrow o'}{\text{ax}}} \quad \frac{\frac{\frac{}{f : (2 \cdot () \rightarrow o) \vdash f : () \rightarrow o} \text{ax}}{f : (2 \cdot () \rightarrow o) \vdash f^\omega : o [7]} \text{app}}{\text{app}}}{\frac{x : (2 \cdot (7 \cdot o) \rightarrow o'), f : (2 \cdot () \rightarrow o) \vdash x f^\omega : o}{\text{abs}}} \text{abs}
\end{array}$$

213 where $\text{supp}(P_{\text{ex}}) = \{\varepsilon, 0, 0 \cdot 1, 0 \cdot 7, 0 \cdot 7 \cdot 1\}$. Remark how f^ω is typed in the derivation using
 214 the type $() \rightarrow o$. $P_{\text{ex}}(0 \cdot 7 \cdot 1) = f : (2 \cdot () \rightarrow o) \vdash f : () \rightarrow o$, we have $\mathsf{T}^{P_{\text{ex}}}(0 \cdot 1) = (7 \cdot o) \rightarrow o$.
 215 Since $\text{supp}(\mathsf{T}^{P_{\text{ex}}}) = \{\varepsilon, 7, 1\}$ and $\mathsf{T}^{P_{\text{ex}}}(\varepsilon) = \Rightarrow$, $\mathsf{T}^{P_{\text{ex}}}(7) = o$ and $\mathsf{T}^{P_{\text{ex}}}(1) = o'$, we have
 216 $(0 \cdot 1, \varepsilon), (0 \cdot 1, 7), (0 \cdot 1, 1) \in \text{bisupp}(P_{\text{ex}})$ and $P_{\text{ex}}(0 \cdot 1, \varepsilon) = \Rightarrow$, $P_{\text{ex}}(0 \cdot 1, 7) = o$, $P_{\text{ex}}(0 \cdot 1, 1) =$
 217 o' . Likewise, $\mathsf{T}^{P_{\text{ex}}}(0 \cdot 7) = o$ and $\mathsf{C}^{P_{\text{ex}}}(0) = x : (2 \cdot (7 \cdot o) \rightarrow o), f : (2 \cdot () \rightarrow o)$, so that, *e.g.*,
 218 $(0, x, 2 \cdot 7), (0, f, 2) \in \text{bisupp}(P_{\text{ex}})$, $P_{\text{ex}}(0, x, 2 \cdot 7) = o$, $P_{\text{ex}}(0, f, 2) = \Rightarrow$.

219 A derivation P is quantitative when the context is computable from the axiom rules:

220 ► **Definition 8** (Quantitative derivation). *Let P be a \mathbf{S} -derivation. Then P is **quantitative***
 221 *if, for all $a \in \text{supp}(P)$, $x \in \mathcal{V}$, $k \in \mathbb{N} \setminus \{0, 1\}$ such that $(a, x, k) \in \text{bisupp}(P)$, there is $a_0 \succeq a$*
 222 *such that $P(a_0) = x : (k \cdot S) \vdash x : S$.*

223 Observe that, in a quantitative derivation $P \triangleright C \vdash t : T$, if $C(x) = (k \cdot S)$ *i.e.*, x is
 224 assigned a singleton sequence type, then there is exactly one **ax**-rule typing x (we use this in
 225 the proof of Claim 21).

An example of non-quantitative derivations is given by the family $(P_k)_{k \geq 2}$ defined by:

$$P_k = \frac{\frac{\frac{}{f : k \cdot ((2 \cdot o) \rightarrow o) \vdash f : (2 \cdot o) \rightarrow o} \text{ax}}{P_{k+1} \triangleright f : ((\ell \cdot (2 \cdot o) \rightarrow o))_{\ell \geq k+1}, x : (2 \cdot o) \vdash f^\omega : o [2]} \text{app}}{f : ((\ell \cdot (2 \cdot o) \rightarrow o))_{\ell \geq k}, x : (2 \cdot o) \vdash f^\omega : o} \text{app}$$

226 The P_k type f^ω with o but they assign a non-empty sequence type to $x \notin \text{fv}(f^\omega)$: this is
 227 why there are not quantitative. Indeed, $(\varepsilon, x, 2) \in \text{bisupp}(P_k)$, but there is no $a_0 \in \text{supp}(P_k)$
 228 such that $P_k(a_0) = x : (2 \cdot o) \vdash x : o$. Remark how the infinite branch of f^ω is used to assign
 229 a type to x whereas it does not occur in the subject. In contrast, if t is a finite λ -term, every
 230 derivation typing t is quantitative. However, Lemma 16 below states that quantitativeness is a
 231 sufficient condition for soundness for normal forms.

232 1.4 Approximability

233 ► **Definition 9** (Approximation). *Let P_0 and P be two \mathbf{S} -derivations typing a same term t .*
 234 *Then P_0 is an **approximation** of P if $\text{bisupp}(P_0) \subseteq \text{bisupp}(P)$ and, for all $\mathbf{p} \in \text{bisupp}(P_0)$,*
 235 *$P_0(\mathbf{p}) = P(\mathbf{p})$. When this holds, we write $P_0 \leq P$.*

Intuitively, $P_0 \leq P$ if the derivation P_0 can be obtained from the derivation P by erasing
 some symbols inside P . For instance, let:

$$P_{\text{ex}}^0 = \frac{\frac{\frac{}{x : (2 \cdot () \rightarrow o') \vdash x : () \rightarrow o'}{\text{ax}}}{\frac{x : (2 \cdot () \rightarrow o') \vdash x f^\omega : o}{\text{app}}} \text{app}}{x : (2 \cdot () \rightarrow o') \vdash \lambda f. x f^\omega : () \rightarrow o'} \text{abs}$$

236 Then $P_{\text{ex}}^0 \leq P_{\text{ex}}$, since P_{ex}^0 has been obtained from P_{ex} by erasing all typing information
 237 on f^ω . Indeed, we check that $\text{supp}(P_{\text{ex}}^0) \subseteq \text{supp}(P_{\text{ex}})$, $\text{bisupp}(P_{\text{ex}}^0) \subseteq \text{bisupp}(P_{\text{ex}})$ and
 238 $P_{\text{ex}}^0(\mathbf{p}) = P_{\text{ex}}(\mathbf{p})$ for all $\mathbf{p} \in \text{bisupp}(P_{\text{ex}}^0)$.

The relation \leq is an order. There are \mathbf{S} -derivations P that do not have finite approximations, *e.g.*, any derivation typing Ω (see [17] for an example), but these derivations are *unsound*: they do not ensure any form of finitary or infinitary normalization. In contrast, a *finite* derivation is sound.

To retrieve validity, we must specify that infinitary derivations should be obtained as asymptotic extensions of *finite* derivations:

► **Definition 10** (Approximability). *Let P be a \mathbf{S} -derivation. Then P is **approximable** if P is the supremum of its finite approximations *i.e.*, if, for all finite sets $B \subseteq \mathbf{bisupp}(P)$, there is a finite derivation P_0 such that $P_0 \leq P$ and $B \subseteq \mathbf{bisupp}(P_0)$.*

A term that is in the conclusion of an approximable derivation is said to be **approximably typable**. Quantitativity is of course a necessary condition for approximability, and types of infinite arity are unsound:

► **Lemma 11.** *If P is approximable, then P is quantitative and contains only 001-types.*

1.5 Soundness and completeness for system \mathbf{S}

The main characterization theorem of system \mathbf{S} states the equivalence between infinitary weak normalization and typability: more precisely, t is \mathbf{WN}_∞ iff there is an unforgetful and approximable \mathbf{S} -derivation typing P . This characterization is proved by the propositions below, that we will also use in this article. One recognizes usual properties that are expected from an intersection type system (subject reduction, expansion, typing of the normal forms), except that they pertain to infinitary objects and computations.

► **Proposition 12** (Infinitary subject reduction). *If $P \triangleright C \vdash t : T$ is approximable and $t \rightarrow_\beta^\infty t'$, then there is an approximable derivation $P' \triangleright C \vdash t' : T$.*

If a term is approximably typable, then, in particular, it is *finitely* typable, so that it is HN, as for usual, inductive intersection type systems:

► **Lemma 13** (Approximability and Head Normalization). *If $P \triangleright C \vdash t : T$ is approximable, then t is head normalizing.*

Approximable \mathbf{S} -derivations ensure only *head* normalization because of the empty sequence $()$, which allows us to leave an argument untyped. For instance, if x is assigned $() \rightarrow o$, then xt is typed with o for any term t . To ensure \mathbf{WN}_∞ , one needs to control the occurrences of $()$: by definition, the empty sequence type $()$ occur negatively in $() \rightarrow T$ (base case), $()$ occurs negatively (resp. positively) in $(S_k)_{k \in K} \rightarrow T$ if it occurs negatively (resp. positively) in T or positively (resp. negatively) in one of the S_k (inductive case).

► **Definition 14** (Unforgetfulness). *Let $P \triangleright C \vdash t : T$ be a derivation. Then P is **unforgetful** when $()$ does not occur negatively in C and does not occur positively in T .*

In particular, when $()$ does not occur in C nor in T , then P is unforgetful.

► **Proposition 15** (Correctness for system \mathbf{S}). *If $P \triangleright C \vdash t : T$ is approximable and unforgetful, then t is infinitary weakly normalizing.*

Completeness for infinitary normal forms—*i.e.*, the fact that they are approximably and unforgetfully typable—is proved in two steps: one shows that every *quantitative* derivation typing a normal form is approximable (this is not true for non-normal forms). Since one finds quantitative unforgetful derivations for each normal form, one concludes:

- 280 ► **Lemma 16** (Completeness for Normal Forms). *Let t be a NF_∞ .*
 281 ■ *If $P \triangleright C \vdash t : T$ and P is quantitative, then P is approximable.*
 282 ■ *t is approximably typable by derivation P such that $\text{supp}(P) = \text{supp}(t)$.*

283 Subject expansion holds for productive reduction paths:

- 284 ► **Proposition 17** (Infinitary subject expansion). *If $t \rightarrow_\beta^\infty t'$ and $P' \triangleright C \vdash t' : T$ is approximable,*
 285 *then there is an approximable derivation concluding with $C \vdash t : T$.*

286 From Lemma 16 and Proposition 17, one concludes that every infinitary weakly normalizing term is approximably and unforgetfully typable.

288 Last, Propositions 12 and 17 entail:

- 289 ► **Lemma 18.** *If t is WN_∞ , then t and $\text{BT}(t)$ have the same approximable typings.*

290 2 Characterizing hereditary permutators

291 We now want to define the *permutator pairs* (S, T) (with S, T types of system \mathbf{S}) so that the
 292 judgments of the form $x : (2 \cdot S) \vdash t : T$ characterize the x -HP (*i.e.*, there is an approximable
 293 $P \triangleright x : (2 \cdot S) \vdash t : T$ iff t is a x -HP). Informally, if $h = \lambda x_1 \dots x_n. x h_{\sigma(1)} \dots h_{\sigma(n)}$ and h is
 294 typed with a type of arity n and x_1, \dots, x_n are the respective head variables of h_1, \dots, h_n ,
 295 then we have:

296 ■ Type of $h = (\text{type of } x_1) \rightarrow \dots \rightarrow (\text{type of } x_n) \rightarrow o$ (eq₁)

297 ■ Type of $x = (\text{type of } h_{\sigma(1)}) \rightarrow \dots \rightarrow (\text{type of } h_{\sigma(n)}) \rightarrow o$ (eq₂)

298 Since x_1, \dots, x_n are the respective head variables of the headed hereditary permutators
 299 h_1, \dots, h_n , the equations (eq₁) and (eq₂), which are the golden thread of the proofs to come
 300 in the remainder of the paper, suggest the following coinductive definition:

- 301 ► **Definition 19** (Permutators pairs).

302 ■ *When o ranges over \mathcal{O} , the set $\text{PP}(o)$ of ***o -permutator pairs*** (S, T) , where S and T are*
 303 *\mathbf{S} -types, is defined by mutual coinduction:*

$$304 \frac{(S_1, T_1) \in \text{PP}(o_1), \dots, (S_n, T_n) \in \text{PP}(o_n) \quad o_1, \dots, o_n, o \text{ pairwise distinct} \quad \sigma \in \mathfrak{S}_n}{(2 \cdot T_{\sigma(1)}) \rightarrow \dots \rightarrow (2 \cdot T_{\sigma(n)}) \rightarrow o, (2 \cdot S_1) \rightarrow \dots (2 \cdot S_n) \rightarrow o \in \text{PP}(o)}$$

305 ■ *A pair $(S, T) \in \text{PP}(o)$ is said to be ***proper***, if, for all $o' \in \mathcal{O}$, o' occurs at most once in S*
 306 *and in T . The set of proper o -permutator pairs is denoted $\text{PPP}(o)$.*

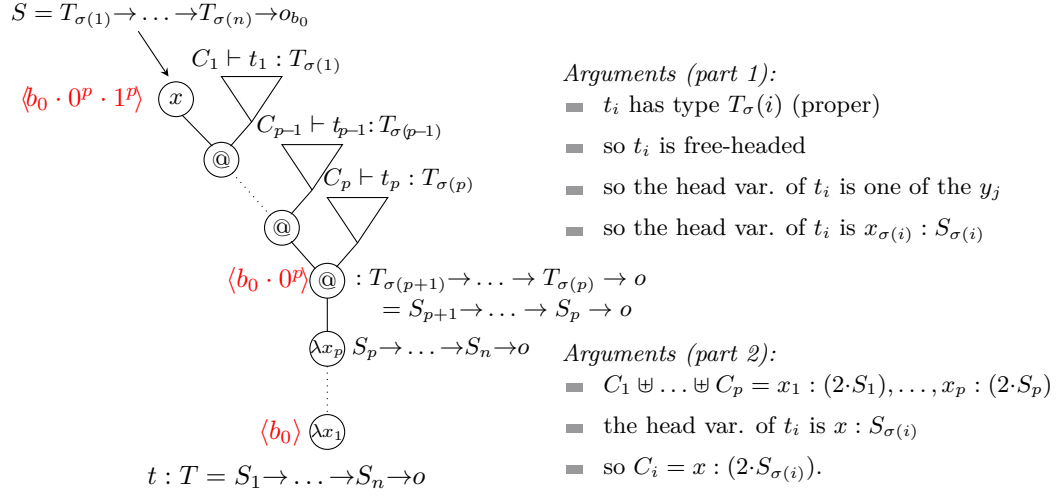
307 Actually, we could allow other tracks than 2 in the definition (*e.g.*, $T = (\ell_1 \cdot S_1) \rightarrow \dots \rightarrow$
 308 $(\ell_n \cdot S_n) \rightarrow o$ would be fine), but it is more convenient to consider this restriction, so that we
 309 are relieved of the care of specifying the values of ℓ_1, \dots, ℓ_n .

310 The condition of properness is here to ensure that every term variable occurs at exactly
 311 one level deeper than its binder and to distinguish them from one another: it is a key point
 312 of the proof of Claim 21, because two distinct variables will have types with distinct targets.

313 The first implication of the characterization is quite natural to prove:

- 314 ► **Claim 20** (From hereditary permutators to permutator pairs). *Let $y \in \mathcal{V}$ and t be a y -head*
 315 *hereditary permutator. Then there is an approximable \mathbf{S} -derivation P and a permutator pair*
 316 *(S, T) such that $P \triangleright y : (2 \cdot S) \vdash h : T$.*

317 **Proof.** We skip the proof of this property (it is given in Appendix A.1). Observe that
 318 Definition 19 is designed so that it holds. The converse claim (Claim 21) is more difficult to
 319 prove and requires to be carefully verified. ◀



■ **Figure 1** Hereditary permutators and permutator pairs

320 ▷ **Claim 21** (From permutator pairs to hereditary permutators). Let $t \in \Lambda^{001}$ be a 001-normal
 321 form and (S, T) a permutator pair and P a quantitative S-derivation typing t .
 322 ■ If $P \triangleright \vdash t : (2 \cdot S) \rightarrow T$, then t is a hereditary permutator.
 323 ■ If $P \triangleright x : (2 \cdot S) \vdash T$, then t is a x -headed hereditary permutator.
 324 In both cases, $\text{supp}(P) = \text{supp}(t)$.

325 **Proof.** The proof uses the following observation: let us say that a HNF is **free-headed** when
 326 its head variable is free. If (S, T) is a *proper* permutator pair and $t = \lambda x_1 \dots \lambda x_p. x_i t_1 \dots t_q$
 327 (with $1 \leq i \leq p$) is a HNF which is not free headed, then, t cannot have the types S and T ,
 328 since the target of the type of x_i appears twice in the type of t .

329
 330 We now start the proof, whose main stages are summarized in Fig. 1, in which we
 331 abusively write S instead of $(2 \cdot S)$. Assume $S = (2 \cdot T_{\sigma(1)}) \rightarrow \dots \rightarrow (2 \cdot T_{\sigma(n)}) \rightarrow o$ and
 332 $T = (2 \cdot S_1) \rightarrow \dots \rightarrow (2 \cdot S_n) \rightarrow o$. We first prove that the first point of the claim reduces to
 333 the second one.

334 Since the context in $\vdash t : (2 \cdot S) \rightarrow T$ is empty, the head variable of t is bound and
 335 the arity of t is ≥ 1 . Thus, $t = \lambda x_0. \lambda y_1 \dots \lambda y_p. x t_1 \dots t_q$ with t_1, \dots, t_q normal forms whose
 336 respective head variables are denoted x_1, \dots, x_q . Note that:

- 337 ■ x is x_1, \dots, x_p or x_0 since x is bound.
 338 ■ The type assigned to x_0 is S . The respective types assigned to x_1, \dots, x_q are S_1, \dots, S_q .
 339 ■ The common target type of T and the type of x_0 is o .

340 Since (S, T) is proper, o does not occur in S_1, \dots, S_n , so necessarily, $x = x_0$ and $x :$
 341 $(2 \cdot S) \vdash \lambda y_1 \dots \lambda y_p. x t_1 \dots t_q : T$ is derivable by means of a quantitative derivation P_* . Thus,
 342 we are now in the second case. The type of $x t_1 \dots t_q$ is both $T_{\sigma(p+1)} \rightarrow \dots \rightarrow T_{\sigma(n)} \rightarrow o$ since
 343 $x : S$ and $S_{p+1} \rightarrow \dots \rightarrow S_n \rightarrow o$ since $t : T$, so $p+1 \leq i \leq n$, $\sigma(i) = i$ and $S_i = T_i$. Let
 344 us denote o_1, \dots, o_n the respective target types of S_1, \dots, S_n . Since the type of x is S , the
 345 respective types of t_1, \dots, t_p must be $T_{\sigma(1)}, \dots, T_{\sigma(p)}$. Moreover, since the “tail” of T is made
 346 of singleton sequence types $(2 \cdot S_i)$, t_1, \dots, t_p are typed once in P and the head variables of
 347 t_1, \dots, t_p are also typed exactly once. In particular, P_* has a subderivation at depth n of the
 348 form:

$$\frac{\frac{\frac{}{x : (2 \cdot S) \vdash x : S} \text{ax} \quad P_i \triangleright C_i \vdash t_i : T_{\sigma(i)}}{x : (2 \cdot S), C_1 \vdash x t_1 : (2 \cdot S_{\sigma(2)}) \rightarrow \dots \rightarrow o} \text{app} \quad \dots}{\dots} \text{app} \quad \frac{\dots \quad P_p \triangleright C_p \vdash t_p : T_{\sigma(p)}}{x : (2 \cdot S), C_1 \uplus \dots \uplus C_p \vdash x t_1 \dots t_p : T'} \text{app}$$

349 where $T' = T_{\sigma(p+1)} \rightarrow \dots \rightarrow T_{\sigma(n)} \rightarrow o = S_{p+1} \rightarrow \dots \rightarrow S_n \rightarrow o$.

350 Let us prove now that, for all $1 \leq i \leq p$, the unique argument derivation of x in P typing
 351 t_i , that we denote P_i , concludes with $x_{\sigma(i)} : (2 \cdot S_{\sigma(i)}) \vdash t_i : T_{\sigma(i)}$.

352 First, since t_i is normal, $t_i = \lambda z_1 \dots z_{p'} . z u_1 \dots u'_q$. Since $t_i : T_{\sigma(i)}$, t_i is free-headed by
 353 the observation above. Moreover, the head variable of t_i is typed once since t_i is typed once.
 354 Thus, z is one of the x_1, \dots, x_p . The only possibility is $z = x_{\sigma(i)}$ since the types of x_1, \dots, x_p
 355 have pairwise distinct targets.

356 Since P is quantitative and $(2 \cdot S_i)$ is a *singleton* sequence type, x_1, \dots, x_p must be
 357 exactly typed once in the subderivation of P typing $x t_1 \dots t_p$. This entails that the **ax**-rule
 358 typing $x_{\sigma(i)}$ as the head variable of t_i concludes with $x_{\sigma(i)} : (2 \cdot S_{\sigma(i)}) \vdash x_{\sigma(i)} : S_{\sigma(i)}$. Thus,
 359 P_i concludes with a judgment of the form $x_{\sigma(i)} : (2 \cdot S_{\sigma(i)}) \uplus C'_i \vdash t_i : T_{\sigma(i)}$ (2nd argument).

360 Since $\uplus_{1 \leq i \leq p} (x_{\sigma(i)} : (2 \cdot S_{\sigma(i)}) \uplus C'_i) = x_1 : (2 \cdot S_1), \dots, x_p : (2 \cdot S_p)$, we deduce that C'_i is
 361 empty for all $1 \leq i \leq p$ (3rd argument). Thus, P_i concludes with $x_{\sigma(i)} : (2 \cdot S_{\sigma(i)}) \vdash t_i : T_{\sigma(i)}$.

362 This easily implies that $x_1 : S_1 \vdash t_{\sigma^{-1}(1)} : T_1, \dots, x_p : S_p \vdash t_{\sigma^{-1}(p)} : T_p$ are judgments of
 363 P_* . In particular, they are approximably derivable. We conclude by coinduction. ◀

364 The two claims, which are valid for 001-normal forms, along with infinitary subject
 365 reduction and expansion, give a type-theoretical characterization of a hereditary permutators
 366 in system **S**:

367 ► **Theorem 22.** *Let $t \in \Lambda^{001}$. Then t is a hereditary permutator iff $\vdash t : (2 \cdot S) \rightarrow T$ is*
 368 *approximably derivable for some proper permutator pair (S, T) .*

369 **Proof.**

- 370 ■ The implication \Leftarrow is given by Claim 21 and Proposition 17.
- 371 ■ Implication \Rightarrow : let t be a hereditary permutator. By Definition 1, its Böhm tree is
 372 of the form $\lambda x . h$ where h is a normal x -headed hereditary permutator. By Claim 20,
 373 there is a proper permutator pair (S, T) and an approximable derivation P such that
 374 $P \triangleright x : (2 \cdot S) \vdash h : T$. By Proposition 17, $\vdash t : (2 \cdot S) \rightarrow T$ is also approximably derivable.
 375 ◀

376 3 A unique type to rule them all

377 In this section, we explain how to enrich system **S** with types constants and typing rules so
 378 that there is one type characterizing the set of hereditary permutators, as expected.

379 In Sec. 2, we proved that a term t is a hereditary permutator iff it can be assigned a type
 380 of the form $(2 \cdot S) \rightarrow T$ where (S, T) is a proper permutator pair. To obtain a unique type for
 381 all the hereditary permutator, one idea is to identify all the types of the form $(2 \cdot S) \rightarrow T$,
 382 where (S, T) ranges over PPP with a type constant **ptyp**. However, since quotienting types
 383 may bring unsoundness (*e.g.*, if o and $o \rightarrow o$ are identified), one must then verify that the
 384 correctness and the completeness of system **S** is preserved, and that the approximability
 385 criterion can be suitably extended. The main argument, given by Lemma 26, is that the

notions of hereditary permutators and permutators pairs, which are infinitary, have arbitrarily big finite approximations, which are defined as truncatures at some applicative depth d . Thus, we may express hereditary permutators and permutator pairs as asymptotic limits and adapt the general methods of system \mathbf{S} .

Our approach parallels that of Tatsuta [14], which uses a family of constants \mathbf{ptyp}_d , with a few differences: in the finite restriction of our system, it is easier to deal with hereditary permutators (normalization is simple to prove in finite non-idempotent type systems), but of course we have to treat the infinitary typings and we consider the constant \mathbf{ptyp} , which is subsumed by all the \mathbf{ptyp}_d , who represent hereditary permutators under level d .

3.1 Permutator schemes

Before presenting the system giving a unique type to all hereditary permutators, we must first explain how the typing of hereditary permutators are approximated in system \mathbf{S} .

► **Definition 23** (Permutator schemes). *Let $d \geq 0$. A term t is a x -headed (resp. closed) permutator scheme of degree d if its Böhm tree is equal to that of a hereditary permutator on $\{b \in \{0, 1, 2\}^* \mid \text{ad}(b) \leq d\}$. The set of x -headed (resp. closed) permutators schemes of degree d is denoted $\mathbf{PS}_d(x)$ (resp. \mathbf{PS}_d).*

The sequence (\mathbf{PS}_d) is decreasing, i.e., $\mathbf{PS}_d \supseteq \mathbf{PS}_{d+1}$, and $\mathbf{HP} = \bigcap_{d \geq 0} \mathbf{PS}_d$.

► **Definition 24** (Permutator pairs of degree d).

■ When o ranges over \mathcal{O} , the set $\mathbf{PP}(o)$ of *o -permutator pairs of degree d* (S, T) , where S and T are \mathbf{S} -types, is defined by induction on d :

$$\frac{((\underbrace{() \rightarrow \dots ()}_{n} \rightarrow o, \underbrace{() \rightarrow \dots ()}_{n} \rightarrow o) \in \mathbf{PP}_0(o))}{(S_1, T_1) \in \mathbf{PP}_{d-1}(o_1), \dots, (S_n, T_n) \in \mathbf{PP}_{d-1}(o_n) \quad o_1, \dots, o_n, o \text{ pairwise distinct} \quad \sigma \in \mathfrak{S}_n}$$

$$\frac{(S_1, T_1) \in \mathbf{PP}_{d-1}(o_1), \dots, (S_n, T_n) \in \mathbf{PP}_{d-1}(o_n) \quad o_1, \dots, o_n, o \text{ pairwise distinct} \quad \sigma \in \mathfrak{S}_n}{(2 \cdot T_{\sigma(1)}) \rightarrow \dots \rightarrow (2 \cdot T_{\sigma(n)}) \rightarrow o, (2 \cdot S_1) \rightarrow \dots \rightarrow (2 \cdot S_n) \rightarrow o \in \mathbf{PP}_d(o)}$$

■ A pair $(S, T) \in \mathbf{PP}_d(o)$ is said to be **proper** if every type variable occurs at most once in S and T . The set of proper permutator pairs of degree d is denoted \mathbf{PP}_d .

We can also see permutator pairs of degree d as truncation of permutator pairs: let U be a \mathbf{S} -type or a sequence type and $d \in \mathbb{N}$. We denote by $(U)^{\leq d}$ the truncation of T at depth d i.e., $\text{supp}((U)^{\leq d}) = \text{supp}(U) \cap \{c \in \mathbb{N}^* \mid \text{ad}(c) \leq d\}$ and $(U)^{\leq d}(c) = U(c)$ for all $c \in \text{supp}((U)^{\leq d})$. It is easy to check that $(U)^{\leq d}$ is a correct type or sequence type. We extend the notation to \mathbf{S} -contexts. Note that, if $d \geq 1$, $((S_k)_{k \in K} \rightarrow T)^{\leq d} = ((S_k)^{\leq d-1})_{k \in K} \rightarrow (T)^{\leq d}$ and $d = 1$, then $((S_k)_{k \in K} \rightarrow T)^{\leq d} = () \rightarrow (T)^{\leq 1}$. By induction on d , this entails that, if $(S, T) \in \mathbf{PPP}$, then $((S)^{\leq d}, (T)^{\leq d}) \in \mathbf{PPP}_d$. Indeed, the base case ($d = 0$) is obvious and if $d \geq 1$, $T = (2 \cdot S_1) \rightarrow \dots \rightarrow (2 \cdot S_n) \rightarrow o$ and $S = (2 \cdot T_{\sigma(1)}) \rightarrow \dots \rightarrow (2 \cdot T_{\sigma(n)}) \rightarrow o$ with $\sigma \in \mathfrak{S}_n$, $(S_i, T_i) \in \mathbf{PPP}$ for $1 \leq i \leq n$, then:

$$(T)^{\leq d} = (2 \cdot (S_1)^{\leq d-1}) \rightarrow \dots \rightarrow (2 \cdot (S_n)^{\leq d-1}) \rightarrow o \quad (\text{eq}_3)$$

$$(S)^{\leq d} = (2 \cdot (T_{\sigma(1)})^{\leq d-1}) \rightarrow \dots \rightarrow (2 \cdot (T_{\sigma(n)})^{\leq d-1}) \rightarrow o \quad (\text{eq}_4)$$

so that, by Definition 24, $((S)^{\leq d}, (T)^{\leq d}) \in \mathbf{PPP}_d(o)$.

► **Proposition 25** (Characterizing permutation schemes). *Let $d \geq 1$ and t be a 001-term. Then $t \in \mathbf{PS}_d$ iff $\vdash t : (2 \cdot S) \rightarrow T$ is approximably derivable for some $(S, T) \in \mathbf{PPP}_d$.*

Proof. \Rightarrow Straightforward induction on the structure of t .

23:12 Typing Hereditary Permutators

426 \Leftarrow The proof is the same as Claim 21, we also obtain that $x_i : (2 \cdot S_i) \vdash t_{\sigma^{-1}(i)} : T_i$ are
 427 judgments of P , except that $(S_i, T_i) \in \text{PPP}_{d-1}$ instead of $(S_i, T_i) \in \text{PPP}$.
 428 \blacktriangleleft

429 It is not enough to know that a x -headed hereditary permutator t is approximably typable
 430 in a judgment $x : (2 \cdot S) \vdash t : T$ with $(S, T) \in \text{PPP}$, which implies that T is the supremum of
 431 a direct family of finite types which be assigned to t : in order to prove soundness regarding
 432 quotienting, we must prove that this typing is the supremum of typings ensuring that t is a
 433 permutator scheme of degree d , *i.e.*, by Proposition 25, one must type t with $(S_d, T_d) \in \text{PPP}_d$
 434 for all d . The lemma below is the missing third ingredient (along with Claims 20 and 21) of
 435 this article and will allow us to define in Sec. 3.2 an extension of system \mathbf{S} giving a unique
 436 type to hereditary permutators:

437 **► Lemma 26** (Approximations and permutator pairs). *If $P \triangleright x : (2 \cdot S) \vdash t : P$ is approximable,*
 438 *where $(S, T) \in \text{PPP}$, then, for all $d \in \mathbb{N}$, there is a finite $P_d \leq P$ such that $P_d \triangleright x : (2 \cdot S_d) \vdash$*
 439 *$t : T_d$ with $(S_d, T_d) \in \text{PPP}_d$.*

440 **Proof.** Since $()$ does not occur in S and T , by Lemma 18, we can assume that t is a
 441 001-normal form without loss of generality. We then reason by induction on d .

442 Let us present the argument informally (a formal proof is given in Appendix A.2). Say that
 443 $t = \lambda x_1 \dots x_p. x t_1 \dots t_p$, $S = T_{\sigma(1)} \rightarrow \dots \rightarrow T_{\sigma(n)} \rightarrow o$ and $T = S_1 \rightarrow \dots \rightarrow S_n \rightarrow o$. Intuitively,
 444 $t : T$ with $x : S$ and for $1 \leq i \leq p$, $t_i : T_{\sigma(i)}$ is a hereditary permutator headed by $x_{\sigma(i)} : S_{\sigma(i)}$,
 445 as specified by Fig. 1. When we truncate the type of t at applicative depth d , we have now
 446 $t : (T)^{\leq d}$ with $x : (S)^{\leq d}$. But, by (eq₃) and (eq₄), we must truncate the types of t_1, \dots, t_p
 447 and x_1, \dots, x_p at applicative depth $d - 1$. Inductively, this demands that we truncate the
 448 types of the arguments of t_1, \dots, t_p at applicative depth $d - 2$. By proceeding so, we obtain
 449 a finite derivation $P_d \leq P$ concluding with $x : (2 \cdot (S)^{\leq d}) \vdash t : (T)^{\leq d}$.
 450 \blacktriangleleft

451 3.2 System \mathbf{S}_{hp}

Let ptyp and ptyp_d ($d \in \mathbb{N}$) be a family of type constants. The set of \mathbf{S}_{hp} -types is defined by:

$$T, S_k ::= o \parallel \text{ptyp}_d \parallel \text{ptyp} \parallel (S_k)_{k \in K} \rightarrow T$$

452 System \mathbf{S}_{hp} has the same typing rules as system \mathbf{S} with the addition of:

$$453 \quad \frac{x : (2 \cdot S) \vdash t : T \quad (S, T) \in \text{PPP}_d}{\vdash \lambda x. t : \text{ptyp}_d} \text{hp}_d \quad \frac{x : (2 \cdot S) \vdash t : T \quad (S, T) \in \text{PPP}}{\vdash \lambda x. t : \text{ptyp}} \text{hp}$$

454 Thus, rule hp_d allows assigning the constant ptyp_d to any normal permutator scheme of
 455 degree d and rule hp assign the constant ptyp to any normal hereditary permutator by
 456 Claims 20 and 21. Intuitively, $\text{ptyp} = \text{ptyp}_\infty$ and we will make this idea more precise with
 457 Definition 27. Note also that if $t : \text{ptyp}_d$ or $t : \text{ptyp}$, t cannot be applied to an argument u ,
 458 even if t is an abstraction: the rules hp_d/hp freeze the terms.

459 The notions of support, bisupport, permutator pairs *etc* naturally extend to \mathbf{S}_{hp} . We
 460 define an order \leq on $\mathcal{O} \cup \{\rightarrow, \text{ptyp}\} \cup \{\text{ptyp}_d \mid d \in \mathbb{N}\}$ by $o \leq o$, $\rightarrow \leq \rightarrow$, $\text{ptyp}_d \leq \text{ptyp}$ and
 461 $\text{ptyp}_d \leq \text{ptyp}'_d$ for $d \leq d'$.

462 **► Definition 27** (Approximation and Approximability in system \mathbf{S}_{h}).

463 **■** Let P_0 and P be two \mathbf{S}_{hp} -derivations. We write $P_0 \leq P$ (P_0 is an approximation of P) if
 464 $\text{bisupp}(P_0) \subseteq \text{bisupp}(P)$ and, for all $p \in \text{bisupp}(P_0)$, $P_0(p) \leq P(p)$.

465 ■ Let P be a $\mathbf{S}_{\mathbf{hp}}$ -derivation. Then P is **approximable** if P is the supremum of its finite
466 approximations.

467 This extends Definition 10: for all \mathbf{S} -derivations P , P is approximable for system \mathbf{S} iff it
468 is approximable for system $\mathbf{S}_{\mathbf{hp}}$. We first notice that rules $\mathbf{hp}_{(d)}$ are invertible for HNF:

469 ► **Lemma 28** (Inverting rules (\mathbf{hp}_d) for head normal forms). *Let t be a HNF. If $\vdash t : \mathbf{ptyp}_d$
470 (resp. $P \triangleright \vdash t : \mathbf{ptyp}$) is approximably derivable, then $t = \lambda x.t_0$ with $x : (2 \cdot S) \vdash t_0 : T$
471 approximably derivable, for some $(S, T) \in \mathbf{PPP}_d$ (resp. $(S, T) \in \mathbf{PPP}$).*

Proof. We consider the case \mathbf{ptyp} (the case \mathbf{ptyp}_d is similar), i.e., we assume that $P' \triangleright \vdash t : \mathbf{ptyp}$ is approximable. For one, $t = x t_1 \dots t_n$ is impossible, because we would have $C(x) \neq ()$ since the head variable x is free in $x t_1 \dots t_n$. So, t is an abstraction, i.e., $t = \lambda x.t_0$ and thus, the last rule of P is either **abs**, \mathbf{hp}_d , **hp**. But it is neither **abs** (we would have an arrow type) nor \mathbf{hp}_d , so it is **hp** and thus, P' is of the form:

$$P' = \frac{P \triangleright x : (2 \cdot S) \vdash t_0 : T \quad (S, T) \in \mathbf{PPP}}{\vdash t : \mathbf{ptyp}} \mathbf{hp}$$

472 Since P' is approximable, P also is. ◀

473 All is now in place to obtain the expected properties of system $\mathbf{S}_{\mathbf{hp}}$:

474 ► **Lemma 29** (Characterizing normal hereditary permutators). *Let t be a 001-normal form.*

475 ■ $t \in \mathbf{PS}_d$ iff $\vdash t : \mathbf{ptyp}_d$ is approximably derivable.

476 ■ $t \in \mathbf{HP}$ iff $\vdash t : \mathbf{ptyp}$ is approximably derivable.

477 **Proof.** The two points are handled similarly. We do not prove the first one, which uses
478 Proposition 25:

■ If $t = \lambda x.h$ is a HP, then, by Claim 20, there is $(S, T) \in \mathbf{PPP}$ and P a \mathbf{S} -derivation such that $P \triangleright x : (2 \cdot S) \vdash h : T$. We then set:

$$P' = \frac{P \triangleright h : (2 \cdot S) \vdash p : T}{\vdash t : \mathbf{ptyp}} \mathbf{hp}$$

By Lemma 26, for all $d \in \mathbb{N}$, there is a finite \mathbf{S} -derivation $P_d \leq P$ such that $P_d \triangleright x : (2 \cdot S_d) \vdash h : T_d$ with $(S_d, T_d) \in \mathbf{PPP}_d$ and $P = \sup_d P_d$. We then set:

$$P'_d = \frac{P_d \triangleright x : (2 \cdot S_d) \vdash h : T_d}{\vdash t : \mathbf{ptyp}_d} \mathbf{hp}_d$$

479 By construction, $\sup_d P'_d = P'$.

■ Conversely, assume that $P' \triangleright \vdash t : \mathbf{ptyp}$ is approximable. By Lemma 28, P' concludes with the **hp**-rule, so P' is of the form:

$$P' = \frac{P \triangleright x : (2 \cdot S) \vdash t_0 : T \quad (S, T) \in \mathbf{PPP}}{\vdash t : \mathbf{ptyp}} \mathbf{hp}$$

480 Let $d \in \mathbb{N}$. Since P' is the supremum of its finite approximations, there is a finite
481 approximation $P'_0 \leq P'$ concluding with¹ $\vdash t : \mathbf{ptyp}$ or $\vdash t : \mathbf{ptyp}_{d'}$ with $d' \geq d$. Thus,
482 $t \in \mathbf{PS}_{d'} \subset \mathbf{PS}_d$ or $t \in \mathbf{HP}$. This proves that $t \in \bigcap_{d \geq 0} \mathbf{PS}_d = \mathbf{HP}$.

¹ The case $P'_0 \triangleright \vdash t : \mathbf{ptyp}$ is possible: there are finite HP and PPP, e.g., $\lambda x.x$ and (o, o) .

483

484 ► **Lemma 30** (Soundness of system S_{hp}). *If t is approximably typable in system S_{hp} , then t is*
 485 *head normalizing.*

486 **Proof.** If t is approximably typable, there is a *finite* S_{hp} -derivation $P \triangleright C \vdash t : T$. If t is a
 487 HNF, we are done. In the other case, $t \rightarrow_h t'$ for some t' . It is routine work in *non-idempotent*
 488 intersection type theory (see [4]) to prove a *weighted* subject reduction property, *i.e.*, that
 489 there is $P' \triangleright C \vdash t' : T$ such that $|\text{supp}(P')| < |\text{supp}(P)|$, *i.e.*, P' contains strictly less
 490 judgments than P does. The only unusual rule are hp_d and hp , which are easily handled.

491 Since $|\text{supp}(P)| \in \mathbb{N}$ and \mathbb{N} is well-founded, weighted subject reduction entails that head
 492 reduction terminates on t . ◀

493 ► **Corollary 31.** *If $\vdash t : \text{ptyp}$ is approximably derivable, then t is WN_∞ .*

494 **Proof.** By Lemma 30, t reduces to a HNF t' . By subject reduction, $\vdash t' : \text{ptyp}$ is also
 495 approximably derivable. Then, by Lemma 28, entails that $t = \lambda x.t_0$ and $x : (2 \cdot S) \vdash t_0 : T$ is
 496 approximably derivable in system S . Since this latter judgment is $()$ -free, Proposition 15
 497 entails that t_0 is WN_∞ . Thus, t also is WN_∞ . ◀

498 More generally, the dynamic properties of system S are preserved in system S_{hp} .

499 ► **Proposition 32** (Infinitary subject reduction). *If $t \rightarrow_\beta^\infty t'$ and $P \triangleright C \vdash t : T$ is an*
 500 *approximable S_{hp} -derivation, then there exists an approximable derivation $P' \triangleright C \vdash t' : T$.*

501 ► **Proposition 33** (Infinitary subject reduction). *If $t \rightarrow_\beta^\infty t'$ and $P' \triangleright C \vdash t' : T$ is an*
 502 *approximable S_{hp} -derivation, then there exists an approximable derivation $P \triangleright C \vdash t : T$.*

503 **Proof.** The proofs of infinitary subject reduction and expansion in system S_{hp} do not differ
 504 of those for system S , which can be found in [17] (in particular, Sec. II.D. and VI.D.) or in
 505 Chapter 10 of [18], so we do not give the details. Once again, the only new rules are hp and
 506 hp_d , which are easily handled in the one step case, then in the asymptotic case.

507 Infinitary subject reduction is easy to prove, but infinitary subject expansion holds
 508 because we can expand *finite* approximations of a derivation P' concluding a productive
 509 reduction path. Why? Because, if $t \rightarrow_\beta^\infty t'$ and P'_f is finite and types t' , then there is a term
 510 $t \rightarrow_\beta^k t_k$ obtained from t after a finite number k of β -reduction steps such that, on $\text{supp}(P')$,
 511 t_k and t' induce the same subtree (this is a consequence of Definition 5). Thus, we can
 512 *substitute* t' with t_k in P'_f : we then obtain P_f^k typing t_k . After k expansion steps, we obtain
 513 from P_f^k a derivation P_f typing t . To conclude this dense summary, let us just say that the
 514 infinitary subject expansion is not so about the rules than about the possibility to replace a
 515 derivation by its finite approximations, which is precisely what Definition 27 captures for
 516 system S_{hp} . ◀

517 We now give a positive answer to TLCA Problem # 20:

518 ► **Theorem 34** (Characterizing hereditary permutators with a unique type). *Let $t \in \Lambda^{001}$. Then*
 519 *t is a hereditary permutator iff $\vdash t : \text{ptyp}$ is approximably derivable in system S_{hp} .*

520 **Proof.** \Rightarrow If t is a HP, let t' be its 001-NF. By Proposition 29, there is an approximable
 521 derivation $P' \triangleright \vdash t' : \text{ptyp}$. By Proposition 33, there is $P \triangleright \vdash t' : \text{ptyp}$ approximable.

522 \Leftarrow Given by Corollary 31.

523

Future work

We plan to adapt system S to characterize other sets of Böhm trees and other notions of infinitary normalization, including weak normalization in the calculi Λ^{101} and Λ^{111} of [11].

References

- 1 Patrick Bahr. Strict ideal completions of the lambda calculus. In FSCD 2018, July 9-12, Oxford, pages 8:1–8:16, 2018.
- 2 Henk Barendregt. The Lambda-Calculus: Its Syntax and Semantics. Ellis Horwood series in computers and their applications. Elsevier, 1985.
- 3 Jan A. Bergstra and Jan Willem Klop. Invertible terms in the lambda calculus. Theor. Comput. Sci., 11:19–37, 1980.
- 4 Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. Non-idempotent intersection types for the lambda-calculus. Mathematical Structures in Computer Science., 2017.
- 5 Daniel De Carvalho. Sémantique de la logique linéaire et temps de calcul. PhD thesis, Université Aix-Marseille, November 2007.
- 6 Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the λ -calculus. Notre Dame Journal of Formal Logic, 4:685–693, 1980.
- 7 Haskell B. Curry and Robert Feys. Combinatory Logic, volume I. North-Holland Co., Amsterdam, 1958. (3rd edn. 1974).
- 8 Lukasz Czakka. A coinductive confluence proof for infinitary lambda-calculus. In Rewriting and Typed Lambda Calculi - Joint International Conference, RTA-TLCA, Vienna, Austria, July 14-17, pages 164–178, 2014.
- 9 Mariangiola Dezani-Ciancaglini. Characterization of normal forms possessing inverse in the lambda-beta-eta-calculus. Theor. Comput. Sci., 2(3):323–337, 1976.
- 10 Philippa Gardner. Discovering needed reductions using type theory. In TACS, Sendai, 1994.
- 11 Richard Kennaway, Jan Willem Klop, M. Ronan Sleep, and Fer-Jan de Vries. Infinitary lambda calculus. Theor. Comput. Sci., 175(1):93–125, 1997.
- 12 Betti Venneri Mario Coppo, Mariangiola Dezani-Ciancaglini. Functional characters of solvable terms. Mathematical Logic Quarterly, 27:45–58, 1981.
- 13 Makoto Tatsuta. Types for hereditary head normalizing terms. In FLOPS, Ise, Japan, April 14-16, pages 195–209, 2008.
- 14 Makoto Tatsuta. Types for hereditary permutators. In LICS, 24-27 June, Pittsburgh, pages 83–92, 2008.
- 15 Steffen van Bakel. Complete restrictions of the intersection type discipline. Theor. Comput. Sci., 102(1):135–163, 1992.
- 16 Steffen van Bakel. Intersection type assignment systems. Theor. Comput. Sci., 151(2):385–435, 1995.
- 17 Pierre Vial. Infinitary intersection types as sequences: a new answer to Klop’s problem. In LICS, Reykjavik, 2017.
- 18 Pierre Vial. Non-Idempotent Typing Operator, beyond the Lambda-Calculus. Phd thesis, Université Sorbonne Paris-Cité, 2017, available on <http://www.irif.fr/~pvial>.

A Appendix

A.1 Hereditary permutators are typable with permutator pairs

▷ Claim (given on p. 8). Let $y \in \mathcal{V}$ and t be a y -head hereditary permutator. Then there is an approximable \mathbf{S} -derivation P and a permutator pair $((2 \cdot S), T)$ such that $P \triangleright y : (2 \cdot S) \vdash h : T$.

Proof. Let ι be an injection from $B = \text{supp}(t)$ to \mathcal{O} . We associate to each $b \in \text{supp}(t)$ two indeterminates X_b and Y_b . The idea is that X_b is a placeholder for the types of head variables and Y_b is a placeholder for the types of the sub-hereditary permutators of t .

We denote by B^{hp} the set of positions b of subterms of t that are x -HP for some $x \in \mathcal{V}$ and, for all $b \in B^{\text{hp}}$, $\text{hvp}(b)$ denotes the position of the head variable of $t|_b$ (hvp stands for “head variable position”). Formally, we have $B^{\text{hp}} = \{\varepsilon\} \cup \{b \cdot 2 \in \text{supp}(t) \mid b \in \{0, 1, 2\}^*\}$ ($b \in B^{\text{hp}}$ is b is the root of t or it is the argument of an application in t) and, for all $b \in B^{\text{hp}}$, $\text{hvp}(b)$ is the longest b_0 such $b_0 \in b \cdot \{0, 1\}^*$. For all $b \in B^{\text{hp}}$, then we denote by x_b the head variable of $\text{hv}(t|_b)$, e.g., if $t = \lambda z_1 z_2. (y (\lambda z_3 z_4. z_2 t_1 t_2)) (\lambda z_5. z_1 t_3)$, then $\text{hvp}(\varepsilon) = 0^2 \cdot 1^2$ and $x_\varepsilon = y$, $t|_{0^2 \cdot 2} = \lambda z_5. z_1 t_3$, so $\text{hvp}(0^2 \cdot 2) = 0^2 \cdot 2 \cdot 1 \cdot 0$ and $x_{0^2 \cdot 2} = z_1$, $\text{hvp}(0^2 \cdot 1 \cdot 2) = 0^2 \cdot 1 \cdot 2 \cdot 1^2 \cdot 0^2$ and $x_{0^2 \cdot 1 \cdot 2} = z_2$. Observe that, if $b \in B^{\text{hp}}$ and $n = \text{ar}(t|_b)$, then $\text{hvp}(b) = b \cdot 0^n \cdot 1^n \in B^{\text{hp}}$. We just write o_b instead of $\iota(\text{hvp}(t|_b))$, so that o_b will be the type atom assigned to the head variable $y = t(\text{hvp}(b))$ of $t|_b$, which is a x -HP.

Moreover, for $b \in B^{\text{hp}}$, then $t|_b$ is of the form $\lambda x_1 \dots x_n. x h_{x_{\sigma(1)}} \dots h_{x_{\sigma(n)}}$ with $n = \text{ar}(t|_b) \geq 0$, $y = x_b$ and $\sigma \in \mathfrak{S}_n$. We then denote by σ_b the permutator σ and we set $b(k) = b \cdot 0^n \cdot 1^{k-1} \cdot 2$ for $1 \leq k \leq n$, so that $b(k)$ is the position of $h_{x_{\sigma(n)}}$. For $1 \leq k \leq n$, we also abusively write $b(\sigma(k))$ instead of $b(\sigma_b(k))$. We then set, for all $b \in B^{\text{hp}}$:

$$\begin{aligned} \mathbf{F}(b) &= (2 \cdot Y_{b(\sigma(1))}) \rightarrow \dots \rightarrow (2 \cdot Y_{b(\sigma(n))}) \rightarrow o_b \\ \mathbf{G}(b) &= (2 \cdot X_{b(1)}) \rightarrow \dots \rightarrow (2 \cdot X_{b(n)}) \rightarrow o_b \end{aligned}$$

We may then implement (eq_1) and (eq_2) by coinductively defining, for all $b \in B^{\text{hp}}$,

$$\begin{aligned} S(b) &= \mathbf{F}(b)[S(b')/X_{b'}, T(b')/Y_{b'}]_{b' \in B^{\text{hp}}} \\ T(b) &= \mathbf{G}(b)[S(b')/X_{b'}, T(b')/Y_{b'}]_{b' \in B^{\text{hp}}} \end{aligned}$$

The definition of $S(b)$ and $T(b)$ is well founded since if $X_{b'}$ or $Y_{b'}$ occur at position c in $S(b)$ or $T(b)$, then $\text{ad}(c) > 0$. By construction, for all $b \in B^{\text{hp}}$, $(S(b), T(b))$ is a *proper* permutator pair. Let us now construct a quantitative \mathbf{S} -derivation P , such that, for all $b \in B^{\text{hp}}$, $P(b) = x_b : (2 \cdot S(b)) \vdash t|_b : T(b)$. In particular, with $b = \varepsilon$, we will have $P \triangleright x : (2 \cdot S) \vdash t : T$ with $S = S(\varepsilon)$ and $T = T(\varepsilon)$. Since t is a normal form, by Lemma 16, P will be approximable, which will conclude the proof.

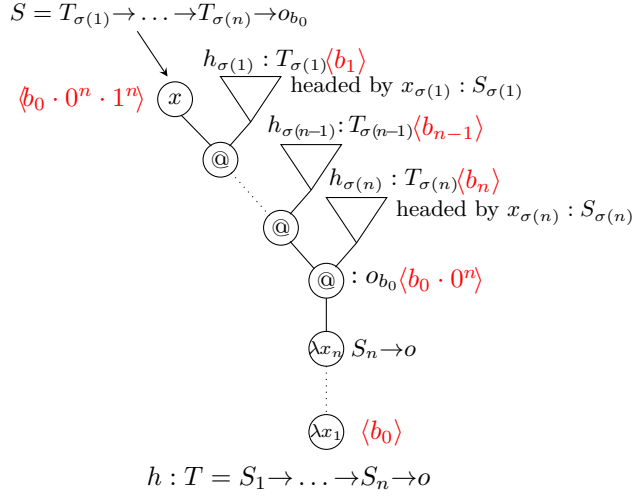
Construction of P : we now build P . The construction is illustrated with Fig. 2. The notation \diamond indicates the position, e.g., $\langle b_0 \rangle$ means that the node labelled with x_1 is at position b_0 . Let $b \in \text{supp}(t)$. There are three possibilities:

■ $t(b) = y$ for some $y \in \mathcal{V}$: then $b = b_0 \cdot 0^n \cdot 1^n$ with $b_0 \in B^{\text{hp}}$ and $n \geq 0$.

■ $t(b) = @$: then $b = b_0 \cdot 0^n \cdot 1^i$ with $b_0 \in B^{\text{hp}}$, $n \geq 0$ and $i < n$.

■ $t(b) = \lambda y$ for some $y \in \mathcal{V}$: then $b = b_0 \cdot 0^i$ with $b_0 \in B^{\text{hp}}$ and $i < n = \text{ar}(t|_{b_0})$.

Let $b_0 \in B^{\text{hp}}$. Then $t|_{b_0}$ is of the form $h = \lambda x_1 \dots x_n. y h_{\sigma(1)} \dots h_{\sigma(n)}$ for some $n \geq 0$, $\sigma \in \mathfrak{S}_n$ and h_1, \dots, h_n hereditary permutators respectively headed with x_1, \dots, x_n . Before defining the judgments in P , we give some preliminary notations and observations: we set $b_j = b_0 \cdot 0^n \cdot 1^{n-j} \cdot 2$ so that b_j is the position of $h_{\sigma(j)}$ for $0 \leq j \leq n$. In particular, $b_j \in B^{\text{hp}}$ and $S(b_j), T(b_j)$ are defined. We also set $b'_j = b_{\sigma^{-1}(j)}$. The type of $h_{\sigma(j)}$ is intended to



■ **Figure 2** Typing a hereditary permutator

606 be $T(b_j)$ and that of $x_{\sigma(j)}$ to be $S(b_j)$. The type of x_j is $S(b_j)$. We may now define $P(b)$
 607 case-wise:

608 ■ For $b = b_0 \cdot 0^n \cdot 1^i$ with $i \leq n$, then $t|_b = y h_{\sigma(1)} \dots h_{\sigma(k)}$ with $k = n - i$, so that
 609 $\text{fv}(t|_b) = \{y, x_{\sigma(1)}, \dots, x_{\sigma(k)}\}$.

610 We define:

611 $P(b) = y : (2 \cdot S(b)), x_{\sigma(1)} : (2 \cdot S(b_1)), \dots, x_{\sigma(k)} : (2 \cdot S(b_k)) \vdash t|_b : T(b_1) \rightarrow \dots \rightarrow T(b_k) \rightarrow o_b$

612 In particular, when $i = n$, $P(b) = y : (2 \cdot S(b_0)) \vdash y : S(b_0)$.

613 ■ For $b = b_0 \cdot 0^i$ with $i \leq n$, then $t|_b = \lambda x_{i+1} \dots x_n. y h_{\sigma(1)} \dots h_{\sigma(n)}$, so that $\text{fv}(t|_b) =$
 614 $\{y, x_1, \dots, x_i\}$. We define:

615 $P(b) = y : (2 \cdot S(b)), x_1 : (2 \cdot S(b'_1)), \dots, x_i : (2 \cdot S(b'_i)) \vdash t|_b : S(b'_i) \rightarrow \dots \rightarrow S(b'_n) \rightarrow o_b$

616 Note that the definition agree with the previous case when $i = n$. In the case $i = 0$ i.e.,
 617 $b = b_0$, we obtain $y : (2 \cdot S(b_0)) \vdash t|_{b_0} : T(b_0)$ as expected.

618 Thus, $P(b)$ is defined for all $b \in \text{supp}(t)$. We just need to check that P is a correct
 619 derivation. For this, let us use the same notations and observe then that, for $1 \leq i \leq n$
 620 $P(b_i) = x_{\sigma(i)} : (2 \cdot S(b_i)) \vdash h_{\sigma(i)} : T(b_i)$, which proves that every application and abstraction
 621 node has correct premises in the case-wise definition of $P(b)$ above. ◀

622 A.2 Approximating permutator pairs

623 The lemma below is enough to prove Lemma 26:

624 ► **Lemma 35.** *If t is a 001-NF and $P \triangleright x : (2 \cdot S) \vdash t : P$ is approximable, where $(S, T) \in \text{PPP}$,
 625 then, for all $d \in \mathbb{N}$, there is a finite $P_d \leq P$ such that $P_d \triangleright x : (2 \cdot (S)^{\leq d}) \vdash t : (T)^{\leq d}$.*

626 **Proof.** By Theorem 22, t is a hereditary permutator. Assume that $t = \lambda x_1 \dots x_p. x t_1 \dots t_p$
 627 with $x \neq x_i$, $S = (2 \cdot T_{\sigma(1)}) \rightarrow \dots \rightarrow (2 \cdot T_{\sigma(n)}) \rightarrow o$ and $T = (2 \cdot S_1) \rightarrow \dots \rightarrow (2 \cdot S_n) \rightarrow o$.
 628 By syntax-direction of system **S**, P is of the form given by Fig. 1. In particular, we have
 629 unique subderivations P_1, \dots, P_p of P at applicative depth 1, respectively typing t_1, \dots, t_p .
 630 We proceed by induction on d .

23:18 Typing Hereditary Permutators

- Case $d = 0$. We set $S_0 = T_0 = () \rightarrow \dots \rightarrow () \rightarrow o$ (arity n) so that $(S_0, T_0) \in \text{PPP}_0$. Let

$$\begin{array}{c}
 \frac{}{x : (2 \cdot S_0) \vdash s : S_0} \text{ax} \\
 \frac{}{x : (2 \cdot S_0) \vdash x t_1 : () \rightarrow \dots () \rightarrow o \text{ (arity } n-1 \text{)}} \text{app} \\
 \frac{}{} \text{app} \\
 \vdots \\
 \frac{}{x : (2 \cdot S_0) \vdash x t_1 \dots t_p : () \rightarrow \dots () \rightarrow o \text{ (arity } n-p \text{)}} \text{app} \\
 \frac{}{x : u(2 \cdot S_0) \vdash \lambda x_1. x t_1 \dots t_p : () \rightarrow \dots () \rightarrow o \text{ (arity } n-p+1 \text{)}} \text{abs} \\
 \frac{}{} \text{abs} \\
 \vdots \\
 \frac{}{x : (2 \cdot S_0) \vdash \lambda x_1 \dots x_p. x t_1 \dots t_p : T_0} \text{abs}
 \end{array}$$

631 By construction, $P_0 \leq P$.

- Case $d > 0$: by the induction hypothesis, there are $P_i^{d-1} \leq P_i$ concluding with $x_{\sigma(i)} : (2 \cdot (S_{\sigma(i)})^{\leq d-1} \vdash t_i : (T_{\sigma(i)})^{\leq d-1})$. We set:

$$\begin{array}{c}
 \frac{}{x : (2 \cdot (S)^{\leq d}) \vdash x : (S)^{\leq d}} \text{ax} \quad P_1^{d-1} \\
 \frac{}{} \text{app} \\
 \vdots \quad P_p^{d-1} \\
 \frac{}{x : (2 \cdot (S)^{\leq d}), x_1 : (2 \cdot (S_{\sigma(1)})^{\leq d-1}), \dots \vdash x t_1 \dots t_p : (2 \cdot (T_{\sigma(p+1)})^{\leq d-1}) \rightarrow o} \text{app} \\
 \frac{}{} \text{abs} \\
 \vdots \\
 \frac{}{x : (2 \cdot (S)^{\leq d}) \vdash t : (T)^{\leq d}} \text{abs}
 \end{array}$$

632 Since $P_i^{d-1} \leq P_i$ for $1 \leq i \leq p$, we conclude that $P_d \leq P$ as expected.

633

