The Collapse of the Sequential Intersection Type System on the Multiset Intersection Type is Surjective

Pierre Vial

IRIF, Université Paris Diderot
pvial@irif.fr

Abstract

We show that every (finite or not) typing derivation of system $\mathcal{M}$, using non-idempotent intersection, which is the infinitary version of de Carvalho's system $\mathcal{M}_0$, can be represented in a *rigid*, non-idempotent intersection type system $\mathtt{S}$. Namely, whereas non-idempotent intersection is represented by multisets in system $\mathcal{M}$, system $\mathtt{S}$ resort to families of types indexed by integers, called *tracks*. The *rigidity* is here related to the fact that those indexes matter as well as the order in which the types are quoted in a familly of types.

1 Introduction

Type Assignment Systems as HORS

The λ-calculus and its variants are HORS. An important dynamical property of those HORS is *normalization*: a term t is $\mathcal{N}$-normalizing if it can be reduced to a term $t' \in \mathcal{N}$, where $\mathcal{N}$ is a fixed subset of terms (called **$\mathcal{N}$-normal forms (NF)**) that is stable under β-reduction. For instance, if $\mathcal{N}$ is the set of **head normal forms (HNF)**, this definition corresponds to the class of the **head normalizing (HN) terms** and if $\mathcal{N}$ is the set of the (full) normal forms (*i.e.* the terms without redexes), then this definition yields the class of **weakly normalizing (WN) terms** (it is to be noted that strong normalization does not fit in this definition).

Intersection type systems, introduced by Coppo and Dezani [**?**] are used to *statically* characterize classes of normalizing terms inside different calculi. However, most of those type systems enjoy a subject reduction and a subject expansion lemmas, meaning that if a term t reduces to t', then, from a derivation Π typing t, we can produce a derivation typing t' (and conversely, from a derivation Π' typing t', we can produce a derivation Π typing t). Thus, intersection type systems can be themselves seen as HORS.

Subject reduction and expansion lemmas are usually pivotal to establish the equivalence between normalizability and typability, *e.g* if every NF is proven to be typable, then the subject expansion lemma grants that every normalizable term is typable. Thus, it is essential that a type system enjoys good dynamical properties w.r.t. (anti)reduction, in order to prove they can statically characterize dynamical behaviours.

Our work focuses on an infinitary type system $\mathtt{S}$, which, as a infinitary HORS is *deterministic* (see §**??**). We prove here that the expressive power of $\mathtt{S}$ is greater than that of another type non-deterministic type system $\mathcal{M}$, despite $\mathtt{S}$ is very low-level and rudimentary (see § **??**): it is actually easy to see that $\mathtt{S}$ can be collapsed into a subsystem of $\mathcal{M}$ (*c.f.* § **??**). However, we show here that this collapse is full (in the sense that every $\mathcal{M}$-derivation is the collapse of a $\mathtt{S}$-derivation), allowing to use the more fine-grained framework of $\mathtt{S}$ to study $\mathcal{M}$.

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Idempotent and Non-Idempotent Intersections

Intersection types allow to assign a variable (and sometimes a term) several different types, thus yielding a kind of polymorphism (*e.g.* if x receives the types A and $A \to B$ *i.e.* x is assigned the intersection type $A \wedge A \to B$, we can type xx with B).

The original type systems feature *idempotent* intersections: the types $A \wedge B \wedge A$ and $A \wedge B$ are *practically* equal (when they are not already *identified*). De Carvalho [**?**] provided a new characterization of the set of HN terms by means of a type system $\mathcal{M}_0$, which resorted to *non-idempotent* intersection types. This framework allowed to replace Tait's realizability argument [**?**] – used to prove the implication «Typable $\Rightarrow$ Normalizable» – by a considerably simpler, arithmetical one.

Non-idempotency means here that types occur with a multiplicity in an intersection type. Namely, the intersection of types $\sigma_1, \ldots, \sigma_n$ is represented by the *multiset* $[\sigma_1, \ldots, \sigma_n]$: the order of this enumeration does not matter, but the number of occurrences of a type does, *e.g.* we have $[\sigma, \tau, \sigma] = [\sigma, \sigma, \tau] \neq [\sigma, \tau]$. In system $\mathcal{M}_0$ (§**??**), the assignments $x : [\sigma, \tau, \sigma]$ and $x : [\sigma, \tau]$ cannot be interchanged, and the application rule *accumulates* the typing information in the environments: if x is typed $[\sigma_1, \sigma_2, \sigma_3]$, $[\sigma_2, \sigma_4]$ and $[\sigma_4]$ in the premises of an @-rule, then it will be typed $[\sigma_1, \sigma_2, \sigma_2, \sigma_3, \sigma_4, \sigma_4]$ in its conclusion.

Thus, in system $\mathcal{M}_0$, a type can be regarded as a *resource*, which the quantitative argument proving that typability head-normalizability relies on: when a typed redex is reduced inside a derivation Π, it yields a new derivation Π' (typing the reduced term) while strictly decreasing a positive integer measure. It means that, at some point, there are no more typed redexes, so that the reduced is a (partial) NF.

Type Characterization inside an Infinitary Framework

De Carvalho's system $\mathcal{M}_0$ drew in the last a great interest towards quantitative – *resource aware* – type systems and it paved the road for many works [**?**, **?**, **?**], either simplifying previous results or establishing new ones.

In [**?**], we investigated the possibility of a type characterization of weak normalizability in Λ^{001}, an infinitary λ-calculus which was introduced in [**?**]. The calculus Λ^{001} is a infinitary HORS, featuring rewriting sequences of infinite length.

The finitary type system $\mathcal{M}_0$ can be given an infinitary variant $\mathcal{M}$ by taking its rules *coinductively* (instead of *inductively*) and allowing multisets to be infinite. However, we proved $\mathcal{M}$ could allow irrelevant infinite derivation, *e.g.* some non-HN terms are typable in $\mathcal{M}$. This observation suggested to use a validity criterion relying on the notion *approximability* to discard irrelevant proofs. However, we showed that this notion of approximability could not be formulated in $\mathcal{M}$, mainly because it is not possible to distinguish two occurrences of the same type in a multiset. This led us to resort to rigid constructions.

Rigid Non-Idempotent Intersections

In order to be able to characterize the of WN terms in an infinitary λ-calculus, we used an infinitary type assignment system `S` with non-idempotent intersections **??**. System `S` differs from the infinitary version $\mathcal{M}$ of the finitary type system $\mathcal{M}_0$ in that, the intersections are **rigid** in `S`: the *multisets* of types (called **multiset types**) used in $\mathcal{M}$ are (coinductively) replaced by *families* of types indexed by integers (those integers are called **tracks**). Such a family of types is called a **forest type (f.t.)**. For instance, the forest type $(T_k)_{k\in\{2,3,8\}}$, with $T_2 = S$, $T_3 = T$ and $T_8 = S$, is an intersection of two occurrences of type S (placed on

tracks 2 and 8) and one occurrence of type T (placed on track 3). The need for rigidity is explained in the next subsection

To be equal, two $\mathtt{S}$-types need to be *syntactically* equal – let us say informally that the equality is **tight** in $\mathtt{S}$ – *e.g.* $(T_k)_{k\in\{2,3,8\}} \neq (T'_k)_{k\in\{2,3,9\}}$ with $T'_2 = S$, $T'_3 = T$ and $T'_9 = S$, whereas equality between multisets types does not depend of the order of enumeration of its elements or the particular form with which we write the types: let us say the equality between multisets is **loose**. Thus, in system $\mathtt{S}$, types and contexts are very low-level and the @-typing rule can be used only in case of tight equality.

Reduction Choices

As HORS, intersection type systems are usually not *deterministic*: if $t \to t'$ and Π is a derivation typing $t = (\lambda x.r)s$, then, the proof of the subject reduction lemma can yield several derivations Π' typing $t' = r[s/x]$. In that case, we say there are *reduction choices*.

For instance, the type system $\mathcal{M}$ is not deterministic: when we reduce t to t', there are several natural ways to produce a derivation Π'. It is possible as soon as the variable x has been assigned several times the same type σ (see §**??**). In sharp contrast, the use of *tight* equalities in system $\mathtt{S}$ makes there is only one built-in way – under the same hypotheses – to produce a derivation typing t'. Thus, system $\mathtt{S}$ is deterministic. We even say that the unique reduction choice is *trivial*, because, as it will turn out (§), it is based upon an identity isomorphism: roughly speaking, reduction is based on the track equality *e.g.* if there is an axiom leaf typing x using track 8, then it will be substituted by an argument derivation located on track 8 and so on, even when x has been assigned several times the type S (with $S = S_8$).

The Question of Representability

Rigid types, forest types and derivations (of system $\mathtt{S}$) can be naturally collapsed into regular types, multisets types and derivations (of system $\mathcal{M}$). Actually, $\mathcal{M}$-types and multisets types are easily identifiable to equivalence classes of rigid (forest) types. We (coinductively) **collapse** families indexed by integers into multisets. For instance, $(T_k)_{k\in K} \equiv (T'_k)_{k\in K'}$ whenever there is a bijection $\sigma : K \to K'$ (called a **track resetting**) s.t. $T'_{\sigma(k)} = T_k$. With the above examples, $(T_k)_{k\in\{2,3,8\}} \equiv (T'_k)_{k\in\{2,3,9\}}$. The equivalence relation coinductively generated by this base rule allows to see the set of $\mathcal{M}$-types (resp. multiset types) as a quotient of the set of rigid types (resp. forest types). When a rigid type T (resp. forest type F) is collapsed on the $\mathcal{M}$-type τ (resp. the multiset type $[\sigma_i]_{i\in I}$), we say that T (resp. F) is a **parser** of τ (resp. $[\sigma_i]_{i\in I}$).

The @-rule of system $\mathcal{M}$ is based upon a *loose* equality: if, inside a rigid derivation P (of system $\mathtt{S}$), we collapse every (forest) type and apply the same recipe, we obtain a $\mathcal{M}$-derivation Π. However, it is not clear that, starting from a $\mathcal{M}$-derivation Π, we can find a rigid P that collapses into Π. For instance, it would demand that we can choose a good parser for every type introduced in an axiom rule, so that we have a (tight) equality in all the @-rules. Since Π can be infinite in depth or in width and the typing constraints propagate in complicated ways inside the derivation, the possibility of such a good choice is not easily granted.

Moreover, another feature of $\mathtt{S}$ can seem limited: in contrast to system $\mathcal{M}$, the substitutions inside an $\mathtt{S}$-derivation are performed *deterministically*, while we reduce the judged term. This can been seen as restrictive, because, even when there are several occurrences of the same type, substitution can be processed only in one way in system $\mathtt{S}$. It can be seen

as a restriction compared to system $\mathcal{M}$. So it raises the following question: can we built a rigid representative P of an $\mathcal{M}$-derivation Π w.r.t. any reduction choice we would have done "by-hand" ? If we perform a reduction choice at each step of a reduction sequence, we speak of **reduction choices sequence**.

1.1 Contributions

In this article, the show that:

- Any *quantitative* derivation Π, approximable or not, has a rigid quantitative representative P.
- Any reduction choices sequence of length $\leqslant \omega$ can be built-in inside such a representative P, without assuming this reduction sequence to be sound (*strongly converging*, [**?**])

We proceed this way: we represent every quantitative $\mathcal{M}$-derivation Π by means of an *hybrid* derivation P_h (in a new type system $\mathtt{S_h}$) in which the tight equality (in the @-rule) is loosened and replaced by a congruence. Next, we endow those hybrid derivations with rigid, deterministic reduction choices (to be called *interfaces*), yielding *operable* derivations (in another type system $\mathtt{S_o}$). We show then that every "by-hand" reduction sequence of (possible) infinite length can be encoded in an interface. The *trivial* derivations are the operable derivation (system $\mathtt{S}$) in which the interface uses only identity isomorphisms. Finally, we prove that every operable derivation is *isomorphic* to a trivial derivation. This result concludes the proof of the Representation Theorem, stating that our non-idempotent, rigid intersection type system $\mathtt{S}$ has more expressive power than the system $\mathcal{M}$.

The most difficult point is the last one, *i.e.* establishing every $\mathcal{M}$-derivation has a *trivial* $\mathtt{S}$-representative. In a finitary framework, this could be possible by studying first the derivations typing a NF (for which representation is granted [**?**]), and then proceeding by subject expansion. However, as we have hinted at, typability in system $\mathcal{M}$ does not imply any kind of normalization (some non-HN terms can be typed). So we will resort to *ad hoc* notions of *referent bipositions*, *syntactic polarity* and *collapsing strategy* (which is a partial reduction strategy) to reach our goal.

1.2 Outline

In Section **??**, we define the set of infinitary terms Λ^{001} and the types. Next, in § **??**, we define the rigid system $\mathtt{S}$ and its weaker variant $\mathtt{S_h}$, as well as an infinitary variant $\mathcal{M}$ of system $\mathcal{M}_0$. Types and multiset types of $\mathcal{M}$ are presented as equivalence classes of (forest) types of $\mathtt{S}$. The subject reduction property is studied in § **??** and we show a "Representation Lemma" (§ **??**): every reduction sequence is representable through a suitable interface. We discuss operable derivations isomorphisms and track resettings in § **??**. A trivial theory associated to an operable derivation is introduced from § **??** to § **??**. It exhibit how we should reset an operable derivation into a trivial one. In the last section, we prove the representation theorem: every operable derivation is isomorphic to a trivial derivation.

2 Parsing Terms and Types

2.1 Conventions on Labelled Trees and Forests

We write $\mathbb{N}^*$ for the set of finite sequences of integers: ε is the empty sequence and $a \cdot a'$ the concatenation of a and a', $a \leqslant a'$ means that $\exists a"$ s.t. $a \cdot a" = a'$ (prefix order). Elements of $\mathbb{N}^*$ are seen as **positions** and natural integers are called **tracks**. An integer $\geqslant 2$ is called an

argument track. The track 0 is dedicated to abstraction, the track 1 to the left part of an application and argument tracks to the right part of an application. If $a = a_0 \cdot \ldots \cdot a_{\ell-1} \in \mathbb{N}$ (where the a_i are integers), ℓ is the **length** of a and is written $|a|$ and the **applicative depth** of a is $\mathrm{ad}(a) = |\{0 \leqslant i \leqslant \ell - 1 \mid a_i \geqslant 2\}|$. When $\ell \neq 0$, we set $\mathrm{last}(a) = a_{\ell-1}$.

We will now introduce the notions of **(labelled) trees and forests**, which we will often encounter: a tree A is a non-empty part of $\mathbb{N}^*$ that is downward-closed for the prefix order (for all $a,\ a' \in \mathbb{N}^*$, $a \in A$ and $a' \leqslant a$ implies $a \in A$). A labelled tree T is a function from $\mathbb{N}^*$ to a set Σ (called **signature**) s.t. the **support** of T – i.e. its domain –, written $\mathtt{supp}(T)$, is a tree.

- If $a \in \mathtt{supp}(T)$, $T(a)$ is the **label** of T at position a.
- If $a \in \mathtt{supp}(T)$, $T|_a$ is the **subtree** of T whose root is at position a, that is, $\mathtt{supp}(T|_a) = \{c \in \mathbb{N}^* \mid a \cdot c \in \mathtt{supp}(T)\}$ and $T|_a(c) = T(a \cdot c)$.

It A is a tree and $a \in A$, then we set $\mathrm{Arg}_A(a) = \{\ell \geqslant 2 \mid a \cdot \ell \in A\}$: $\mathrm{Arg}_A(a)$ is the set of argument tracks originating from a in A.

A **(labelled) forest** F on the signature Σ is a function F from $A - \{\varepsilon\}$ to Σ, where A is a tree. The set $A - \{\varepsilon\}$ is also called the support of F. The sets of **roots** of F is $\mathrm{Rt}(F) = \{k \in \mathbb{N} \mid k \in \mathtt{supp}(F)\}$ (for practical reasons, we will always assume that $k \in \mathrm{Rt}(F)$ implies $k \geqslant 2$). Thus, for all $k \in \mathrm{Rt}(F)$, $F|_k$ is a tree. It is usually convenient to write a position of the domain of a labelled forest F as $k \cdot c$, where $k \in \mathrm{Rt}(F)$ and $c \in \mathtt{supp}(F|_k)$. A family $(F^i)_{i\in I}$ of labelled forests is said to be **disjoint** if the $\mathrm{Rt}(F^i)$ are pairwise disjoint. In that case, we define its **join** as the unique labelled forest F s.t. $\mathtt{supp}(F) = \bigcup_{i\in I} \mathtt{supp}(F^i)$ and $F(k \cdot a) = F^i(k \cdot a)$, where i is the unique index s.t. $(k \cdot a) \in \mathtt{supp}(F^i)$. If $(F^i)_{i\in I}$ is not disjoint, we say there is a **track conflict**.

The family $(T_k)_{k\in K}$ (where T_k is a labelled tree for all $k \in K$) will (abusively) denote the labelled forest F s.t. $\mathrm{Rt}(F) = K$ and $F|_k = T_k$.

In quantitative type systems, the arguments of an application occur with a multiplicity. That is why each track ≥ 2 will stand for one occurrence of an argument. It leads to define the **collapse** $\overline{a}$ of any $a \in \mathbb{N}^*$ by $\overline{a} = \min(a_0,\, 2) \cdot \min(a_1,\, 2) \cdot \ldots \cdot \min(a_{\ell-1},\ \ 2)$ with $\ell = |a|$. Thus, $\overline{a} \in \{0,\, 1,\, 2\}^*$. If $A \subset \mathbb{N}^*$, we set $\overline{A} = \{\overline{a} \mid a \in A\}$. If $b \in \{0,\, 1,\, 2\}^*$, a **representative** of b is a $a \in \mathbb{N}^*$such that $\overline{a} = b$. If A is a tree of $\mathbb{N}^*$, we set $\mathrm{Rep}_A(b) = \{a \in A \mid \overline{a} = b\}$. We will often simply write $\mathrm{Rep}(b)$. The tracks 0 and 1 have a specific role, but the arguments tracks $\geqslant 2$ can be permuted. It motivates:

▶ **Definition 1.** Let U_1 and U_2 be two (labelled) trees or forests.
A **01-stable (labelled) tree or forest isomorphism** ϕ from U_1 to U_2 is a bijection from $\mathtt{supp}(U_1)$ towards $\mathtt{supp}(U_2)$ such that:

- ϕ is monotonic for the prefix order.
- If $a \cdot k \in \mathtt{supp}(U_1)$ with $a \in \mathbb{N}^*$ and $k = 0,\, 1$, then $\phi(a \cdot k) = \phi(a) \cdot k$.
- In the labelled case: for all $a \in \mathtt{supp}(U_1)$, $U_2(\phi(a)) = U_1(a)$.

If U is a (labelled) tree or forest and ϕ is a monotonic, length-preserving injection from $\mathtt{supp}(U)$ to $\mathbb{N}^*$ s.t. $\phi(a \cdot k) = \phi(a) \cdot k$ whenever $k = 0,\, 1$, we write $\phi(U)$ for the *unique* (labelled) tree or forest s.t. $\mathtt{supp}(\phi(U)) = \{\phi(a) \mid a \in \mathtt{supp}(U)\}$ and $\phi(U)(a') = U(\phi^{-1}(a'))$.

2.2 Infinitary Terms

Let $\mathcal{V}$ be a countable set of term variables. The set of terms Λ^{111} is defined *coinductively*:

$$t,\, u \ ::= \ x \parallel \lambda x.t \parallel tu$$

The parsing tree of $t \in \Lambda^{111}$, also written t, is the labelled tree on $\Sigma_t := \mathcal{V} \cup \{\lambda x \mid x \in \mathcal{V}\} \cup \{@\}$ defined coinductively by: $\mathtt{supp}(x) = \{\varepsilon\}$ and $x(\varepsilon) = x$, $\mathtt{supp}(\lambda x.t) = \{\varepsilon\} \cup 0 \cdot \mathtt{supp}(t)$, $(\lambda x.t)(\varepsilon) = \lambda x$ and $(\lambda x.t)(0 \cdot b) = t(b)$, $\mathtt{supp}(tu) = \{\varepsilon\} \cup 1 \cdot \mathtt{supp}(t) \cup 2 \cdot \mathtt{supp}(u)$, $(tu)(\varepsilon) = @$, $(tu)(1 \cdot b) = t(b)$ and $(tu)(2 \cdot b) = u(b)$.

The abstraction λx binds x in $\lambda x.t$, and α-equivalence and substitution of a variable x by a term s inside t can be defined coinductively.

The relation $t \xrightarrow{b} t'$ is defined by induction on $b \in \{0, 1, 2\}^*$: $(\lambda x.r)s \xrightarrow{\varepsilon} r[s/x]$, $\lambda x.t \xrightarrow{0 \cdot b} \lambda x.t'$ if $t \xrightarrow{b} t'$, $t_1 t_2 \xrightarrow{1 \cdot b} t_1' t_2$ if $t_1 \xrightarrow{b} t_1$, $t_1 t_2 \xrightarrow{2 \cdot b} t_1 t_2'$ if $t_2 \xrightarrow{b} t_2'$. We define **β-reduction** by $\rightarrow = \bigcup_{b \in \{0,1,2\}^*} \xrightarrow{b}$

The calculus Λ^{001} is the set of terms $t \in \Lambda^{111}$ such that, for every infinite branch b of t, $\lim \mathrm{ad}(b|_n) = \infty$, where $b|_n$ is the length n prefix of b. Thus, for 001-terms, infinity is allowed, provided we descend infinitely many times in application arguments.

2.3 Rigid Types

Let $\mathcal{X}$ be a countable set of types variables (denoted α). The sets of rigid types Types^+ noted T and rigid forest types FTypes^+ noted F are are coinductively defined by:

$$\begin{array}{rcl} T & ::= & \alpha \parallel F \to T \\ F & ::= & (T_k)_{k \in K} \end{array}$$

▶ **Remark 1.** K must be a set of integers $\geqslant 2$. Thus, forest types, that represent the *intersection* of the types they hold, are lists of types with holes.

Their parsing trees and forests, also written T and F, are defined coinductively: $\mathtt{supp}(\alpha) = \varepsilon$ and $\alpha(\varepsilon) = \alpha$, if $F = (T_k)_{i \in K}$, then $\mathtt{supp}(F) = \bigcup_{k \in K} k \cdot \mathtt{supp}(T_k)$ and $F(k \cdot c) = T_k(c)$, $\mathtt{supp}(F \to T) = \mathtt{supp}(F) \cup 1 \cdot \mathtt{supp}(T) \cup \{\varepsilon\}$ and $(F \to T)(\varepsilon) = \to$, $(F \to T)(k \cdot c) = F(k \cdot c)$ if $k \geqslant 2$ and $(F \to T)(1 \cdot c) = T(c)$. Compared to the types of $\mathcal{M}_0$, our (forest) types are low-level objects. That is why they do have a parsing tree, contrary to the types of $\mathcal{M}_0$ (once again because multisets are unable to distinguish two occurrences of the same type).

The sets Types and FTypes are the subsets of Types^+ and FTypes^+ whose elements do not hold an infinite branch ending by 1^ω. A rigid (forest) type is said to be finite when its support is. We write $(\ )$ for the forest type whose support is empty and, if T is a type and $k \geqslant 2$, $(T)_k$ denotes the f.t. F s.t. $\mathrm{Rt}(F) = \{k\}$ and $F|_k = T$.

▶ **Definition 2.** Let U_1 and U_2 be two (forest) types. A **(forest) type-isomorphism ((f.)t.-isomorphism)** from U_1 to U_2 is a 01-stable labelled isomorphism from U_1 to U_2

We write $U_1 \equiv U_2$ when there is such an isomorphism.

- If $(F^i)_{i \in I}$ is a disjoint (§ **??**) family of forest types, its join is also a forest type.
- If $S = F \to T$, where F is a forest type and T a type, we write $\mathrm{Tl}(S)$ for the f.t. F and $\mathrm{Hd}(S)$ for the type T.
- If $\psi : F_1 \longrightarrow F_2$ is a f.t. isomorphism and $\phi : T_1 \to T_2$ is a type isomorphism, then $\psi \to \phi : (F_1 \to T_1) \to (F_2 \to T_2)$ is the type isomorphism defined by: $(\psi \to \phi)(k \cdot c) = \phi(k \cdot c)$ when $k \geqslant 2$ and $(\psi \to \phi)(1 \cdot c) = 1 \cdot \phi(c)$. If F is a f.t., then $F \to \phi$ denotes $\mathtt{id}_F \to \phi$.
- Conversely, if $\phi : (F_1 \to T_1) \longrightarrow (F_2 \to T_2)$ is a type isomorphism, then $\mathrm{Hd}(\phi)$ and $\mathrm{Tl}(\phi)$ are resp. the type isomorphism from T_1 to T_2 and the f.t. isomorphism from F_1 to F_2 induced by ϕ. Thus, $\phi = \mathrm{Tl}(\phi) \to \mathrm{Hd}(\phi)$ and $\mathrm{Hd}(\psi \to \phi) = \phi$ and $\mathrm{Tl}(\psi \to \phi) = \phi$.

3 Some Infinitary Type Systems with Non-Idempotent Intersection

3.1 Rigid Derivations (Hybrid and Trivial)

A **(rigid) context** C is a function from $\mathcal{V}$ to FTypes. If $x \in \mathcal{V}$, the context $C - x$ is defined by $(C-x)(y) = C(y)$ for any $y \neq x$ and $(C-x)(x) = (\ )$. We define the join of contexts point-wise. A **judgment** is a sequent of the form $C \vdash t : T$, where C is a context, t a 001-term and $T \in$ Types. The set HDeriv of **hybrid derivations (h.d.)** (denoted P) is defined coinductively by the following rules:

$$\frac{}{x : (T)_k \vdash x : T \ (\text{pos. } \varepsilon)}\ \texttt{ax} \qquad \frac{\dfrac{P'}{C \vdash t : T \ (\text{pos. } 0)}}{C - x \vdash \lambda x.t : \ C(x) \to T \ (\text{pos. } \varepsilon)}\ \texttt{abs}$$

$$\frac{\dfrac{P'}{C \vdash t : \ (S^{\mathrm{L}}_k)^{k\in K(\mathrm{L})} \to T \ (\text{pos. } 1)} \qquad \left(\dfrac{P'_k}{D_k \vdash u : \ S^{\mathrm{R}}_k \ \ (\text{pos. } k)}\right)_{k\in K(\mathrm{R})}}{C \cup \bigcup\limits_{k\in K(\mathrm{R})} D_k \vdash t(u) : T \ (\text{pos. } \varepsilon)}\ \texttt{happ}$$

▶ **Remark 2.** Positions constraints are indicated between brackets, *e.g.* if the root is an abstraction, we must have $0 \in \mathtt{supp}(P)$.
- In the application rule, $K(\mathrm{L})$ and $K(\mathrm{R})$ are two sets of integers greater or equal than 2 such that $(S^{\mathrm{L}}_k)_{k\in K(\mathrm{L})} \equiv (S^{\mathrm{R}}_k)_{k\in K(\mathrm{L})}$.
- In the application rule, the contexts must be disjoint, so that no track conflict occurs.

We write $P \rhd C \vdash t : T$ to mean that the conclusion of the hybrid derivation P is the judgment $C \vdash t : T$.

Let P be a h.d. and $A = \mathtt{supp}(P)$. We set $A[@] = \{a \in \mathbb{N}^* \mid a \cdot 1 \in A\}$ (thus, for $a \in A$, $a \in A[@]$ iff $t(\overline{a}) = @$).

For $a \in A$, we set $\mathrm{L}_P(a) = \mathrm{Tl}(T(a \cdot 1))$ and $\mathrm{R}_P(a) = (T_{a\cdot\ell})_{\ell\in \mathrm{Arg}(a)}$. We often write those f.t. simply $\mathrm{R}(a)$ and $\mathrm{L}(a)$.The condition of the `happ` rule is equivalent to $\mathrm{L}_P(a) \equiv \mathrm{R}_(\ a)$.

The set Deriv of **trivial derivations (t.d.)** is coinductively defined like HDeriv, except we use the rule `app` below instead of rule `happ`:

$$\frac{\dfrac{P'}{C \vdash t : \ (S_k)_{k\in K} \to T \ (\text{pos. } 1)} \qquad \left(\dfrac{P'_k}{D_k \vdash u : \ S_k \ \ (\text{pos. } k)}\right)_{k\in K}}{C \cup \bigcup\limits^{k\in K} D_k \vdash t(u) : T \ (\text{pos. } \varepsilon)}\ \texttt{app}$$

▶ **Remark 3.** In the application rule, K is a set of integers greater or equal than 2.
- In the application rule, the contexts must be disjoint, so that no track conflict occurs.
- Thus, a t.d. is a h.d. P such that, for all $a \in \mathtt{supp}(P)[@]$, $\mathrm{L}(a) = \mathrm{R}(a)$.

3.2 Positions, Bipositions, Tracks and Quantitativity

Thanks to rigidity, we can designate, identify and name every part of a derivation, thus allowing to formulate many associate, useful notions.

Let P be a hybrid derivation. We write $C(a) \vdash t|_{\overline{a}} : T(a)$ for $P(a)$, when $a \in \mathtt{supp}(P)$. When $a \in \mathtt{supp}(P)$, then a is an **outer position** of P. If $a \in \mathtt{supp}(P)$ and $c \in \mathtt{supp}(T(a))$

(resp. $k \cdot c \in C(a)(x)$), we say the triple $(a,\, c)$ (resp. the triple $(a,\, x,\, k \cdot c)$) is a right (resp. left) **biposition**. In that case, c (resp. $(x,\, k \cdot c)$) is a right (resp. left) **inner position** of $P(a)$. The set of the bipositions of P is called its **bisupport**, and is written $\mathrm{bisupp}(P)$. We can see a derivation P typing a given term t as a function from its bisupport B towards $\mathcal{X} \cup \{\rightarrow\}$. Thus, $P(a,\, c) = T(a)(c)$ and $P(a,\, x,\, c) = C(a)(x)(c)$.

If $a \in A$ and $x \in \mathcal{V}$, we set $\mathrm{Ax}(a)(x) = \{a' \in A \mid a' \geqslant a \text{ and } t(\overline{a'}) = x\}$ if x is free at position a and $\mathrm{Ax}(a)(x) = \emptyset$ if not: it is the set of (positions of) axiom leaves typing x above a.

We distinguish likewise 3 kinds of tracks: if $a \cdot k \in \mathtt{supp}(P)$, then k is an **outer track** a position a, if $(a,\, c \cdot k) \in \mathrm{bisupp}(P)$, k is a **(right) inner track** at biposition $(a,\, c)$, if $(a,\, x,\, \ell \cdot c \cdot k) \in \mathrm{bisupp}(P)$ is a **(left) inner track** at biposition $(a,\, x,\, \ell \cdot c)$. If $(a,\, x,\, k) \in \mathrm{bisupp}(P)$, then k is an **axiom track** at position a (w.r.t. variable x). If $A = \mathtt{supp}(P)$ and $a \in A$, we set $\mathrm{OutTr}(a) = \{\ell \in \mathbb{N} \mid a \cdot \ell \in\}$ and $\mathrm{Arg}(a) = \mathrm{Arg}_A(a) = \{\ell \geqslant 2 \mid a \cdot \ell \in A\}$.

If $a \in A$ is an axiom leaf (*i.e.* $t(\overline{a}) = x$ for some $x \in \mathcal{V}$), we write $\mathrm{tr}(a)$ (or $\lfloor a \rfloor$) for the unique $k \geqslant 2$ such that $P(a)$ is $x : (T(a))_k \vdash x : T(a)$.

3.3 An Infinitary Type System with Multiset Constructions

We present here a definition of type assignment system $\mathcal{M}$, which is an infinitary version of De Carvalho's system $\mathcal{M}_0$.

It is usual in mathematics to resort to quotient sets $X/\equiv$ (where $\equiv$ is an equivalence relation) to "capture" intuitive objects (*e.g.* $\mathbb{Z}$ can be constructed as a quotient set of $\mathbb{N} \times \mathbb{N}$) and to define operations on $X/\equiv$ by first, defining operation on X, next, proving that those operations are compatible with $\equiv$ (*e.g.* the sum of $\mathbb{Z}$ is the quotient of the product sum on $\mathbb{N} \times \mathbb{N}$). For instance, if C is a countable set and $X := \mathtt{S}(\mathtt{A})$ is the set of **sequences** on C *i.e.* functions from $\{\ell \geqslant 2 \mid \ell \in \mathbb{N}\}$ to C, then $\mathcal{M}(C)$, the set of countable multisets on C is $X/\equiv$, where the relation $\equiv$ is defined by $(x_k)_{k \in K} \equiv (y_\ell)_{\ell \in L}$ iff there is a bijection $\sigma : K \rightarrow K$ s.t. $y_{\sigma(k)} = x_k$. Relation $x \equiv y$ means that the sequence y can be obtained from the sequence x by injectively **resetting** the values of the indexes used in x.

System $\mathcal{M}$ has to feature infinitary nesting of multisets (*unforgetful* typing of NF requires so []). It is better to grant that those infinitarily nested multisets can be formalized and that we can define countable sum on multisets. We do this below.

If two (forest) types U_1 and U_2 are isomorphic, we write $U_1 \equiv U_2$. The set $\mathrm{Types}_{\mathcal{M}}$ is the set $\mathrm{Types}/\equiv$ and the set $\mathcal{M}(\mathrm{Types})$ of multiset types is defined as $\mathrm{FTypes}/\equiv$.

If U is a forest or a rigid type, its equivalence class is written $\overline{U}$. The equivalent class of a forest type F is the multiset type written $[\overline{F_{|k}}]_{k \in \mathrm{Rt}(F)}$ and the one of the rigid type $F \rightarrow T$ is the type $\overline{F} \rightarrow \overline{T}$. If α is a type variable, $\overline{\alpha}$ is written simply α (instead of $\{\alpha\}$). It defines coinductively the multiset style writing of $\overline{U}$. If $\overline{U} = \upsilon$, we say υ is the **collapse** of U and that U is a **parser** of υ.

We define the **countable sum** of two multiset types: let $(F^i)_{i \in I}$ and $(G^j)_{j \in J}$ be two countable families of f.t. representing the same family of multiset types, modulo permutation, *i.e.* there is a bijection $\phi : I \rightarrow J$ s.t. $F_i \equiv G_{\phi(i)}$.

Let us write, for all $i \in I$, $F_i = (T_{i,k})_{k \in K(i)}$ and for all $j \in J$, $G_j = (U_{j,\ell})_{\ell \in L(j)}$. Thus, for all $i \in I$, when $j = \phi(i)$, we have $[\overline{T_{i,k}}]_{k \in K[i]} = [\overline{U}_{j,\ell}]_{\ell \in L(j)}$.

There is a bijection $\psi : \bigcup_{i \in I} K(i) \times \{i\} \rightarrow \bigcup_{j \in J} L(j) \times \{j\}$, inducing ϕ on its 2nd projection, s.t., for all $i \in I$ and $k \in K(i)$, $T_{i,k} \equiv U_{\psi(k,i)}$.

Since I and J are countable, let $f : I\times(\mathbb{N}-\{0,1\}) \to \mathbb{N}-\{0,1\}$ and $g : I\times(\mathbb{N}-\{0,1\}) \to \mathbb{N}-\{0,1\}$ be two injections. Let F and G be the two f.t. defined by $\mathrm{Rt}(F) = \{f(i,k) \mid i \in I,\ k \in K(i)\}$ $\mathrm{Rt}(G) = \{g(j,k) \mid j \in J,\ \ell \in L(j)\}$ and $F|_{f(i,k)} = T_{i,k}$, $G|_{g(j,\ell)} = U_{j,\ell}$. Then $F \equiv G$ (we can define a f.t. isomorphism $F \to G$ using f, g and ψ). It allows to write, without ambiguity, $\overline{F} = \sum_{i\in I}[\overline{T_{i,k}}]_{k\in K(i)}$.

A $\mathcal{M}$-context is a function from the set of term variables $\mathcal{V}$ to the set $\mathrm{Types}_{\mathcal{M}}$. The set of $*$-derivations, written Deriv_* is defined coinductively by the following rules:

$$\frac{}{x : [\tau] \vdash x : \tau \ (\text{pos. } \varepsilon)}\ \text{ax} \qquad \frac{\dfrac{P'}{\Gamma \vdash t : \tau \ (\text{pos. } 0)}}{\Gamma - x \vdash \lambda x.t : \ \Gamma(x) \to \tau \ (\text{pos. } \varepsilon)}\ \text{abs}$$

$$\frac{\dfrac{P'}{\Gamma \vdash t : [\sigma_i]_{i\in I} \to \tau \ (\text{pos. } 1)} \qquad \left(\dfrac{P'_k}{\Delta_i \vdash u : \sigma_i \ \ (\text{pos. } k_i)}\right)_{i\in I}}{\Gamma + \sum_{i\in I}\Delta_i \vdash t(u) : \tau \ (\text{pos. } \varepsilon)}\ \text{app}$$

In the app rule, the k_i must be pairwise distinct integers $\geqslant 2$.

Let P_1 and P_2 be two $*$-derivations. A $*$-isomorphism from P_1 to P_2 is a 01-labelled isomorphism from P_1 to P_2 and the set MDeriv is defined by $\mathrm{Deriv}_* / \equiv$.

A hybrid derivation P (with the usual notations C, t, T) **represents** a derivation Π if the $*$-derivation P^* defined by $\mathtt{supp}(P^*) = \mathtt{supp}(P)$ and $P^*(a) = \overline{C(a)} \vdash t|_{\overline{a}} : \overline{T(a)}$, is a representative of Π. We write $P_1 \overset{\mathcal{M}}{\equiv} P_2$ when P_1 and P_2 both represent the same $\mathcal{M}$-derivation Π. The $\mathcal{M}$-derivation $\overline{P^*}$ is often written simply $\overline{P}$.

3.4 Quantitativity and Coinduction

Let Γ be a $\mathcal{M}$-context and f^ω the 001-term defined coinductively by $f^\omega = f(f^\omega)$, which is a Böhm tree. We write $[\alpha]_\omega$ for the multiset type contain infinitely many occurrences of α (thus, $[\alpha]_\omega = [\alpha]_\omega + [\alpha]$). Using the infinite branch of f^ω, the following $\mathcal{M}$-derivation Π'_Γ is correct:

$$\Pi'_\Gamma = \frac{\dfrac{}{f : [[\alpha]\to\alpha] \vdash f^\omega : [\alpha]\to\alpha}\ \text{ax} \qquad \dfrac{\Pi'_\Gamma}{f : [[\alpha]\to\alpha]_{n\in\omega} + \Gamma \vdash f^\omega : \alpha}}{f : [[\alpha]\to\alpha]_{n\in\omega} + \Gamma \vdash f^\omega : \alpha}\ \text{app}$$

If, for instance, we choose the context Γ to be $x : \tau$, from a quantitative point of view (that intuitively forbids weakenings), the variable x (that is not in the typed term f^ω) should not be present in the context. We have been able to "call" the type τ using coinduction through an infinite branch., which is a rule we do not want in quantitive type systems. Thus, we can enrich the type of any variable that is free in any part of a derivation, as long it is below an infinite branch. It motivates the following definition:

▶ **Definition 3.** A hybrid derivation P is **quantitative** if, for all $a \in \mathtt{supp}(P)$ and $V \in \mathcal{V}$, $C(a)(x) = ((T(a))_{\mathrm{tr}(a_0)})_{a_0\in\mathrm{Ax}(a)(x)}$.

- A $*$-derivation P is quantitative if, for all $a \in \mathtt{supp}(P)$ and $x \in \mathcal{V}$, $\Gamma(a)(x) = [\tau(a')]_{a'\in\mathrm{Ax}(a)(x)}$.
- A $\mathcal{M}$-derivation Π is quantitative if any of its $*$-representatives is (in that case, all of them are quantitative).

Thus, when a derivation is quantitative, the knowledge of the types given at axiom rules and of the outer support is enough to reconstruct it.

▶ **Proposition 1.** If a hybrid derivation P is quantitative, then the $\mathcal{M}$-derivation $\overline{P^*}$ is quantitative.

However, we have shown in [], appendix ???, that there were $\mathcal{M}$-derivations Π which have both quantitative and unquantitative hybrid representatives in the rigid framework. It means that quantitativity is a notion that is not well-fit to system $\mathcal{M}$.

Now, assume P is a **quantitative hybrid derivation (q.h.d.)**. For all $a \in A$ and $x \in \mathcal{V}$, we set $\mathrm{AxTr}(a)(x) = \mathrm{Rt}(C(a)(x))$. For all $a \in A$, $x \in \mathcal{V}$ and $k \in \mathrm{AxTr}(a)(x)$, we write $\mathrm{pos}(a, x, k)$ for the unique position $a' \in \mathrm{Ax}(a)(x)$ such that $\mathrm{tr}(a') = k$.

3.5 The Hybrid Construction

Let Π be a quantitative $\mathcal{M}$-derivation. We show here that Π has a hybrid, quantitative representative P.

Let $\tilde{P}$ be a quantitative $*$-derivation representing $\tilde{P}$ and $A = \mathtt{supp}(\tilde{P})$. For each $a \in \mathrm{Ax}$, we choose an integer $\mathrm{tr}(a)$ greater $\geqslant 2$ s.t. no conflict arises (that is, if x is not bound at position a, there are no a', a" $\in \mathrm{Ax}(a)(x)$ s.t. $a' \neq a$" and $\mathrm{tr}(a') = \mathrm{tr}(a$")).

For each axiom leaf $a \in A$ of Π, we choose $T_{\mathrm{ax}}(a)$, a type representing $\tau(a)$.

We write $C(a)(x)$ for the f.t. $(\mathrm{tr}(a_0) \cdot T_{\mathrm{ax}}(a_0))_{a_0 \in \mathrm{Ax}(a)(x)}$. and we choose for each $a \in A$ a representative $T(a)$ of $\tau(a)$, by a $\searrow$-induction.

- If a is an axiom leaf, we set $T(a) = T_{\mathrm{ax}}(a)$.
- If $a \cdot 0 \in A$, then $t(a) = \lambda x$ and we set $T(a) = C(a \cdot 0)(x) \to T(a \cdot 0)$.
- If $a \cdot 1 \in A$, then we set $T(a) = \mathrm{Hd}(T(a \cdot 1))$.

The above induction and the quantitativity of $\tilde{P}$ show that the definition is sound ($\overline{T(a)} = \tau(a)$ for all $a \in A$) and that we have $\overline{P^*} = \Pi$. We call this process the **hybrid construction**.

▶ **Remark 4.** The hybrid construction deeply differs from the trivial construction [] (useto show every normal term is typable) in that, the trivial construction is only applicable for NF (or Böhm trees) and does not involve or need any $\mathcal{M}$-derivation.

4 Subject Reduction

In this section, we consider *(root)-interfaces i.e.* we endow hybrid derivations with partial or complete descriptions of how to perform substitution when we reduce a redex inside the judged term (those interface can encode very different dynamical behaviours). Next (§ **??** to **??**), we show interfaces actually provide a sound way to produce a derivation. In § **??**, we show that every intuitive *reduction choice sequence* can be built-in inside an interface, yielding the Representation Lemma.

While working with reduction, the following assumptions are made: $t|_b = (\lambda x.r)s$, $t \xrightarrow{b} t'$ (with $b \in \{0, 1, 2\}^*$), P is a q.h.d. typing t (with the notations C, T, pos, tr defined in § **??**), whose support is A. In that case, for all $a \in \mathrm{Rep}_A(b)$, we set $\mathrm{RedTr}(a) = \mathrm{AxTr}(a \cdot 10, x)$ and for all $k \in \mathrm{RedTr}(a)$, we write a_k for the unique postfix s.t. $a \cdot 10 \cdot a_k = \mathrm{pos}(a \cdot 10, x\, k)$.

4.1 Interfaces

We consider here a hybrid derivation P, with the usual associated notations (C, T, pos…) Let $a \in A[@]$. By typing constraints, there is a f.t. isomorphism ϕ from $\mathrm{L}(a)$ to $\mathrm{R}(a)$. We call such a ϕ a **interface isomorphism at pos.** a. For all such a, we write $\mathrm{Inter}(a)$ for the set of interface isomorphisms at pos. a.

A bijection ρ from $\mathrm{Rt}(\mathrm{L}(a))$ to $\mathrm{Arg}(a)$ s.t. for all $k \in \mathrm{Rt}(\mathrm{L}(a))$, $T(a{\cdot}1)|_k \equiv T(a{\cdot}\rho(k))$ (*i.e.* $\mathrm{L}(a)|_k \equiv \mathrm{R}(a)|_{\rho(k)}$) is called a **root-interface isomorphism** Thus, a root-interface interface is a function (between root tracks) that can be extended to an interface isomorphism.

If $b \in \{0, 1, 2\}^*$ is the position of an application in t, a family $(\phi_a)_{\mathrm{Rep}(b)}$ of interface isomorphisms (resp. a familly of $(\rho_a)_{a\in\mathrm{Rep}(b)}$ of root-interface isomorphisms) for each $a \in A$ s.t. $\overline{a} = b$ is called a **interface at position** b (resp. a **root-interface at position** b).

The intuitive meaning of a root-interface is the following: assume $t|_b = (\lambda x.r)s$ and so on. Then, for all $a \in \mathrm{Rep}_A(b)$, we have $T(a \cdot 10 \cdot a_k) \equiv T(a \cdot \rho_a(k))$. Thus, when we fire the redex at position b, the root-interface suggests we should replace the axiom leaf at position $a \cdot 10 \cdot a_k$ (using track k) by the argument subderivation at position $a \cdot \rho_a(k)$ (using track $\rho_a(k)$). This idea is developped further and proved to be sound in the coming subsections.

When $t|_b$ is a redex, a root-interface at position b is the minimal piece of information we need to perform subject-reduction (it enables to define the support A' of the reduced derivation and the types used in its axiom rules). If we have actually an interface at position b, we can define the residuals of most right bipositions of P inside the reduced h.d. and even, as we will see, useful between sets of interfaces of P and P'.

A family (ϕ_a) of interface isomorphisms for each application node in A is called a **(complete) interface**. We shall usually write ρ_a for $\phi_{a|\,\mathrm{Rt}}$.

The application rule `happ` (§**??**) precisely requires that, for all $a \in A[@]$, there is an interface isomorphism from $\mathrm{L}(a)$ to $\mathrm{R}(a)$, so every h.d. can be endowed with a complete interface.

An **operable (hybrid) derivation (o.d.)** is a hybrid derivation endowed with a complete interface. Thus, when a derivation P is trivial, it can naturally be endowed with the interface s.t. every interface isomorphism is the identity (a choice we implicitly make).

4.2 Residual Derivation

Let us assume that we have a root-interface $(\rho_a)_{a\in\mathrm{Rep}(b)}$ at position b. Using this root-interface, We now build a hybrid derivation proving $C' \vdash t' : T'$, where $C' \equiv C$ and $T' \equiv T$.

First, we define the residual position $\mathrm{Res}_b(\alpha)$ for each $\alpha \in A$ except when α is of the form a, $a \cdot 1$ or $a \cdot 10 \cdot a_k$ (for $a \in \mathrm{Rep}(b)$).

- Case Right: if $\alpha = a \cdot k_R \cdot \alpha_0$ where $\overline{a} = b$ and $k_R \in \mathrm{Arg}(a)$, then $\mathrm{Res}_b(\alpha) = a \cdot a_{k_L} \cdot \alpha_0$ with $k_L = \rho_a^{-1}(k_R)$.
- Case Left: if $\alpha = a \cdot 10 \cdot \alpha_0$ where $a \in \mathrm{Rep}(b)$ and $\alpha_0 \neq a_k$, then $\mathrm{Res}_b(\alpha) = a \cdot \alpha_0$.
- Case Out: if $b \not\leqslant \overline{\alpha}$, then $\mathrm{Res}_b(\alpha) = \alpha$

We set $A' = \{\mathrm{Res}_b(\alpha) \mid \alpha \in A\}$. So A' is a tree.

Induction and reduction: assume $\mathrm{Res}_b(\alpha_i) = \alpha'_i$ ($i = 1, 2$) and $\alpha_1 \leqslant \alpha'_2$. If α'_1 is "Right" residual, then α_2 alors is and if α'_1 is a "Left Residual", then α_2 is a "Left" or "Right" residual. So, when we proceed by $\searrow$-induction on A', we study first the case Right, then the case Left, then the case Out (so that induction on A' respects induction on A).

Conversely, we define $\mathrm{Res}_b^{-1}(\alpha')$ for all $\alpha' \in A'$ so that we have $\mathrm{Res}_b(\mathrm{Res}_b^{-1}(\alpha')) = \alpha'$ for all $\alpha' \in A'$ and $\mathrm{Res}_b^{-1}(\mathrm{Res}_b(\alpha)) = \alpha$ for all α s.t. $\mathrm{Res}_b(\alpha)$ is defined:

- If $b \notin \overline{\alpha'}$, we set $\mathrm{Res}_b^{-1}(\alpha') = \alpha'$.
- If $\alpha' = a{\cdot}\alpha'_0$ where $a \in \mathrm{Rep}(b)$ but there is no k s.t. $b \geqslant a{\cdot}a_k$, we set $Res_b^{-1}(\alpha') = a{\cdot}10{\cdot}\alpha'_0$.
- If $\alpha' = a \cdot a_k \cdot \alpha'_0$ where $a \in \mathrm{Rep}(b)$, we set $\mathrm{Res}_b^{-1}(\alpha') = a \cdot \rho_a(k) \cdot \alpha'_0$.

Those functions are built so that:

▶ **Lemma 1.** For all $\alpha' \in A'$ and α s.t. $\mathrm{Res}_b(\alpha) = \alpha'$:
- $\overline{\alpha'} \in \mathtt{supp}(t')$
- $t'(\overline{\alpha'}) = t'(\overline{\alpha})$.
- $\mathrm{OutTr}'(\alpha') = \mathrm{OutTr}(\alpha)$ (where we have set $\mathrm{OutTr}'(\alpha') = \{k \in \mathbb{N} \mid \alpha' \cdot k \in A'\}$).

4.3 Residual Types and Contexts

Let Ax' the set of leaves of A'. For all $\alpha' \in A'$ and $y \in \mathcal{V} - \{x\}$, we set $\mathrm{Ax}'(\alpha)(y) = \{\alpha'_0 \in A' \mid \alpha'_0 \geqslant \alpha'$ and $\mathrm{Res}_b^{-1}(\alpha) \in \mathrm{Ax}(y)\}$ and $\mathrm{Ax}'(y) = \mathrm{Ax}'(\varepsilon)(y)$.

We observe that, for all $\alpha' \in \mathrm{Ax}'$, $\mathrm{Res}_b^{-1}(\alpha') \in \mathrm{Ax}$. Then, we set $\mathrm{tr}'(\alpha') = \mathrm{tr}(\mathrm{Res}_b^{-1}(\alpha'))$ and, for all $\alpha' \in A'$ and $y \in \mathcal{V} - \{x\}$, $C'(\alpha')(y) = (\mathrm{tr}'(\alpha'_0) \cdot T(\mathrm{Res}_b^{-1}(\alpha'_0)))_{\alpha'_0 \in \mathrm{Ax}'(\alpha')(y)}$. This definition is sound, because, if $\alpha_1,\ \alpha_2 \in \mathrm{Ax}(\alpha')(y)$, then $\mathrm{tr}'(\alpha_1) = \mathrm{tr}'(\alpha_2)$ implies $\alpha_1 = \alpha_2$ (case analysis).

When $t'(\overline{\alpha'}) = \lambda y$ (with $\alpha' \in A'$), a case analysis shows that $\mathrm{Ax}(\alpha')(y) = \{\mathrm{Res}_b(\alpha_0) \mid \alpha_0 \in \mathrm{Ax}(\alpha)(y)\}$ where $\alpha' \in A'$. Thus, in that case, $C(\alpha')(y) = (\mathrm{tr}'(\alpha'_0){\cdot}T(\mathrm{Res}_b^{-1}(\alpha'_0)))_{\alpha'_0 \in \mathrm{Ax}'(\alpha')(y)} = (\mathrm{tr}(\alpha_0) \cdot T(\alpha_0))_{\alpha_0 \in \mathrm{Ax}(\alpha)(y)} = C(\alpha)(y)$.

By a $\searrow$-induction on $\alpha' \in A'$, we define now $T'(\alpha')$:
- When $\alpha' \in \mathrm{Ax}'$, $T'(\alpha') = T(\mathrm{Res}_b^{-1}(\alpha')$.
- When $t'(\overline{\alpha'}) = \lambda y$, we set $T'(\alpha) = C(\alpha \cdot 0)(y) \to T'(\alpha \cdot 0)$.
- When $t'(\overline{\alpha'}) = @$, we set $T'(\alpha') = \mathrm{Hd}(T'(\alpha' \cdot 1))$.

We define then P' as the labelled tree s.t. $\mathtt{supp}(P') = A'$ and for all $\alpha' \in A',\ P'(\alpha') = C'(\alpha') \vdash t'|_{\overline{\alpha'}} : T'(\alpha')$. We intend to prove that P' is a correct h.d. (typing t').

4.4 Soundness

We define now the **quasi-residual** $\mathrm{QRes}_b(\alpha)$ for any $\alpha \in A$ such that $\overline{\alpha} \neq b{\cdot}1$ by $\mathrm{QRes}_b(\alpha) = \mathrm{Res}_b(\alpha)$ when $\mathrm{Res}_b(\alpha)$ is defined, $\mathrm{QRes}_b(a) = a$ and $\mathrm{QRes}_b(a \cdot 10 \cdot a_k) = a \cdot a_k$ when $\overline{a} = b$ and $k \in \mathrm{RedTr}(a)$.

▶ **Remark 5.** We do not necessary have $t(\alpha) = t'(\alpha')$ or $\mathrm{OutTr}(\alpha) = \mathrm{OutTr}'(\alpha')$ when $\alpha' = \mathrm{QRes}_b(\alpha)$ (contrary to lemma **??**). However, quasi-residuals are more convenient to define the isomorphisms $\mathrm{Res}_{b|\alpha}$, $\mathrm{ResR}_{b|\alpha}$ and $\mathrm{ResL}_{b|\alpha}$ below.

▶ **Lemma 2.**
- For all $\alpha' \in A',\ T'(\alpha') \equiv T(\alpha)$ where $\alpha = \mathrm{Res}_b^{-1}(\alpha')$. Besides, if $\alpha' \in \mathrm{Ax}'$ or $\alpha' \geqslant a \cdot a_k$ (for $\overline{a} = b$), then $T'(\alpha') = T(\alpha)$.
- More precisely, if P is endowed with an interface (ϕ_a) at position b (extending the root-interface (ρ_a)), then, for all $\alpha \in A$ and $\alpha' \in A'$ such that $\mathrm{QRes}_b(\alpha) = \alpha'$, we can define a type isomorphism $\mathrm{QRes}_{b|\alpha} : T(\alpha) \to T'(\alpha')$ by $\searrow$-induction on α'.

- When $\mathrm{Res}_b(\alpha) = \alpha'$, we write $\mathrm{Res}_{b|\alpha}$ instead of $\mathrm{QRes}_{b|\alpha}$. Moreover, $\mathrm{Res}_{b|\alpha}$ is the identity if $\alpha' \in \mathrm{Ax}'$ or $\alpha' \geqslant a \cdot a_k$ for some $a \in \mathrm{Rep}(b)$ and $k \in \mathrm{AxTr}(a \cdot 10)(x)$.

Proof.
- Case Right: $\mathrm{Res}_b(\alpha) = \alpha'$ and $\alpha' \geqslant a \cdot a_k$ (for some $a \in \mathrm{Rep}(b)$ and $k \in \mathrm{RedTr}(a)$).
  - Subcase $t(\overline{\alpha}) = y$: here, $t(\overline{\alpha}) = y \neq x$ and $T'(\alpha') = T(\alpha)$.
  - Subcase $t(\overline{\alpha}) = \lambda y$: $\alpha \cdot 0 \in A$, $\mathrm{Res}_b(\alpha \cdot 0) = \alpha' \cdot 0$ and by IH, we have $T'(\alpha' \cdot 0) = T(\alpha \cdot 0)$ and $\mathrm{Res}_{b|\alpha \cdot 0}$ is the identity.
    Since $T(\alpha) = C(\alpha \cdot 0)(y) \to T(\alpha \cdot 0)$ and $T'(\alpha') = C(\alpha \cdot 0)(y) \to T'(\alpha' \cdot 0)$, we also have $T(\alpha) = T'(\alpha)$ and we set $\mathrm{Res}_{b|\alpha} = \mathtt{id}$.
  - Subcase $t(\overline{\alpha}) = @$: $\alpha \cdot 1 \in A$, $\mathrm{Res}_b(\alpha \cdot 1) = \alpha' \cdot 1$ and by IH, we have $T'(\alpha \cdot 0) = T(\alpha' \cdot 0)$ and $\mathrm{Res}_{b|\alpha \cdot 1}$ is the identity.
    Moreover, $T(\alpha) = \mathrm{Hd}(T(\alpha \cdot 1))$ and $T'(\alpha') = \mathrm{Hd}(T'(\alpha'))$. So $T(\alpha) = T(\alpha')$ and we set $\mathrm{Res}_{b|\alpha} = \mathtt{id}$.

- Case Left: $\alpha \geqslant a \cdot 10$ and $\alpha' \geqslant a$ (for some $a \in \mathrm{Rep}(b)$):
  - Subcase $\alpha = a \cdot 10 \cdot a_{k_L}$ and $\alpha' = a \cdot a_{k_L}$: $\alpha' = \mathrm{Res}_b(a \cdot k_R)$ (where $k_R = \rho_a(k_L)$) and by IH, $T'(\alpha') = T(a \cdot k_R)$. Moreover, since $T(\alpha) = \mathrm{L}(\alpha)|_{k_L}$, we can set $\mathrm{QRes}_{b|\alpha} = \phi_{a|k_L}$.
  - Subcase $t(\overline{\alpha}) = y \neq x$: $\mathrm{Res}_b(\alpha) = \alpha'$ and $T'(\alpha') = T(\alpha)$.
  - Subcase $t(\overline{\alpha}) = \lambda y$: $\alpha \cdot 0 \in A$, $\mathrm{Res}_b(\alpha \cdot 0) = \alpha' \cdot 0$: we set $\mathrm{Res}_{b|\alpha} = C(\alpha \cdot 0)(y) \to \mathrm{QRes}_{b|\alpha \cdot 0}$.
  - Subcase $t(\overline{\alpha}) = @$: $\alpha \cdot 1 \in A$, $\mathrm{Res}_b(\alpha \cdot 1) = \alpha' \cdot 1$: we set $\mathrm{Res}_{b|\alpha} = \mathrm{Hd}(\mathrm{QRes}_{b|\alpha \cdot 1})$.

- Case Out: $\overline{\alpha} \not\geqslant b$:
  - Subcase $\alpha' \in \mathrm{Ax}'$: here, $t(\overline{\alpha}) = y \neq x$ and $T'(\alpha') = T(\alpha)$.
  - Subcase $\alpha = \alpha' = a$ (for a $a \in \mathrm{Rep}(b)$): $a = \mathrm{Res}_b(a \cdot 10)$ and by IH, we have an type isomorphism $\mathrm{QRes}_{b|a \cdot 10} : T(a \cdot 10) \to T(a)$. Since $T(a \cdot 10) = T(a)$, we can set $\mathrm{QRes}_{b|a} = \mathrm{QRes}_{b|a \cdot 10}$.
  - Subcase $t(\overline{\alpha}) = \lambda y$: $\alpha \cdot 0 \in A$, $\mathrm{QRes}_b(\alpha \cdot 0) = \alpha' \cdot 0$ and we set $\mathrm{Res}_{b|\alpha} = C(\alpha \cdot 0)(y) \to \mathrm{QRes}_{b|\alpha \cdot 0}$
  - Subcase $t(\overline{\alpha}) = @$: $\alpha \cdot 1 \in A$, $\mathrm{QRes}_b(\alpha \cdot 1) = \alpha' \cdot 1$: we set $\mathrm{Res}_{b|\alpha} = \mathrm{Hd}(\mathrm{QRes}_{b|\alpha \cdot 1})$.

◀

4.5 Residual Interface

We notice that if $\alpha \in A[@]$ and $\overline{\alpha} \neq b$, then $\mathrm{Res}_b(\alpha)$ is defined. So, for $\alpha' \in A'[@]$, we set $\mathrm{L}'(\alpha') = \mathrm{Tl}(T(\alpha'))$, $\mathrm{Arg}'(\alpha') = \{k \geqslant 2 \mid \alpha' \in A'\}$ and $\mathrm{R}'(\alpha') = (k \cdot T'(\alpha' \cdot k))_{k \in \mathrm{Arg}'(\alpha)}$. We write then $\mathrm{Inter}'(\alpha')$ for the set of f.t. isomorphisms from $\mathrm{L}'(\alpha')$ to $\mathrm{R}'(\alpha')$.

Let us write $\alpha = \mathrm{Res}_b^{-1}(\alpha')$, so that $\alpha \cdot 1 \in A$, $\overline{\alpha} \neq b \cdot 1$, $\mathrm{QRes}_b(\alpha \cdot 1) = \alpha' \cdot 1$ and $\mathrm{OutTr}(\alpha) = \mathrm{OutTr}'(\alpha')$ (thanks to lemma **??**).

- Since $\mathrm{QRes}_{b|\alpha \cdot 1} : T(\alpha \cdot 1) \to T'(\alpha' \cdot 1)$ is a type isomorphism and $\mathrm{L}'(\alpha') = \mathrm{Tl}(T'(\alpha' \cdot 1))$, we define the f.t. isomorphism $\mathrm{ResL}_{b|\alpha}$ by $\mathrm{ResL}_{b|\alpha} = \mathrm{Tl}(\mathrm{Res}_{b|\alpha \cdot 1})$.
- We can define $\mathrm{ResR}_{b|\alpha}$ by $\mathrm{ResR}_{b|\alpha}(k \cdot \gamma) = k \cdot \mathrm{Res}_{b|\alpha \cdot k}(\gamma)$. It is a f.t. isomorphism from $\mathrm{L}(\alpha)$ to $\mathrm{L}'(\alpha')$.

Thus, for each application node α s.t. $\alpha \notin \mathrm{Rep}(b)$, $\alpha' = \mathrm{Res}_b(\alpha)$ is defined and we can define a bijection $\mathrm{ResI}_{b|\alpha}$ from $\mathrm{Inter}(\alpha)$ to $\mathrm{Inter}'(\alpha')$ by $\mathrm{ResI}_{b|\alpha}(\phi) = \mathrm{ResR}_{b|\alpha} \circ \phi \circ \mathrm{ResL}_{b|\alpha}^{-1}$, so that the following diagram is commuting:

$$\begin{array}{ccc} \mathrm{L}(\alpha) & \xrightarrow{\phi} & \mathrm{R}(\alpha) \\ \Big\downarrow{\scriptstyle \mathrm{ResL}_{b|\alpha}} & & \Big\downarrow{\scriptstyle \mathrm{ResR}_{b|\alpha}} \\ \mathrm{L}'(\alpha') & \xrightarrow{\mathrm{ResI}_{b|\alpha}(\phi)} & \mathrm{R}'(\alpha') \end{array}$$

It means that the set of interface isomorphisms at position $\alpha' = \mathrm{Res}_b(\alpha)$ in P' can also be seen as the residual bijective image of the set of interface isomorphisms at position α in P. This observation is pivotal to prove the Representation Lemma (next subsec.).

Assume P is endowed with a complete interface (ϕ_a). For all $\alpha' \in A'[@]$, we set $\phi'_{\alpha'} = \mathrm{ResI}_{b|\alpha}(\phi_\alpha)$, where $\alpha = \mathrm{Res}_b^{-1}(\alpha')$. Notice that we can get back ϕ_α from $\phi'_{\alpha'}$ since $\mathrm{ResI}_{b|\alpha}$ is a bijection. We have enough to state:

▶ **Proposition 2.** The labelled tree P' defined at end of § **??** is a hybrid derivation.

- When P is endowed with an interface (ϕ_α), for all $\alpha' \in A'$, $\phi'_{\alpha'}$ is an interface isomorphism at pos. α'.
- Thus, the family $(\phi'_{\alpha'})$ is a complete interface for P'. We call it the **residual interface** of (ϕ_α) after firing the redex at position b. When P' is endowed with the residual interface, it is an o.d.

Thus, if needed, we can apply a new β-reduction in t' according to this residual interface without having to define a new one. It allows us to define deterministically the way we perform reduction (inside a derivation) in any reduction sequence of length $\ell \leqslant \omega$. To sum up:

▶ **Remark 6.** We need only a root-interface at position b to define the h.d. P'.

- If we have an interface at position b, it allows to define the isomorphisms $\mathrm{Res}_{b|\alpha} : T(\alpha) \to T'(\alpha')$ (resp. $\mathrm{QRes}_{\alpha|\alpha} : T(\alpha) \to T'(\alpha')$ for all α s.t. $\alpha' := \mathrm{Res}_b(\alpha)$ is defined (resp. s.t. $\alpha' := \mathrm{QRes}_b(\alpha)$) is defined, as well as $\mathrm{ResI}_{b|\alpha}$ for all $\alpha \in A(@)$ s.t. $\overline{\alpha} \neq b$.
- It allows to choose other interfaces (at positions different from b) *after* firing the redex at position b. This observation is only one we need to prove the Representation Lemma.

4.6 Representation Lemma

We prove here that every sequence of subject-reduction steps that we perform "by hand" – so called a **reduction choice sequence** – starting from a derivation Π can be built-in inside an o.d.

Let's assume $t|_b = (\lambda x.r)s$, $t \xrightarrow{b} t'$ and $\Pi \rhd \Gamma \vdash t : \tau$ and have a second look at the figure representing subject reduction in Appendix **??**.

Morally, subject-reduction consists in choosing which axiom rule typing x will be replaced by which argument derivation (for each $a \in \mathrm{Rep}_A(b)$). It yields a derivation $\Pi' \rhd \Gamma \rhd t' : \tau$, which depends on the substitution choices we have made.

If a h.d. P represents Π (the hybrid construction grants there is one), any reduction choice sequence can be represented by a root-interface (ρ_a) at position b. Namely, if we reduce P according to (ρ_a), we produce a new hybrid derivation P' typing t' which represents our chosen derivation Π'.

Let $\Pi \rhd \Gamma \vdash t : \tau$ be a derivation, P a h.d. representing Π and $t = t_0 \overset{b_0}{\to} t_1 \overset{b_1}{\to} t_2 \overset{b_2}{\to} \dots \overset{b_{i-1}}{\to} t_i \overset{b_i}{\to} \dots$ a sequence of reduction of length $\ell \leqslant \omega$ (when $\ell = \omega$, we do not need to assume strong convergence [**?**]).

We write $\mathcal{R}$ for the sequence $(b_i)_{i<\ell}$ and $\mathcal{R}(n)$ for the sequence $(b_i)_{i<n}$ for all $n < \ell$. If we perform subject-reduction on P along reduction sequence $\mathcal{R}$, we get a sequence of h.d. P_0 (with $P_0 = P$), P_1, $P_2, \dots$ such that P_i types t_i.

More precisely, for each step $i < \ell$ of $\mathcal{R}$, we have to choose a root-interface (ρ^i_a) at position b_i in P_i (typing t_i) corresponding to our reduction choice step, then reduce the h.d. P_i according to (ρ^i_a), which yields a new h.d. P_{i+1}. We proceed by induction on i.

Those reduction choices are heuristically made step-by-step. This raises the following question: is the notion of operable derivation expressive enough ? That is: can we endow P with a complete interface, such that performing $\mathcal{R}$ on P follows exactly our step-by-step choices of substitution? The answer is positive, according to the next lemma.

For now, we set $A_i = \mathsf{supp}(P_i)$ for all $i < \ell$ and we define by induction on $i < \ell$ a partial function $\mathrm{Res}_{\mathcal{R}(i)}$ from A to A_i:

- $\mathrm{Res}_{\mathcal{R}(0)}$ is the identity on A.
- $\mathrm{Res}_{\mathcal{R}(i+1)} = \mathrm{Res}_{i|b_i} \circ \mathrm{Res}_{\mathcal{R}(i)}$, where $\mathrm{Res}_{i|b_i}$ is the residual function on support (in the sense of § **??**) defined w.r.t. the root-interface (ρ^i_a) at position b_i in P_i.

For all $i < \ell$, let $A_{\mathcal{R}(i)}$ denote the domain of $\mathrm{Res}_{(i)}$. Thus, $A_{\mathcal{R}(0)} = A$, $(A_{\mathcal{R}(i)})_{i<\ell}$ is a decreasing sequence (w.r.t. $\subset$) and, by induction on i, $\mathrm{Res}_{(i)}$ is a bijection from $A_{\mathcal{R}(i)}$ to A_i. We write $\mathrm{Res}^{-1}_{\mathcal{R}(i)}$ for the converse bijection from A_i to $A_{\mathcal{R}(i)}$.

To lighten notations, we write $A_{(i)}$ and $\mathrm{Res}_{(i)}$ instead of $A_{\mathcal{R}(i)}$ and $\mathrm{Res}_{\mathcal{R}(i)}$. We also set $A_{(i)}[@] = A_{(i)} \cap A[@]$. Thus, $\mathrm{Res}_{(i)}$ induces a bijection from $A_{(i)}$ to A_i.

Now, for all $i < \ell$, we chose an interface (ϕ^i_a) at position b_i in P_i such that $\phi^i_{a|\,\mathrm{Rt}} = \rho^i_a$ for all $a \in \mathrm{Rep}_{A_i}(b_i)$.

We define by induction on i a type isomorphism $\mathrm{Res}_{(i)|\alpha} : T(\alpha) \to T_i(\mathrm{Res}_{(i)}(\alpha))$ for all $\alpha \in A_{(i)}$ and a bijection $\mathrm{ResI}_{(i)|\alpha} : \mathrm{Inter}(\alpha) \to \mathrm{Inter}_i(\mathrm{Res}_{(i)}(\alpha))$ for all $\alpha \in A_{(i)|\alpha}$ by:

- $\mathrm{Res}_{(0)|\alpha}$ and $\mathrm{Res}_{(i)|\alpha}$ are respectively the identity functions on $T(\alpha)$ and $\mathrm{Inter}(\alpha)$.
- $\mathrm{Res}_{(i+1)|\alpha} = \mathrm{Res}_{b_i|\alpha_i} \circ \mathrm{Res}_{(i)|\alpha}$, where $\alpha_i = \mathrm{Res}_{(i)}(\alpha)$ and $\mathrm{Res}_{b_i|\alpha_i} : T_i(\alpha_i) \to T_{i+1}(\alpha_{i+1})$ (with $\alpha_{i+1} = \mathrm{Res}_{(i+1)}(\alpha)$) is the residual type isomorphism (in the sense of §**??**) w.r.t. the interface (ϕ^i_a) at position b in P_i.
  We set likewise $\mathrm{ResI}_{(i+1)|\alpha} = \mathrm{ResI}_{b_i|\alpha_i} \circ \mathrm{ResI}_{(i)|\alpha}$, where $\mathrm{ResI}_{b_i|\alpha_i}$ is the bijection (in the sense of §**??**) w.r.t. the interface (ϕ^i_a) at position b in P_i.

Now, let $a \in A(@)$. There are two cases:

- $\mathrm{Res}_{(i)}(a)$ is defined for all $i < \ell$. In that case, we choose an arbitrary $\phi_a \in \mathrm{Inter}(\alpha)$.
- There is a unique $0 \leqslant i < \ell$ such that $\alpha_i = \mathrm{Res}_{(i)}(\alpha)$ is defined, but $\mathrm{Res}_{(i+1)}(\alpha)$ is not. In that case, $\overline{\alpha_i} = b_i$ and we have already chosen an interface isomorphism $\phi^i_{\alpha_i} \in \mathrm{Inter}_i(\alpha_i)$ (that extends $\rho^i_{\alpha_i}$). We set then $\phi_a = \mathrm{ResI}^{-1}_{(i)|a}(\phi^i_{\alpha_i})$

By construction, the complete interface (ϕ_a) emulates the reduction w.r.t. the family (ρ^i_a). Thus:

▶ **Lemma 3.** Every reduction choice sequence in a quantitative derivation Π can be built-in in an operable derivation representing Π.

5 Resetting Track Values

In the present section, we consider *isomorphisms* between operable derivations (h.d. endowed with complete interface, § **??**). Roughly, two derivations P_1 and P_2 judging the same term are isomorphic when their supports are 01-isomorphic (§ **??**) and, at corresponding axiom leaves a_1 (in P_1) and a_2 (in P_2), we find isomorphic types $T_1(a_1)$ and $T_2(a_2)$. In that case, the typing constraints allow to $\searrow$-inductively define type and context isomorphisms at each position inside P_1 and P_2. Next, we notice that, in order to define an isomorphism, we only need to indicate what value (inner or outer) track – which is not equal to 0 or 1 – should receive. It leads to the notions of **mutable positions** and of **track resetting**.

In Subsections **??**, our studys of track resetting will help us formulate the conditions that allow to get, from any *operable* derivation P, a *trivial* derivation P_s that is isomorphic to P.

5.1 Isomorphisms of Operable Derivations

Let P_1 and P_2 be two hybrid derivations built from the same derivation Π. We write Ax_i for the set of axiom leaves of P_i $(i = 1, 2)$ and $C_i(\alpha)$, $T_i(\alpha)$ for the context and type at position $\alpha \in A_i$, tr_i, pos_i are the corresponding notations w.r.t. P_i.

A **hybrid derivation isomorphism** $\Psi : P_1 \to P_2$ is given by:

- A 01-stable tree-isomorphism $\Psi_{\mathtt{supp}} : A_1 \to A_2$. We often write Ψ instead of $\Psi_{\mathtt{supp}}$.
- For each $\alpha_1 \in \mathrm{Ax}_1$, we have a type isomorphism $\Psi_{\alpha_1} : T_1(\alpha_1) \to T_2(\alpha_2)$, where $\alpha_2 = \Psi_{\mathtt{supp}}(\alpha_1)$.

This is enough to define many useful isomorphisms:

- Since $\Psi_{\mathtt{supp}}$ is a tree isomorphism, it induces a bijection from Ax_1 to Ax_2.
  Thus, for all $\alpha_1 \in A^1$, we can set $\Psi_{\mathrm{tr}}(\alpha_1) = \mathrm{tr}_2(\Psi(\alpha_1))$.

- Since $\Psi_{\mathtt{supp}}$ induces a bijection from $\mathrm{Ax}_1(\alpha_1)(x)$ to $\mathrm{Ax}_2(\alpha_2)(x)$ for all $\alpha_1 \in A_1$, $\alpha_2 = \Psi(\alpha_1)$ and $x \in \mathcal{V}$, we can define a context isomorphism $\Psi_{\alpha_1,x} : C_1(\alpha_1)(x) \to C_2(\alpha_2)(x)$ by $\Psi_{\alpha_1,x}(k_1 \cdot \gamma_1) = \Psi_{\mathrm{tr}}(\alpha_{01}) \cdot \Psi_{\alpha_{01}}(\gamma_1)$, where $\alpha_{01} = \mathrm{pos}_1(\alpha_1, x, k_1)$.

- It allows to define a type isomorphism $\Psi_{\alpha_1} : T_1(\alpha_1) \to T_2(\alpha_2)$ for any $\alpha_1 \in A_1$ and $\alpha_2 = \Psi(\alpha_1)$ by $\searrow$-induction.
  - Ψ_{α_1} is already defined when $\alpha_1 \in \mathrm{Ax}_1$.
  - If $t(\overline{\alpha_1}) = \lambda x$, we set $\Psi_{\alpha_1} = \Psi_{\alpha_1 \cdot 0, x} \to \Psi_{\alpha_1 \cdot 0}$
  - If $t(\overline{\alpha_1}) = @$, we set $\Psi_{\alpha_1} = \mathrm{Hd}(\Psi_{\alpha_1 \cdot 1})$.

- We assume here that $t(\overline{\alpha_1}) = @$ and $\alpha_2 = \Psi(\alpha_1)$.
  - We set $\Psi^{\mathrm{L}}_{\alpha_1} = \mathrm{Tl}(\Psi_{\alpha_1})$.
  - Since $\Psi_{\mathtt{supp}}$ induces a bijection from $\mathrm{Arg}_1(\alpha_1)$ to $\mathrm{Arg}_2(\alpha_2)$, we define a f.t. isomorphism $\Psi^{\mathrm{R}}_{\alpha_1} : \mathrm{R}_1(\alpha_1) \to \mathrm{R}_2(\alpha_2)$ by $\Psi^{\mathrm{R}}_{\alpha_1}(k_1 \cdot \gamma_1) = k_2 \cdot \gamma_2$, where $\alpha_2 \cdot k_2 = \Psi_{\mathtt{supp}}(\alpha_1 \cdot k_1)$ and $\gamma_2 = \Psi_{\alpha_1 \cdot k_1}(\gamma_1)$.

- We define now Ψ on bisupports (with $\alpha_2 = \Psi_{\mathtt{supp}}(\alpha_2)$).
  - Left case: $\Psi(\alpha_1,\, x,\, k_1 \cdot \gamma_2) = (\alpha_2,\, x,\, \Psi_{\alpha_1,x}(k_1 \cdot \gamma_1))$

- Right case: $\Psi(\alpha_1, \gamma_1) = (\alpha_2, \Psi_{\alpha_1}(\gamma_1))$.

Let P_1 and P_2 be two o.d. typing the same term. Their interface isomorphisms are written $(\phi_{i,\alpha})$ $(i = 1, 2)$. A **operable derivation isomorphism** is a a hybrid derivation isomorphism $\Psi : P_1 \to P_2$ such that for all $\alpha_1 \in A_1[@]$ and $\alpha_2 = \Psi_{\mathtt{supp}}(\alpha_1)$, the following diagram is commuting:

$$\begin{array}{ccc} \mathrm{L}_1(\alpha_1) & \xrightarrow{\phi_{1,\alpha_1}} & \mathrm{R}_1(\alpha_1) \\ \downarrow{\scriptstyle \Psi^{\mathrm{L}}_{\alpha_1}} & & \downarrow{\scriptstyle \Psi^{\mathrm{R}}_{\alpha_1}} \\ \mathrm{L}_2(\alpha_2) & \xrightarrow{\phi_{2,\alpha_2}} & \mathrm{R}_2(\alpha_2) \end{array}$$

5.2 Resetting an Operable Derivation

Let P an operable derivation. We reuse the notations A, C, T, Ax, tr, pos and (ϕ_α).
In the purpose of proving that every operable derivation has a trivial representant, we want to define from P new operable derivations P_0 that are isomorphic to P.

A **bisupport resetting** of P is given by the following:

- $\Psi_{\mathtt{supp}}$ a function from A to $\mathbb{N}^*$ inducing a 01-stable tree isomorphism from A to its codomain A_0. We write Ax_0 for the set of leaves of A_0.
- A function T_0 from Ax_0 to Types and, for each $\alpha \in \mathrm{Ax}$ a type isomorphism $\Psi_\alpha : T(\alpha) \to T_0(\Psi_{\mathtt{supp}}(\alpha))$
- An *injection* $\Psi_{\mathrm{tr}} : \mathrm{Ax} \to \mathbb{N} - \{0, 1\}$.

- Let $\mathrm{Ax}_0(\Psi(\alpha))(x) = \{\Psi_{\mathtt{supp}}(\alpha_0) \mid \alpha_0 \in \mathrm{Ax}(\alpha)(x)\}$ for all $\alpha \in A$ and $x \in \mathcal{V}$.
  Since $\Psi_{\mathtt{supp}}$ induces a bijection from Ax to Ax_0, we can set $\mathrm{tr}_0(\Psi(\alpha)) = \Psi_{\mathrm{tr}}(\alpha)$ for all $\alpha \in \mathrm{Ax}$ and $\mathrm{AxTr}_0(\Psi(\alpha))(x) = \Psi_{\mathrm{tr}}(\mathrm{Ax}(\alpha)(x))$
  Since Ψ_{tr} is an injection whose domain is Ax, we can define $\mathrm{pos}_0 : \mathrm{codom}(\Psi_{\mathrm{tr}}) \to \mathrm{Ax}'$ with $\mathrm{pos}_0(k_0) = \alpha_0$, where $\alpha_0 \in \mathrm{Ax}_0$ is the unique leaf of A_0 such that $\mathrm{tr}_0(\alpha_0) = k_0$ (pos_0 requires only one argument, contrary to pos).

- We define then $C_0(\Psi(\alpha))(x) = (\mathrm{tr}_0(\Psi(\alpha_0)) \cdot T(\Psi(\alpha_0)))_{\alpha_0 \in \mathrm{Ax}(\alpha)(x)}$. So we can write $\Psi_{\alpha,x}$ for the context isomorphism from $C(\alpha)(x)$ to $C_0(\alpha_0)(x)$ such that $\Psi_{\alpha,\ x}(k \cdot \gamma) = \Psi_{\mathrm{tr}}(\alpha_0) \cdot \Psi_{\alpha_0}(\gamma)$, where $\alpha_0 = \mathrm{pos}(\alpha, x, k)$.

- We can now define a type $T_0(\Psi(\alpha))$ and a type isomorphism $\Psi_\alpha : T(\alpha) \to T_0(\Psi(\alpha))$ for all $\alpha \in A$ by $\searrow$-induction.
  - $T_(\alpha_0)$ and Ψ_α are already defined when $\alpha \in \mathrm{Ax}$.
  - If $t(\overline{\alpha}) = \lambda x$, we set $T_0(\alpha_0) = C_0(\alpha_0)(x) \to T_0(\alpha_0 \cdot 0)$ and $\Psi_\alpha = \Psi_{\alpha\cdot 0,x} \to \Psi_{\alpha\cdot 0}$.
  - If $t(\overline{\alpha}) = @$, we set $T_0(\alpha_0) = \mathrm{Hd}(T_0(\alpha_0)$ and $\Psi_\alpha = \mathrm{Hd}(\Psi_{\alpha\cdot 1})$.

- We assume here that $t(\overline{\alpha}) = @$ and $\alpha_0 = \Psi(\alpha)$. We set then $\mathrm{Arg}_0(\Psi(\alpha)) = \{k_0 \geqslant 2 \mid \exists k \in \mathrm{Arg}(\alpha),\ \alpha_0{\cdot}k_0 = \Psi(\alpha{\cdot}k)\}$, $\mathrm{L}_0(\alpha) = \mathrm{Tl}(T_0(\alpha_0{\cdot}1))$ and $\mathrm{R}_0(\alpha_0) = (k_0{\cdot}T_0(\alpha_0{\cdot}k_0))_{k_0 \in \mathrm{Arg}_0(\alpha_0)}$.
  - We set $\Psi^{\mathrm{L}}_\alpha = \mathrm{Tl}(\Psi_\alpha)$.

- We define a f.t. isomorphism $\Psi^{\mathrm{R}}_{\alpha} : \mathrm{R}(\alpha) \to \mathrm{R}_0(\alpha_0)$ by $\Psi^{\mathrm{R}}_{\alpha}(k \cdot \gamma) = k_0 \cdot \gamma_0$, where $\alpha_0 \cdot k_0 = \Psi(\alpha_1 \cdot k_1)$ and $\gamma_0 = \Psi_{\alpha \cdot k}(\gamma_0)$.

We can set $\phi_{0,\alpha_0} = \Psi^{\mathrm{R}}_{\alpha} \circ \phi_{\alpha} \circ (\Psi^{\mathrm{L}}_{\alpha})^{-1}$. Thus, $\phi_{0,\alpha_0} : \mathrm{L}_0(\alpha_0) \to \mathrm{R}_0(\alpha_0)$ is a f.t. isomorphism.

The following property stems from the previous constructions:

▶ **Proposition 3.** Let P_0 be the labelled tree such that $\mathtt{supp}(P_0) = A_0$ and $P_0(\alpha_0) = C_0(\alpha_0) \vdash t|_{\overline{\alpha_0}} : T_0(\alpha_0)$.
Then P_0 is a hybrid derivation and $(\phi_{0,\,\alpha_0})$ is a complete interface that makes P_0 isomorphic to P as an o.d via the isomorphism Ψ.

5.3 Track Resetting

In a quantitative hybrid derivation P, every type and context is determined by the types and axiom tracks given in the axiom leaves (if the forget about outer argument tracks). So resetting P (hopefully, into a *trivial* derivation) is mainly a matter of giving good values to tracks in the axiom leaves, and also to argument (outer) tracks at application nodes.

If we keep in mind that the track values 0 and 1 leave us no choice, it motivates the following definitions:

▶ **Definition 4.**
- Let U be a labelled tree or forest. The set of **mutable positions** of U – written $\mathtt{supp}^{?}(U)$ – is defined by $\mathtt{supp}^{?}(U) = \{a \in \mathtt{supp}(U) - \{\varepsilon\} \mid *(a) \geqslant 2\}$.
- The set of **referent bipositions** $\mathrm{rbp}(P)$ is defined by $\{(\alpha, \gamma) \in \mathrm{bisupp}(P) \mid \eta \in \mathrm{Ax} \text{ and } \zeta \in \mathtt{supp}^{?}(T(\alpha))\}$.
- The set of **outer argument positions** Arg is defined by $\mathrm{OAP} = \{\alpha \in A - \{\varepsilon\} \mid \mathrm{last}(\alpha) \geqslant 2\}$, *i.e.* $\mathrm{OAP} = \mathtt{supp}^{?}(P)$. If $a \in A[@]$, we wet $\mathrm{OAP}(a) = a \cdot \mathrm{Arg}(a)$.
- The set of **referents** of P is defined by $\mathrm{ref}\, P = \mathrm{rbp}(P) \cup \mathrm{Ax} \times \{\varepsilon\} \cup \mathrm{OAP}$.

▶ **Remark 7.** We use $\mathrm{Ax} \times \{\varepsilon\}$ because Ax and OAP may be not disjointed. In that case, if $\alpha \in \mathrm{Ax} \cap \mathrm{OAP}$, α holds an axiom track $\mathrm{tr}(\alpha)$ as well as its outer argument track $\mathrm{last}(\alpha)$ and they have to be distinguished.

A **track resetting** Θ on P is given by:
- A function Θ_{OAP} from $\mathtt{supp}^{?}(P)$ to $\mathbb{N} - \{0, 1\}$ such that, for all $\alpha \in \mathbb{N}^*$, k', k" $\in \mathbb{N}$ s.t. $\alpha \cdot k'$, $\alpha \cdot k$" $\in \mathtt{supp}^{?}(P)$, $\Theta_{\mathrm{OAP}}(\alpha \cdot k) = \Theta_{\mathrm{OAP}}(\alpha \cdot k$") entails $k' = k$".
- An *injection* Θ_{Ax} from Ax to $\mathbb{N} - \{0, 1\}$.
- A function Θ_{rbp} from $\mathrm{rbp}(P)$ to $\mathbb{N} - \{0, 1\}$ such that, for all $\gamma \in \mathbb{N}^*$, k', k" $\in \mathbb{N}$ s.t. $(\alpha, \gamma \cdot k')$, $(\alpha, \gamma \cdot k$") $\in \mathrm{rbp}(P)$, $\Theta(\alpha, \gamma \cdot k') = \Theta(\alpha, \gamma \cdot k$") entails $k' = k$".

When such a Θ is given:
- We define a function $\Psi_{\mathtt{supp}} : A \to \mathbb{N}^*$, setting $\Psi_{\mathtt{supp}}(\varepsilon) = \varepsilon$, and when $\alpha \cdot k \in A$, $\Psi_{\mathtt{supp}}(\alpha \cdot k) = \Psi_{\mathtt{supp}}(\alpha) \cdot k$ if $k = 0$, 1 and and $\Psi_{\mathtt{supp}}(\alpha \cdot k) = \Psi_{\mathtt{supp}}(\alpha) \cdot \Theta_{\mathrm{OAP}}(\alpha \cdot k)$ if $k \geqslant 2$.
- We set $\Psi_{\mathrm{tr}} = \Theta_{\mathrm{Ax}}$.
- For all $\alpha \in \mathrm{Ax}$, we define a function $\Psi_{\alpha} : \mathtt{supp}(T(\alpha)) \to \mathbb{N}^*$ by induction, setting $\Psi_{\alpha}(\varepsilon) = \varepsilon$ and, when $\gamma \cdot k \in \mathtt{supp}(T(\alpha))$, $\Psi_{\alpha}(\gamma \cdot k) = \Psi_{\alpha}(\gamma) \cdot 1$ if $k = 1$ and $\Psi_{\alpha}(\gamma \cdot k) = \Psi_{\alpha}(\gamma) \cdot \Theta_{\alpha}(\gamma \cdot k)$ if $k \geqslant 2$..

Now, for all $\alpha \in \mathrm{Ax}$, we set $A^{\Theta} = \mathrm{codom}(\Psi_{\mathtt{supp}})$, we define Ax^{Θ} as the set of the leaves of A^{Θ} and $T^{\Theta}(\Psi_{\mathtt{supp}}(\alpha)) = \Psi_{\alpha}(T(\alpha))$ (see Def. **??**).

▶ **Lemma 4.** $\Psi_{\mathtt{supp}}$ is a 01-stable isomorphism from A to A^{Θ}.
- For all $\alpha \in \mathrm{Ax}$, the labelled tree $T^{\Theta}(\Psi_{\mathtt{supp}}(\alpha))$ is a type and Ψ_{α} is a type isomorphism from $T(\alpha)$ to $T^{\Theta}(\Psi_{\mathtt{supp}}(\alpha))$.

The lemma entails that Ψ is a bisupport setting function. We can reuse all the constructions and notations of the previous subsection In particular, it yields a new o.d. P^{Θ} such that Ψ is a o.d. isomorphism from P to P^{Θ}.

We can now define $\Theta_{\alpha} : \mathtt{supp}^{?}(T(\alpha)) \to \mathbb{N}-\{0\}$ for all $\alpha \in A$ by $\Theta_{\alpha}(\gamma{\cdot}k) = \mathrm{last}(\Psi_{\alpha}(\gamma{\cdot}k))$ and then, for all $\alpha \in A[@]$, we can set:
- For all $k \cdot \gamma \in \mathtt{supp}^{?}(\mathrm{L}(\alpha))$, $\Theta^{\mathrm{L}}_{\alpha}(k \cdot \gamma) = \mathrm{last}(\Psi^{\mathrm{L}}(k \cdot \gamma))$.
- For all $k \cdot \gamma \in \mathtt{supp}^{?}(\mathrm{R}(\alpha))$, $\Theta^{\mathrm{R}}_{\alpha}(k \cdot \gamma) = \mathrm{last}(\Psi^{\mathrm{R}}(k \cdot \gamma))$.

5.4 Referent Positions, Syntactic Polarity and Track Variables

We define now the **referent** $\mathrm{ref}(a,\, c)$ and its **(syntactic) polarity** $\mathrm{Pol}(a,\, c)$ for each mutable right biposition in P by $\searrow$-induction. Both referent biposition and syntactic polarity are related to how we build and write types in the hybrid construction.

- If a is an axiom leaf and $c \in \mathtt{supp}^{?}(T(a))$, $\mathrm{ref}(a,\, c) = (a,\, c) \in \mathrm{rbp}(P)$ and $\mathrm{Pol}(a,\, c) = \oplus$. If $c \notin \mathtt{supp}^{?}(T(a))$, $\mathrm{ref}(a,\, c)$ and $\mathrm{Pol}(a,\, c)$ are left undefined.
- If $a \cdot 0 \in A$ - say $t(\overline{a}) = \lambda x$ -, there are 2 cases:
  - If $c \in \mathtt{supp}^{?}(T(a \cdot 0))$, then $\mathrm{ref}(a,\, 1 \cdot c) = \mathrm{ref}(a \cdot 0,\, c)$ and $\mathrm{Pol}(a,\, 1 \cdot c) = \mathrm{Pol}(a \cdot 0,\, c)$.
  - For any $k \in \mathrm{Rt}(\mathrm{Tl}(T(a)))$, $\mathrm{ref}(a,\, k) = (a',\, \varepsilon) \in \mathrm{Ax} \times \{\varepsilon\}$, where $a' := \mathrm{pos}(a \cdot 0,\, x,\, k)$, $\mathrm{Pol}(a,\, k) = \ominus$ and, for all $c \in \mathtt{supp}^{?}(T(a'))$, $\mathrm{ref}(a,\, k \cdot c) = (a',\, c)$ and $\mathrm{Pol}(a,\, k \cdot c) = \ominus$
- If a is an application node, then $a \cdot 1 \in A$ and we set $\mathrm{ref}(a,\, c) = \mathrm{ref}(a \cdot 1,\, 1 \cdot c)$ and $\mathrm{Pol}(a,\, c) = \mathrm{Pol}(a \cdot 1,\, 1 \cdot c)$.

▶ **Remark 8.** Thus, the syntactic polarity of a mutable right biposition is negative when it was "produced" by an abstraction rule. It is positive when it was directly "created" in an axiom leaf.
- The referent of $(a,\, c)$ is usually a referent biposition, but it can also be an axiom position, because of the "root-case" for abstraction rules.

We can extend those definitions to the $\mathrm{L}(a)$ and $\mathrm{R}(a)$. The referent of an inner position in $\mathrm{R}(a)$ can be an argument position:

▶ **Definition 5.** Let $a \in A[@]$.
- For all $k{\cdot}c \in \mathtt{supp}^{?}(\mathrm{L}(a))$, we set $\mathrm{ref}^{\mathrm{L}}(a,\, c) = \mathrm{ref}(a{\cdot}1,\, k{\cdot}c)$ and $\mathrm{Pol}^{\mathrm{L}}(a,\, c) = \mathrm{Pol}(a{\cdot}1,\, k{\cdot}c)$.
- For all $k \in \mathrm{Rt}(\mathrm{R}(a))$, we set $\mathrm{ref}^{\mathrm{R}}(a,\, k) = a_k \in \mathrm{OAP}$ and $\mathrm{Pol}^{\mathrm{R}}$ is left undefined, and next, for all $c \in \mathtt{supp}^{?}(T(a))$, $\mathrm{ref}^{\mathrm{R}}(a,\, k \cdot c) = \mathrm{ref}(a \cdot k,\, c)$.

5.5 Resetting into a Trivial Derivation

Let Θ be a resetting for P. We use the same notations as in subsec. **??**. We have then:

▶ **Lemma 5.** For all $\alpha \in A$ and $\gamma \in \mathtt{supp}^{?}(T(a))$, there are two cases:

- If $\mathrm{ref}(\alpha,\,\gamma) = (\eta,\,\zeta)$, then $\Theta_\alpha(\gamma) = \Theta_\eta(\zeta)$.
- If $\mathrm{ref}(\alpha,\,\gamma) = \eta$, then $\Theta_\alpha(\gamma) = \Theta_{\mathrm{tr}}(\alpha)$.

Proof. By $\searrow$-induction on α. ◀

▶ **Lemma 6.** For all $\alpha \in A[\alpha]$.
- If $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k\cdot\gamma) = (\eta,\,\zeta)$, then $\Theta^{\mathrm{L}}_\alpha(k\cdot\gamma) = \Theta_\eta(\zeta)$ and if $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k\cdot\gamma) = \eta$, then $\Theta^{\mathrm{L}}_\alpha(k\cdot\gamma) = \Theta_{\mathrm{tr}}(\eta)$.
- We have $\mathrm{ref}^{\mathrm{R}}(\alpha,\,k) = a\cdot k$ and when $\gamma \neq \varepsilon$, if $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k\cdot\gamma) = (\eta,\,\zeta)$, then $\Theta^{\mathrm{L}}_\alpha(\gamma) = \Theta_\eta(\zeta)$ and if $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k\cdot\gamma) = \eta$, then $\Theta^{\mathrm{L}}_\alpha(\gamma) = \Theta_{\mathrm{tr}}(\eta)$.

We observe next that:
- The o.d. P^Θ is trivial iff, for all $\alpha \in A[@]$ and $k\cdot c \in \mathtt{supp}(\mathrm{L}(k\cdot\alpha))$, $\Psi^{\mathrm{L}}(k\cdot\gamma) = \Psi^{\mathrm{R}}(\phi_\alpha(k\cdot\gamma))$.
- Let $\gamma_1,\ \gamma_2 \in \mathbb{N}^*$ such that $|\gamma_1| = |\gamma_2| = \ell$. We write $\gamma_i|_k$ for the length k prefix of γ_i. Then, $\gamma_1 = \gamma_2$ iff, for all $1 \leqslant k \leqslant \ell$, $\mathrm{last}(\gamma_1|_k) = \mathrm{last}(\gamma_2|_k)$.

Thus, P^Θ is trivial iff, for all $\alpha \in A[@],\ k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}} \in \mathtt{supp}(\mathrm{L}(\alpha)),\ k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}} \in \mathtt{supp}(\mathrm{R}(\alpha))$ s.t. $\phi_\alpha(k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) = k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}$, we have $\Theta^{\mathrm{L}}(k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) = \Theta^{\mathrm{R}}(k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}})$.

That, and the previous lemma, motivates:

▶ **Definition 6.**
- For all $\alpha \in A[@]$, we define a partial relation $\overset{\alpha}{\nrightarrow}$ on $\mathrm{ref}(P)$ by $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) \overset{\alpha}{\nrightarrow} \mathrm{ref}^{\mathrm{R}}(k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}})$ for any $(k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}},\,k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}) \in \phi_\alpha$.
- Moreover, if $\circledast, \circledcirc \in \{\oplus, \ominus\}$, we write $\overset{\circledast}{\underset{\alpha}{\nrightarrow}}$, $\overset{\alpha}{\nrightarrow}^{\circledcirc}$ and $\overset{\circledast}{\underset{\alpha}{\nrightarrow}}^{\circledcirc}$ for the restriction of $\overset{\alpha}{\nrightarrow}$ with the supplementary condition (in the previous definition) that $\mathrm{Pol}^{\mathrm{L}}(\alpha,\,k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) = \circledast$ or/and $\mathrm{Pol}^{\mathrm{R}}(\alpha,\,k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}) = \circledcirc$.

For instance, $r_{\mathrm{L}} \overset{\alpha}{\nrightarrow}^{\ominus} r_{\mathrm{R}}$ is equivalent to the existence of $k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}} \in \mathtt{supp}(\mathrm{L}(\alpha)),\ k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}} \in \mathtt{supp}(\mathrm{R}(\alpha))$ s.t. $\mathrm{ref}^{\mathrm{L}}(\alpha,\,k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) = r_{\mathrm{L}},\ \mathrm{ref}^{\mathrm{R}}(\alpha,\,k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}) = r_{\mathrm{R}},\ \phi(k_{\mathrm{L}}\cdot\gamma_{\mathrm{L}}) = k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}$ and $\mathrm{Pol}^{\mathrm{R}}(\alpha,\,k_{\mathrm{R}}\cdot\gamma_{\mathrm{R}}) = \ominus$.

Since $\mathrm{L}(\alpha)$ is either of the form $C(\alpha\cdot 0)(y)$ or $\mathrm{Tl}(\mathrm{Hd}^n(T(\alpha_0)))$, where α_0 is an axiom leaf, the relation $\overset{\alpha}{\nrightarrow}$ is functionnal and we write $\widetilde{\phi_\alpha}$ the underlying (partial) function from $\mathrm{ref}(P)$ to itself. Thus:

▶ **Lemma 7.** P^Θ is a trivial derivation iff, for all $\alpha \in A[@],\ r_{\mathrm{L}},\ r_{\mathrm{R}} \in \mathrm{ref}(P),\ \widetilde{\phi_\alpha}(r_{\mathrm{L}}) = r_{\mathrm{R}}$ implies $\Theta(r_{\mathrm{L}}) = \Theta(r_{\mathrm{R}})$ (where Θ can be $\Theta_{\mathrm{Ax}},\ \Theta_{\mathrm{OAP}}$ or Θ_{rbp} according to the respective nature of r_{L} and r_{L}).

5.6 Trivial Theory

We must now study the possibility of fulfil the conditions of the previous lemma. It turns out that the constraints defining a track resetting Θ yielding a trivial derivation can be described by a first order theory ranging on constants representing the value given by Θ on the referent tracks.

To each $r \in \mathrm{ref}(P)$, we attribute a variable X_r (representing the value of $\Theta(r)$, where Θ stands for $\Theta_{\mathrm{Ax}},\ \Theta_{\mathrm{OAP}}$ or Θ_{rbp} according to the nature of r).

Let $r_1,\ r_2 \in \mathrm{ref}(P)$. We say that r_1 and r_2 are **brother referents** if $r_1,\ r_2 \in \mathrm{Ax}(x)$ for some $x \in \mathcal{V}$ or if $r_1 = \alpha\cdot k_1,\ r_2 = \alpha\cdot k_2$ with $\alpha \in A[@]$ and $k_1 \neq k_2$ or if $r_1 = (\alpha,\,\gamma\cdot k_2),\ r_2 =$

$(\alpha, \gamma \cdot k_2)$ with $k_1 \neq k_2$.

The track soundess theory on P is the conjunction of the $X_{r_1} \neq X_{r_2}$ for any brother referents $r_1,\ r_2 \in \mathrm{ref}(P)$. It states that two brother referents tracks cannot be mapped onto the same value.

For any $\alpha \in A[\alpha]$, the equality theory on P at pos. α is the conjunction of the $X_{r_\mathrm{L}} = X_{r_\mathrm{R}}$, for any referents $r_\mathrm{L},\ r_\mathrm{R}$ s.t. $\widetilde{\phi_\alpha}(r_\mathrm{L}) = r_\mathrm{R}$. It states that Θ must satisfy the conditions of the previous lemma at α.

The equality theory on P is the conjuction of all equality theories at pos. α, when α spans over $A[\alpha]$.

The **trivial theory** on P is the conjuction the track soundess theory and the equality theory. Notice that if the equality theory on P doesn't prove any equality of the form $X_{r_1} = X_{r_2}$ where r_1 and r_2 are two brother referents, then the trivial theory on P is consistent.

▶ **Lemma 8.** If the trivial theory on P is consistent, P is isomorphic (as an o.d.) to a trivial derivation.

Proof. Assume the trivial theory is consistent. We define an equivalence relation on $\mathrm{ref}(P)$ by: $r_1 \equiv r_2$ iff the trivial theory proves $X_{r_1} = X_{r_2}$.

Let $\mathrm{Val} :\ \mathrm{ref}(P)/\equiv \longrightarrow \mathbb{N} - \{0,\ 1\}$ be an injection from the quotient set $\mathrm{ref}(P)/\equiv$ to the set of integers greater than 1.

Then we can define a track resetting Θ by $\Theta(r) = \mathrm{Val}(\overline{r})$, where $\overline{r}$ is the equivalence class of $r \in \mathrm{ref}(P)$ and Θ stands for $\Theta_{\mathrm{Ax}},\ \Theta_{\mathrm{OAP}}$ or Θ_{rbp} according to the nature of r.

The definition is licit, because Θ satisfies the track soundess theory it defines a trivial derivation P^Θ, thanks to lemma **??**. ◀

6 Representation Theorem

We intend to prove here the following, using the observation of the former section:

▶ **Theorem 1.** Every operable derivation is isomorphic to a trivial derivation.

Since lemma **??** grants that every way of performing finite or not sequences of subject-reduction can be built-in inside an o.d., it etablishes that the "rigid" framework `S` does not cause any loss of expressivity compared to type system $\mathcal{M}$, which resorts to multisets.

6.1 Consumption

For all $\alpha \in A[@]$, we write $\overset{\alpha}{\nleftarrow}$ for the symmetric relation of $\overset{\alpha}{\nrightarrow}$ and $\overset{a}{\nleftrightarrow}$ for $\overset{\alpha}{\nrightarrow} \cup \overset{\alpha}{\nleftarrow}$. We do likewise for the "polarized" restrictions of $\overset{\alpha}{\nleftarrow}$. We observe next that:

▶ **Lemma 9.** If $r \in \mathrm{Ax} \times \{\varepsilon\} \cup \mathrm{OAP}$, there is at most one $\alpha \in A[@]$ such that $\exists r' \in \mathrm{ref}(P),\ r \overset{\alpha}{\nleftrightarrow} r'$.
In that case, either $\exists! r' \in \mathrm{ref}(P),\ r \overset{\alpha}{\nrightarrow} r'$, either $\exists! r' \in \mathrm{ref}(P),\ r \overset{\alpha}{\nleftarrow} r'$.

- If $r \in \mathrm{rbp}(P)$ and $\circledast \in \{\oplus, \ominus\}$, there is at most one $\alpha \in A[@]$ such that $\exists r' \in \mathrm{ref}(P),\ r^{\circledast} \overset{\alpha}{\nrightarrow} r'$.
  In that case, either $\exists! r' \in \mathrm{ref}(P),\ r^{\circledast} \overset{\alpha}{\to} r'$, either $\exists! r' \in \mathrm{ref}(P),\ r^{\circledast} \overset{\alpha}{\leftarrow} r'$.
  We say then that r (resp. $r^{\circledast}$) is **consumed** at pos. α.

Moreover:

▶ **Lemma 10.** If $\circledast \in \{\oplus, \ominus\}$ and r_1 and r_2 are brother referents (that are not in OAP), then $r_1^{\circledast}$ is consumed at pos. α iff r_2 is.
In the case where $r_i \in \mathrm{dom}\,\widetilde{\phi_\alpha}$, $\widetilde{\phi_\alpha}(r_1)$ and $\widetilde{\phi_\alpha}(r_2)$ are brother referents.

6.2 Residuals of Referents

Assume $t \xrightarrow{b} t'$ with $b \in \mathtt{supp}(t)$.
We write $P',\ A',\ \mathrm{pos}',\ \mathrm{L}',\ \mathrm{R}',\ \mathrm{rbp}',\ \mathrm{ref}',\ \phi'$ for the usual notions defined w.r.t. t'.

We define the residual $\widetilde{\mathrm{Res}_b}(r) \in \mathrm{ref}'(P')$ of any $r \in \mathrm{ref}(P)$, *except* when $r = (\alpha,\ \varepsilon) \in \mathrm{Ax} \times \{\varepsilon\}$ with $\alpha \in \mathrm{Ax}(x)$ or $r = \alpha \in \mathrm{OAP}$ with $\overline{\alpha} = b \cdot 2$.

- If $\alpha \in \mathrm{Ax} - \mathrm{Ax}(x)$, then $\widetilde{\mathrm{Res}_b}(\alpha,\ \varepsilon) = (\alpha',\ \varepsilon)$, with $\alpha' = \mathrm{Res}_b(\alpha)$.
- If $(\alpha,\ \gamma) \in \mathrm{rbp}(P)$ (with $\gamma \neq \varepsilon$), there are two subcases:
  - Subcase $\alpha \notin \mathrm{Ax}(x)$: we set $\widetilde{\mathrm{Res}_b}(\alpha,\ \gamma) = \mathrm{Res}_b(\alpha,\ \gamma)$, *i.e.* $\widetilde{\mathrm{Res}_b}(\alpha,\ \gamma) = (\alpha',\ \gamma)$, with $\alpha' = \mathrm{Res}_b(\alpha)$.
  - Subcase $\alpha \in \mathrm{Ax}(x)$: we set $\widetilde{\mathrm{Res}_b}(\alpha,\ \gamma) = \mathrm{ref}'(\mathrm{QRes}_b(\alpha,\ \gamma))$, *i.e.* $\widetilde{\mathrm{Res}_b}(\alpha,\ \gamma) = \mathrm{ref}'(a \cdot a_{k_L},\ k_R \cdot \gamma_R)$ where $\overline{a} = b,\ \alpha = a \cdot a_{k_L},\ k_R \cdot \gamma_R = \widetilde{\phi_a}(k_L \cdot \gamma_L)$.
- If $\alpha \cdot k \in \mathrm{OAP}$, then $\widetilde{\mathrm{Res}_b}(\alpha \cdot k) = \mathrm{Res}_b(\alpha \cdot k)$, *i.e.* $\widetilde{\mathrm{Res}_b}(\alpha \cdot k) = \alpha' \cdot k$ where $\alpha' = \mathrm{Res}_b(\alpha)$.

▶ **Lemma 11.** Let $(\alpha,\ k \cdot \gamma)$ be in ??? Then $\mathrm{rbp}'(\mathrm{QRes}_b(\alpha,\ \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{rbp}(\alpha,\ \gamma))$ (one side of this equality is defined iff the other one is).

▶ **Lemma 12.** Assume $y \neq x$ is free at pos. $\alpha \in A$ and $\overline{\alpha} \neq b \cdot 1$.
Then $\mathrm{Res}_b(\mathrm{pos}(\alpha,\ y,\ k) = \mathrm{pos}'(\mathrm{QRes}_b(\alpha),\ y,\ k)$.

Proof. It stems immediately from the monoticity of QRes_b w.r.t. the prefix order and the fact that for any $\alpha \in A$ s.t. y is free at pos. α, and $k \geqslant 2$, there is at most one $\alpha_0 \in \mathrm{Ax}(\alpha_0)(y)$ s.t. $\mathrm{tr}(\alpha_0) = k$. ◀

▶ **Lemma 13.** Let $(\alpha,\ \gamma)$ a right ?-biposition s.t. $\overline{\alpha} \neq b \cdot 1$.
Then $\mathrm{ref}'(\mathrm{QRes}_b(\alpha,\ \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha,\ \gamma))$.

Proof. By $\searrow$-induction on α.

- When $\alpha \in \mathrm{Ax}$, it stems from the definition of $\widetilde{\mathrm{Res}_b}$ (whether $t(\overline{\alpha}) = x$ or not).

- Assume $t(\overline{\alpha}) = \lambda y$. Since $\overline{\alpha} \neq b \cdot 1$, $y \neq x$, $\alpha' = \mathrm{Res}_b(\alpha)$ is defined, $y \neq x$ and $t'(\overline{\alpha'}) = \lambda y$
  - Subcase $1 \cdot \gamma$: we write $\mathrm{Res}_b(\alpha,\ 1 \cdot \gamma) = (\alpha',\ 1 \cdot \gamma')$ (γ' is the unique pos. s.t. $1 \cdot \gamma' = \mathrm{Res}_{b|\alpha}(\gamma')$), s.t. $\mathrm{QRes}_b(\alpha \cdot 0,\ 1 \cdot \gamma) = (\alpha' \cdot 0,\ \gamma')$.
    By definition, $\mathrm{ref}(\alpha,\ 1 \cdot \gamma) = \mathrm{ref}(\alpha \cdot 0,\ \gamma)$ and $\mathrm{ref}'(\alpha',\ 1 \cdot \gamma') = \mathrm{ref}(\alpha' \cdot 0,\ \gamma')$, By IH, $\mathrm{ref}'(\mathrm{QRes}_b(\alpha \cdot 0,\ \gamma) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha \cdot 0,\ \gamma)$. So $\mathrm{ref}'(\alpha' \cdot 0,\ \gamma') = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha \cdot 0,\ \gamma)$. Thus, $\mathrm{ref}'(\alpha',\ 1 \cdot \gamma') = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha,\ 1 \cdot \gamma)$.

- Subcase $k \cdot \gamma$ with $k \in \mathrm{AxTr}(\alpha \cdot 0)(y)$: we set $\eta = \mathrm{pos}(\alpha \cdot 0,\, y,\, k)$, s.t. $\mathrm{ref}(\alpha,\, k \cdot \gamma) = (\eta,\, \gamma)$ for any $\gamma \in \widetilde{\mathtt{supp}^?}(T(\alpha)) \cup \{\varepsilon\}$.
  Since $y \neq x$, we have $\widetilde{\mathrm{Res}_b}(\eta,\, \gamma) = (\eta',\, \gamma)$ where $\eta' = \mathrm{Res}_b(\eta)$, whether $\gamma = \varepsilon$ or not.
  On the left side, $\mathrm{Res}_b(\alpha,\, k \cdot \gamma) = (\alpha',\, k \cdot \gamma)$ (by definition of $\mathrm{Res}_{b|\alpha}$) and $\mathrm{ref}'(\alpha',\, k \cdot \gamma) = (\mathrm{pos}'(\alpha' \cdot 0,\, y,\, k \cdot \gamma) = (\eta',\, \gamma)$, according to the previous lemma. The equality is proven.

- Assume $t(\overline{\alpha}) = @$. We set $(\alpha',\, \gamma') = \mathrm{QRes}_b(\alpha,\, \gamma)$.
  - Subcase $\overline{\alpha} \neq b$: since $\overline{\alpha} \neq b$, we have actually $\mathrm{Res}_b(\alpha) = \alpha'$, $t'(\alpha') = @$ and $\mathrm{QRes}_b(\alpha \cdot 1,\, 1 \cdot \gamma) = (\alpha' \cdot 1,\, 1 \cdot \gamma')$ by definition of QRes_b.
    By definition of referents, $\mathrm{ref}(\alpha,\, \gamma) = \widetilde{\mathrm{ref}}(\alpha \cdot 1,\, 1 \cdot \gamma)$ and $\mathrm{ref}'(\alpha',\, \gamma') = \mathrm{ref}'(\alpha' \cdot 1,\, 1 \cdot \gamma')$.
    By IH, $\mathrm{ref}'(\mathrm{QRes}_b(\alpha \cdot 1,\, 1 \cdot \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha \cdot 1,\, 1 \cdot \gamma)$. So $\mathrm{ref}'(\mathrm{QRes}_b(\alpha,\, \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{rbp}(\alpha,\, \gamma))$
  - Subcase $\overline{\alpha} = b$: by IH, $\mathrm{ref}'(\mathrm{QRes}_b(\alpha \cdot 10,\, \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}(\alpha \cdot 10,\, \gamma)$ and moreover, $\mathrm{ref}(\alpha,\, \gamma) = \mathrm{ref}(\alpha \cdot 10,\, \gamma)$.
    On the left side, $\mathrm{QRes}_b(\alpha \cdot 10,\, \gamma) = (\alpha,\, \gamma') = \mathrm{QRes}_b(\alpha,\ \gamma)$. The equality is proven.

◀

▶ **Corollary 1.** If $\alpha \in A[@]$ and $\overline{\alpha} \neq b$, for all $k \cdot \gamma \in \mathrm{dom}\,\mathrm{L}(\alpha)$ (resp. $k \cdot \gamma \in \mathrm{R}(\alpha)$), $(\mathrm{ref}^{\mathrm{L}})'(\mathrm{ResL}_b(k \cdot \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}^{\mathrm{L}}(k \cdot \gamma))$ (resp. $(\mathrm{ref}^{\mathrm{R}})'(\mathrm{ResR}_b(k \cdot \gamma)) = \widetilde{\mathrm{Res}_b}(\mathrm{ref}^{\mathrm{R}}(k \cdot \gamma))$).

Proof. Set $\alpha' = \mathrm{Res}_b(\alpha)$ (since $\overline{\alpha} \neq @$).
If $k \in \mathrm{Arg}(\alpha)$, then $\mathrm{ref}^{\mathrm{R}}(k) = \alpha \cdot k$, $\mathrm{ResR}_b(k) = k$ (by definition of ResR) and $(\mathrm{ref}^{\mathrm{R}})'(k) = \alpha' \cdot k$. So the equality is true.
Every other case immediately comes from the previous lemma. ◀

6.3 Collapsing Referents

▶ **Lemma 14.** Assume $r_{\mathrm{L}} \overset{\alpha}{\nrightarrow} r_{\mathrm{R}}$.

- If $\alpha \notin \mathrm{Rep}(b)$, then $\alpha' := \mathrm{Res}_b(\alpha)$ is defined and $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}}) \overset{\alpha'}{\nrightarrow} \widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$ (the relation $\overset{\alpha'}{\nrightarrow}$ is meant w.r.t. P' instead of P).
- If $\alpha \in \mathrm{Rep}(b)$, then $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}})$ is defined iff $\widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$.
  In that case, $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}}) = \widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$.

Proof. The hypothesis means that there are $k_{\mathrm{L}} \cdot \gamma_{\mathrm{L}},\ k_{\mathrm{R}} \cdot \gamma_{\mathrm{R}}$ s.t. $\mathrm{ref}^{\mathrm{L}}_{\alpha}(k_{\mathrm{L}} \cdot \gamma_{\mathrm{L}}) = r_{\mathrm{L}},\ \mathrm{ref}^{\mathrm{R}}_{\alpha}(k_{\mathrm{R}} \cdot \gamma_{\mathrm{R}}) = r_{\mathrm{R}}$ and $\phi_\alpha(k_{\mathrm{L}} \cdot \gamma_{\mathrm{L}}) = k_{\mathrm{R}} \cdot \gamma_{\mathrm{R}}$.

- Case $\overline{\alpha} \neq b$: consequence of the definition of the residual interface § **??**, of Definition **??** and of Lemma **??**.
- Case $\overline{\alpha} = b$: we write a instead of α and $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}})$ is not defined iff $r_{\mathrm{L}} = (\eta,\, \varepsilon)$ with $\eta \in \mathrm{Ax}(\alpha \cdot 10)(x)$ and $\widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$ is not defined iff $r_{\mathrm{R}} \in \mathrm{OAP}(a)$. Moreover, $\widetilde{\phi_\alpha}$ induces a bijection from $\mathrm{AxTr}(a \cdot 10)(x)$ to $\mathrm{OAP}(a \cdot 1)$, so the first part is proven.
  We assume now that $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}})$ and $\widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$ are defined. We must have $r_{\mathrm{L}} = (a \cdot 10 \cdot a_{k_{\mathrm{L}}}$ and $\zeta_{\mathrm{L}} = \gamma_{\mathrm{L}}$.
  Thus, $\widetilde{\mathrm{Res}_b}(r_{\mathrm{L}}) = \widetilde{\mathrm{Res}_b}(a \cdot 10 \cdot a_{k_{\mathrm{L}}},\, \gamma_{\mathrm{L}})) = \mathrm{ref}'(\mathrm{QRes}_b(a \cdot 10 \cdot a_{k_{\mathrm{L}}},\, \gamma_{\mathrm{L}})) = \mathrm{ref}'(a \cdot a_{k_{\mathrm{L}}},\, \gamma_{\mathrm{R}}) = \mathrm{ref}'(\mathrm{Res}_b(a \cdot k_{\mathrm{R}},\, \gamma_{\mathrm{R}})) = \mathrm{Res}_b(\mathrm{ref}(a \cdot k_{\mathrm{R}},\, \gamma_{\mathrm{R}})) = \widetilde{\mathrm{Res}_b}(r_{\mathrm{R}})$.

◀

Assume we have $r_{\mathrm{L}}{}^{\ominus} \overset{a}{\nrightarrow} r_{\mathrm{R}}$ for some $r_{\mathrm{L}},\, r_{\mathrm{R}} \in \mathrm{ref}(P)$. Thus, $r_{\mathrm{L}} = (\eta,\ \gamma)$ with $\eta \in \mathrm{Ax}(y)$ and $\gamma \in \widetilde{\mathtt{supp}^?}(T(\eta)) \cup \{\varepsilon\}$ (for some $x \in \mathcal{V}$). There is a unique $\alpha \in {}^{\searrow}\eta$ s.t. $t(\overline{\alpha}) = \lambda x$ and $a < \alpha$.
We define the **collapsing strategy w.r.t.** $r_{\mathrm{L}}^{\ominus}$ by induction on $h := |\alpha| - |a|$:

- Case $h = 1$: there is a redex in t at pos. $\overline{a}$. We fire it and the strategy is completed.
- Case $h > 1$: since $r_{\mathrm{L}}{}^{\ominus} \overset{\alpha}{\nrightarrow}$, there is a (maximal) $a_0 \in {}^{\searrow}\alpha$ s.t. $t(\overline{a_0}) = @$. We can assume a_0 is maximal in length: it entails that $a_0 \cdot 0 \in A$ *i.e.* $t|_{\overline{a_0}}$ is a redex. We fire it, so that the height h decreases, and we go on with the strategy.

Let $\mathcal{R}(h)$ be the sequence of reductions representing the collapsing strategy. The former lemma entails that $\widetilde{\mathrm{Res}}_{\mathcal{R}(h)}(r_{\mathrm{L}}) = \widetilde{\mathrm{Res}}_{\mathcal{R}(h)}(r_{\mathrm{R}})$.

6.4 Do we have a Degenerate Equality Theory?

A proof of $X_{r'} = X_{r"}$ in the trivial theory, where r', $r" \in \mathrm{ref}(P)$ are brother referents, must rely on a sequence of relations $r' = r_1 \overset{\alpha_1}{\leftrightarrow} r_2 \overset{\alpha_2}{\leftrightarrow} r_3 \overset{\alpha_3}{\leftrightarrow} \dots \overset{\alpha_{\ell-1}}{\leftrightarrow} r_\ell = r"$.

For the sake of contradiction, we assume there is such a sequence. It allows to assume the length ℓ of the previous chain to be minimal among all the possible o.d.

- Due to the Consumption Lemma **??**, no r_i can be in OAP or in $\mathrm{Ax} \times \{\varepsilon\}$ on the negative side of an arrow: it would entail that $\ell = 2$ and that the considered chain is $r_1^{\oplus} \overset{a}{\nrightarrow} r_2$ with $r_1 \in \mathrm{Ax} \times \varepsilon$ and $r_2 \in \mathrm{OAP}$ (or the symmetric case) and therefore, not contradictory. It also shows that every r_i (for $1 \leqslant i \leqslant 2$) has a syntactic polarity.

- Because of the collapsing strategy, no referent of negative polarity can occur on the negative side of an arrow: in that case, we could decrease the length of the chain, contradicting its minimality.

- Due to the Consumption Lemma, we cannot have $r_{i-1} \overset{\alpha_{i-1}}{\leftrightarrow} {}^{\oplus} r_i {}^{\oplus} \overset{\alpha_i}{\leftrightarrow} r_{i+1}$ for $2 \leqslant i \leqslant \ell - 1$, because it would entail $\alpha_{i-1} = \alpha_i$ and $r_{i-1} = r_{i+1}$ the chain would not be minimal.

- Thus, for each $2 \leqslant i \leqslant \ell - 1$, we have 4 kinds of possible "crossing": $r_{i-1}{}^{\oplus} \overset{\alpha_{i-1}}{\nrightarrow} {}^{\ominus} r_i {}^{\oplus} \overset{\alpha_i}{\nrightarrow} r_{i+1}$ (kind r) or $r_{i-1} \overset{\alpha_{i-1}}{\nleftarrow} {}^{\oplus} r_i {}^{\ominus} \overset{\alpha_i}{\nleftarrow} {}^{\oplus} r_{i+1}$ (kind l) or $r_{i-1}{}^{\oplus} \overset{\alpha_{i-1}}{\nrightarrow} {}^{\ominus} r_i {}^{\oplus} \overset{\alpha_i}{\nrightarrow} r_{i+1}$ (kind r) or $r_{i-1}{}^{\oplus} \overset{\alpha_{i-1}}{\nrightarrow} {}^{\ominus} r_i {}^{\oplus} \overset{\alpha_i}{\nleftarrow} {}^{\oplus} r_{i+1}$ (kind mp) or $r_{i-1}{}^{\oplus} \overset{\alpha_{i-1}}{\nrightarrow} {}^{\oplus} r_i {}^{\ominus} \overset{\alpha_i}{\nleftarrow} {}^{\oplus} r_{i+1}$ (kind mp)

- If there were no "crossing" of kind mp or pm, either every "crossing" would be of kind r, or every "crossing" would be of kind l. By symmetry, we can assume it is of kind r. The lemma below entails that $\mathrm{d}_{001}(r_i)$ is monotonic. Since brother referent have the same (underlying) 001-depth, except when they are in $\mathrm{Ax} \times \{\varepsilon\}$, it entails that $r_1,\ r_\ell \in \mathrm{Ax} \times \{\varepsilon\}$.
  Since a referent of $\mathrm{Ax} \times \{\varepsilon\}$ has a negative polarity, r_1 occurs negatively at the negative side of $\overset{\alpha_1}{\nrightarrow}$. As seen above, it cannot happen.
  ▶ **Lemma 15.** If $(\eta_1, \zeta_1)^{\oplus} \overset{\alpha}{\nrightarrow} (\eta_2, \zeta_2)$, then $\mathrm{ad}(\eta_1) < \mathrm{ad}(\eta_2))$.

  Proof. The positive polarity entails $\mathrm{ad}(\eta_1) = \mathrm{ad}(\alpha)$ and $\overline{\alpha} \cdot 2 \leqslant \overline{\eta_2}$ entails $\mathrm{ad}(\eta_1) \geqslant \mathrm{ad}(\alpha) + 1$.
  ◀

- So there is a crossing of kind pm or of kind mp. The can only be one. By symmetry, we can always assume that it is of kind mp. On its left, the crossings are of kind r and on

its right of kind l.
All the r_i are thus referent *bipositions* and the chain starts with $r_1 \overset{\oplus}{\ominus} \overset{\alpha_1}{\rightarrow} \overset{\ominus}{} r_2$ and ends with $r_{\ell-1} \overset{\ominus}{} \overset{\alpha_{\ell-1}}{\leftarrow} \overset{\oplus}{} r_\ell$.
Since r_1 and r_ℓ are brother referent bipositions, there are $\eta,\ \zeta \in \mathbb{N}^*$, $k_1, k_\ell \geqslant 2$ s.t. $r_1 = (\eta,\ \zeta \cdot k_1)$, $r_\ell = (\eta, \zeta \cdot k_\ell)$. Due to lemma **??**, $\alpha_1 = \alpha_{\ell-1}$ and r_2 and $r_{\ell-1}$ are also brother bipositions. It contradicts the minimality of the chain and it completes the proof of the representability theorem.

References

1 Antonio Bucciarelli, Delia Kesner, and Simona Ronchi Della Rocca. The inhabitation problem for non-idempotent intersection types. In *Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings*, pages 341–354, 2014.

2 Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. Strong normalization through intersection types and memory. In *Proc. of the 10th Int. Workshop on Logical and Semantical Frameworks, with Applications (LSFA), ENTCS, Natal, Brazil, August-September 2015*.

3 Daniel De Carvalho. *Sémantique de la logique linéaire et temps de calcul*. PhD thesis, Université Aix-Marseille, November 2007.

4 Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980.

5 Richard Kennaway, Jan Willem Klop, M. Ronan Sleep, and Fer-Jan de Vries. Infinitary lambda calculus. *Theor. Comput. Sci.*, 175(1):93–125, 1997.

6 Pierre Vial. Unpublished work.

Contents