# NON-IDEMPOTENT TYPES FOR CLASSICAL CALCULI IN NATURAL DEDUCTION STYLE

DELIA KESNER AND PIERRE VIAL

IRIF (CNRS and Université Paris-Diderot), France
*e-mail address*: kesner@irif.fr

IRIF (CNRS and Université Paris-Diderot), France
*e-mail address*: pvial@irif.fr

Abstract. In the first part of this paper, we define two resource aware typing systems for the $\lambda\mu$-calculus based on non-idempotent *intersection* and *union* types. The non-idempotent approach provides very simple combinatorial arguments –based on decreasing measures of type derivations– to characterize head and strongly normalizing terms. Moreover, typability provides upper bounds for the lengths of the head-reduction and the maximal reduction sequences to normal-form.

In the second part of this paper, the $\lambda\mu$-calculus is refined to a small-step calculus called $\lambda\mu_{\mathtt{s}}$, which is inspired by the *substitution at a distance* paradigm. The $\lambda\mu_{\mathtt{s}}$-calculus turns out to be compatible with a natural extension of the non-idempotent interpretations of $\lambda\mu$, *i.e.* $\lambda\mu_{\mathtt{s}}$-reduction preserves and decreases typing derivations in an extended appropriate typing system. We thus derive a simple arithmetical characterization of strongly $\lambda\mu_{\mathtt{s}}$-normalizing terms by means of typing.

## 1. Introduction

The Curry-Howard Isomorphism is the well-known relationship between programming languages and logical systems. While Curry first introduced the analogy between Hilbert-style deductions and combinatory logic, Howard highlighted the one between simply typed lambda calculus and natural deduction. Both examples use *intuitionistic* logic. The extension of the Curry-Howard Isomorphism to *classical* logic took more than two decades, when Griffin [26] observed that Felleisen's $\mathcal{C}$ operator can be typed with the double-negation elimination. A major step in this field was done by Parigot [42], who proposed the $\lambda\mu$-calculus as a simple term notation for classical natural deduction proofs. The $\lambda\mu$-calculus is an extension of the simply typed $\lambda$-calculus that encodes usual *control operators* as the Felleisen's $\mathcal{C}$ operator mentioned so far. Other calculi were proposed since then, as for example Curien-Herbelin's $\overline{\lambda}\mu\tilde{\mu}$-calculus [14] based on classical sequent calculus.

The Curry-Howard correspondence has already contributed to the understanding of many aspects of programming languages by establishing a rich connection between logic and computation. However, there are still some crucial aspects of computation, like the use of resources (*e.g.* time and space), that still need to be logically understood in the

*Key words and phrases:* lambda-mu-calculus, classical logic, intersection types, normalization.

LOGICAL METHODS
IN COMPUTER SCIENCE

DOI:10.2168/LMCS-???

classical setting. Establishing the foundations of resource consumption is nowadays a big challenge facing the programming language community. It would lead to a new generation of programming languages and proof assistants, with a clean type-theoretic account of resource capabilities.

**From qualitative....** Several notions of type assignment systems for $\lambda$-calculus have been defined since its creation, including among others *simple types* and *polymorphic types*. However, even if polymorphic types are powerful and convenient in programming practice, they have several drawbacks. For example, it is not possible to assign a type to a term of the form $(\lambda z.\lambda y.y(z\,\mathtt{I})(z\,\mathtt{K}))(\lambda x.x\,x)$, where $\mathtt{I} = \lambda w.w$ and $\mathtt{K} = \lambda x.\lambda y.x$, which can be understood as a meaningful program specified by a terminating term. *Intersection* types, pioneered by Coppo and Dezani [12, 13], introduce a new constructor $\cap$ for types, allowing the assignment of a type of the form $((\sigma \Rightarrow \sigma) \cap \sigma) \Rightarrow \sigma$ to the term $\lambda x.xx$. The intuition behind a term $t$ of type $\tau_1 \cap \tau_2$ is that $t$ has both types $\tau_1$ and $\tau_2$. The symbol $\cap$ is to be understood as a mathematical intersection, so in principle, intersection type theory was developed by using *idempotent* ($\sigma \cap \sigma = \sigma$), *commutative* ($\sigma \cap \tau = \tau \cap \sigma$), and *associative* ($(\sigma \cap \tau) \cap \delta = \sigma \cap (\tau \cap \delta)$) laws.

Intersection types have been used as a *behavioural* tool to reason about several operational and semantic properties of programming languages. For example, a $\lambda$-term/program $t$ is strongly normalizing/terminating if and only if $t$ can be assigned a type in an appropriate intersection type assignment system. Similarly, intersection types are able to describe and analyze models of $\lambda$-calculus [6], characterize *solvability* [37], *head normalization* [37], *linear-head normalization* [29], and *weak-normalization* [37, 35] among other properties.

**.... to quantitative Intersection types:** This technology turns out to be a powerful tool to reason about *qualitative* properties of programs, but not about *quantitative* ones. Indeed, for example, there is a type system characterizing head normalization (*i.e.* $t$ is typable in this system if and only if $t$ is head normalizing) and which gives simultaneously a proof that $t$ is head-normalizing if and only if the head reduction strategy terminates on $t$. But the type system gives no information about the number of head-reduction steps that are necessary to obtain a head normal form. Here is where *non-idempotent* types come into play, thus making a clear distinction between $\sigma \cap \sigma$ and $\sigma$, because intuitively, using the resource $\sigma$ twice or once is not the same from the quantitative point of view. This change of perspective can be related to the essential spirit of Linear Logic [23], which removes the contraction and weakening structural rules in order to provide an explicit control of the use of logical resources, *i.e.* to give a full account of the number of times that a given proposition is used to derive a conclusion.

**The case of the $\lambda$-calculus:** Non-idempotent types were pioneered by Philippa Gardner [22], Assaf Kfoury [32]. But is Daniel de Carvalho [17] who first established in his PhD thesis a relation between the size of a typing derivation in a non-idempotent intersection type system for the lambda-calculus and the head/weak-normalization execution time of head/weak-normalizing lambda-terms, respectively. Relational models of $\lambda$-calculi based on non-idempotent types have been investigated by de Carvalho and Ehrhard in [17, 18, 21]. The results of de Carvalho are distilled in [18]. Non-idempotency is used to reason about the longest reduction sequence of strongly normalizing terms in both the lambda-calculus [8, 16, 9] and in different lambda-calculi with explicit substitutions [9, 29].

Non-idempotent types also appear in linearization of the lambda-calculus [32], type inference and inhabitation [33, 38, 10], different characterizations of solvability [41], verification of higher-order programs [40].

**The case of the $\lambda\mu$-calculus:** It is essential to go beyond the $\lambda$-calculus to focus on the challenges posed by the *advanced features* of modern higher-order programming languages and proof assistants. We want in particular to associate quantitative information to languages being able to express control operators, as they allow to enrich declarative programming languages with imperative features.

**Related works:** The non-idempotent intersection and union types for $\lambda\mu$-calculus that we present in this article can be seen as a quantitative refinement of Girard's translation of classical logic into linear logic. Different qualitative and/or quantitative models for classical calculi were proposed in [45, 48, 51, 3], thus limiting the characterization of operational properties to head-normalization. Intersection and union types were also studied in the framework of classical logic [36, 47, 34, 20], but no work addresses the problem from a quantitative perspective. Type-theoretical characterization of strong-normalization for classical calculi were provided both for $\lambda\mu$ [49] and $\bar{\lambda}\mu\tilde{\mu}$-calculus [20], but the (idempotent) typing systems do not allow to construct decreasing measures for reduction, thus a resource aware semantics cannot be extracted from those interpretations. Combinatorial strong normalization proofs for the $\lambda\mu$-calculus were proposed for example in [15], but they do not provide any explicit decreasing measure, and their use of structural induction on simple types does not work anymore with intersection types, which are more powerful than simple types as they do not only ensure termination but also characterize it. Upper bounds for the $\lambda\mu$-calculus are studied in [7] by passing through standard reduction and the non erasing $\lambda\mu I$-calculus. Different small step semantics for classical calculi were developed in the framework of neededness [5, 43], without resorting to any resource aware semantical argument.

**Contributions:** Our first contribution is the definition of a resource aware type system for the $\lambda\mu$-calculus based on non-idempotent *intersection* and *union* types. The non-idempotent approach provides very simple combinatorial arguments, only based on a decreasing *measure*, to characterize head and strongly normalizing terms by means of typability. Indeed, we show that for every typable term $t$ with type derivation $\Pi$, if $t$ reduces to $t'$, then $t'$ is typable with a type derivation $\Pi'$ such that the measure of $\Pi$ is strictly greater than that of $\Pi'$. In the well-known case of the $\lambda$-calculus, such a measure is simply based on the structure of type tree derivations and it is given by the number of its nodes, which strictly decreases along reduction. However, in the $\lambda\mu$-calculus, the creation of nested applications during $\mu$-reduction may increase the number of nodes of the corresponding type derivations, so that such a simple definition of measure is not decreasing anymore. We then need to also take into account the structure (*multiplicity*) of certain types appearing in the type derivations, thus ensuring an overall decreasing of the measure during reduction. This first result has been previously presented in [31].

The second contribution of this paper is the definition of a new small-step operational semantics for $\lambda\mu$, called $\lambda\mu_{\mathtt{s}}$, inspired from the *substitution at a distance paradigm* [2], which is compatible with the non-idempotent typing system characterizing strong normalization for $\lambda\mu$, in that the latter extends to $\lambda\mu_{\mathtt{s}}$. The operational semantics of $\lambda\mu_{\mathtt{s}}$ is linear, *i.e.* a single reduction step only implements substitution/replacement on one (free) occurrence

of some variable/name at a time. We then extend the typing system for $\lambda\mu$ characterizing strong normalization, so that the small-step reduction calculus $\lambda\mu_{\mathtt{s}}$ preserves (and decreases the size of) typing derivations. We generalize the type-theoretical characterization of strong normalization to this explicit classical calculus, thus particularly simplifying existing proofs of strong normalization for small-step operational semantics of classical calculi [44].

## 2. The $\lambda\mu$-Calculus

This section gives the syntax (Section 2.1) and the operational semantics (Section 2.2) of the $\lambda\mu-$calculus [42]. But before this we first introduce some preliminary general notions of rewriting that will be used all along the paper, and that are applicable to any system $\mathcal{R}$. We denote by $\to_{\mathcal{R}}$ the (one-step) reduction relation associated to system $\mathcal{R}$. We write $\to_{\mathcal{R}}^*$ for the reflexive-transitive closure of $\to_{\mathcal{R}}$, and $\to_{\mathcal{R}}^n$ for the composition of $n$-steps of $\to_{\mathcal{R}}$, thus $t \to_{\mathcal{R}}^n u$ denotes a finite $\mathcal{R}$-reduction sequence of length $n$ from $t$ to $u$. A term $t$ is in $\mathcal{R}$-normal form, written $t \in \mathcal{R}$-nf, if there is no $t'$ s.t. $t \to_{\mathcal{R}} t'$; and $t$ has an $\mathcal{R}$-normal form iff there is $t' \in \mathcal{R}$-nf such that $t \to_{\mathcal{R}}^* t'$. A term $t$ is said to be strongly $\mathcal{R}$-normalizing, written $t \in \mathcal{SN}(\mathcal{R})$, iff there is no infinite $\mathcal{R}$-sequence starting at $t$. When $\mathcal{R}$ is finitely branching and strongly $\mathcal{R}$-normalizing, $\eta_{\mathcal{R}}(t)$ denotes the maximal length of an $\mathcal{R}$-reduction sequence starting at $t$, we simply write $\eta(t)$ if $\mathcal{R}$ is clear from the context.

2.1. **Syntax.** We consider a countable infinite set of **variables** $x, y, z, \ldots$ and **continuation names** $\alpha, \beta, \gamma, \ldots$. The set of **objects** ($\mathcal{O}_{\lambda\mu}$), **terms** ($\mathcal{T}_{\lambda\mu}$) and **commands** ($\mathcal{C}_{\lambda\mu}$) of the $\lambda\mu$-calculus are given by the following grammars

$$\begin{array}{lllll} (\textbf{objects}) & o & ::= & t \mid c \\ (\textbf{terms}) & t, u, v & ::= & x \mid \lambda x.t \mid tu \mid \mu\alpha.c \\ (\textbf{commands}) & c & ::= & [\alpha]t \end{array}$$

We write $\mathcal{T}_{\lambda}$ for the the set of $\lambda$-terms, which is a subset of $\mathcal{T}_{\lambda\mu}$. We abbreviate $(\ldots((tu_1)u_2)\ldots u_n)$ as $tu_1\ldots u_n$ or $t\overline{u}$ when $n$ is clear from the context. The grammar extends $\lambda$-terms with two new constructors: commands $[\alpha]t$ and $\mu$-abstractions $\mu\alpha.c$. **Free** and **bound variables** of objects are defined as expected, in particular $\mathtt{fv}(\mu\alpha.c) := \mathtt{fv}(c)$ and $\mathtt{fv}([\alpha]t) := \mathtt{fv}(t)$. **Free names** of objects are defined as expected, in particular $\mathtt{fn}(\mu\alpha.c) := \mathtt{fn}(c) \setminus \{\alpha\}$ and $\mathtt{fn}([\alpha]t) := \mathtt{fn}(t) \cup \{\alpha\}$. **Bound names** are defined accordingly.

We work with the standard notion of **$\alpha$-conversion** *i.e.* renaming of bound variables and names, thus for example $[\delta](\mu\alpha.[\alpha](\lambda x.x))z \equiv [\delta](\mu\beta.[\beta](\lambda y.y))z$. **Substitutions** are (finite) functions from variables to terms specified by the notation $\{x_1/u_1, \ldots, x_n/u_n\}$ $(n \geq 0)$. **Application** of the **substitution** $\sigma$ to the object $o$, written $o\sigma$, may require $\alpha$-conversion in order to avoid capture of free variables/names, and it is defined as expected. **Replacements** are (finite) functions from names to terms specified by the notation $\{\alpha_1 /\!\!/ u_1, \ldots, \alpha_n /\!\!/ u_n\}$ $(n \geq 0)$. Intuitively, the operation $\{\alpha /\!\!/ u\}$ passes the term $u$ as an argument to any command of the form $[\alpha]t$, so that it *replaces* every occurrence of $[\alpha]t$ in a term by $[\alpha]tu$. Formally, the **application** of the **replacement** $\Sigma$ to the object $o$, written $o\Sigma$, may require $\alpha$-conversion in order to avoid the capture of free variables/names, and is defined as follows:

$$\begin{array}{rcl@{\qquad}rcl} x\{\alpha /\!\!/ u\} & := & x & (\lambda z.t)\{\alpha /\!\!/ u\} & := & \lambda z.t\{\alpha /\!\!/ u\} \\ ([\alpha]t)\{\alpha /\!\!/ u\} & := & [\alpha](t\{\alpha /\!\!/ u\})u & (tv)\{\alpha /\!\!/ u\} & := & t\{\alpha /\!\!/ u\}v\{\alpha /\!\!/ u\} \\ ([\gamma]t)\{\alpha /\!\!/ u\} & := & [\gamma]t\{\alpha /\!\!/ u\} & (\mu\gamma.c)\{\alpha /\!\!/ u\} & := & \mu\gamma.c\{\alpha /\!\!/ u\} \end{array}$$

For example, if $\mathtt{I} = \lambda z.z$, then $(x(\mu\alpha[\alpha]y)(\lambda z.zx))\{x/\mathtt{I}\} = \mathtt{I}(\mu\alpha[\alpha]y)(\lambda z.z\mathtt{I})$, and $([\alpha]x(\mu\beta.[\alpha]y))\{\alpha /\!\!/ \mathtt{I}\} = [\alpha](x\mu\beta.[\alpha]y\mathtt{I}))\mathtt{I}$.

2.2. **Operational Semantics.** We consider the following set of contexts:

$$\begin{array}{lrcl} \textbf{(term contexts)} & \mathtt{T} & := & \Box \mid \mathtt{T}\,t \mid t\,\mathtt{T} \mid \lambda x.\mathtt{T} \mid \mu\alpha.\mathtt{C} \\ \textbf{(command contexts)} & \mathtt{C} & := & [\alpha]\mathtt{T} \\ \textbf{(contexts)} & \mathtt{O} & := & \mathtt{T} \mid \mathtt{C} \end{array}$$

The hole $\Box$ can be replaced by a term: indeed, $\mathtt{T}[t]$ and $\mathtt{C}[t]$ denote the replacement of $\Box$ in the context by the term $t$.

The $\lambda\mu$-calculus is given by the set of objects introduced in Section 2.1 and the **reduction relation** $\to_{\lambda\mu}$, sometimes simply written $\to$, which is the closure by all contexts of the following rewriting rules:

$$\begin{array}{lcl} (\lambda x.t)u & \mapsto_\beta & t\{x/u\} \\ (\mu\alpha.c)u & \mapsto_\mu & \mu\alpha.c\{\alpha /\!\!/ u\} \end{array}$$

defined by means of the substitution and replacement application notions given in Section 2.1. A **redex** is a term of the form $(\lambda x.t)u$ or $(\mu\alpha.c)u$.

An alternative specification of the $\mu$-rule [4] is given by $(\mu\alpha.c)u \mapsto_\mu \mu\gamma.c\{\alpha /\!\!/ \gamma.u\}$, where $\{\alpha /\!\!/ \gamma.u\}$ denotes the **fresh replacement** meta-operation assigning $[\gamma](t\{\alpha /\!\!/ \gamma.u\})u$ to $[\alpha]t$ (thus changing the name of the command), in contrast to the standard replacement operation $\{\alpha /\!\!/ u\}$ introduced in Section 2.1. We remark however that the resulting terms $\mu\alpha.c\{\alpha /\!\!/ u\}$ and $\mu\gamma.c\{\alpha /\!\!/ \gamma.u\}$ are $\alpha$-equivalent; thus *e.g.* $\mu\alpha.([\alpha]x)\{\alpha /\!\!/ u\} = \mu\alpha.[\alpha]xu \equiv \mu\gamma.[\gamma]xu = \mu\gamma.([\alpha]x)\{\alpha /\!\!/ \gamma.u\}$. We will come back to this alternative definition of $\mu$-reduction in Section 7.

A simple example is given by the following $\lambda\mu$-reduction sequence:

$$\begin{array}{ll} (\lambda x.\mu\alpha.[\alpha]x(\lambda y.\mu\delta.[\alpha]y))\ \mathtt{I}\ \mathtt{I} & \to_\beta \\ (\mu\alpha.[\alpha](\mathtt{I}(\lambda y.\mu\delta.[\alpha]y)))\mathtt{I} & \to_\beta \\ (\mu\alpha.[\alpha](\lambda y.\mu\delta.[\alpha]y))\mathtt{I} & \to_\mu \\ \mu\alpha.[\alpha](\lambda y.\mu\delta.[\alpha]y\mathtt{I})\mathtt{I} & \to_\beta \\ \mu\alpha.[\alpha](\mu\delta.[\alpha]\mathtt{I}\mathtt{I}) & \to_\beta \\ \mu\alpha.[\alpha](\mu\delta.[\alpha]\mathtt{I}) & \end{array}$$

Another typical example, given by Parigot [42], which illustrates the expressivity of the $\lambda\mu$-calculus is the control operator **call-cc** [26], coming from `Scheme` and enabling backtracking, specified in the $\lambda\mu$-calculus by the term $\lambda x.\mu\alpha.[\alpha]x(\lambda y.\mu\delta.[\alpha]y)$. The term **call-cc** is assigned $((A \to B) \to A) \to A$ (Peirce's Law) in the simply typed $\lambda\mu$-calculus.

A reduction step $o \to o'$ is said to be **erasing** iff $o = \mathtt{O}[(\lambda x.u)v]$ and $x \notin \mathtt{fv}(u)$ and $o' = \mathtt{O}[u\{x/v\}] = \mathtt{O}[u]$, or $o = \mathtt{O}[(\mu\alpha.c)u]$ and $\alpha \notin \mathtt{fn}(c)$ and $o' = \mathtt{O}[\mu\alpha.c\{\alpha /\!\!/ u\}] = \mathtt{O}[\mu\alpha.c]$. Thus *e.g.* $(\lambda x.z)y \to_\beta z$ and $(\mu\alpha.[\beta]x)\mathtt{I} \to_\mu \mu\alpha.[\beta]x$ are erasing steps. A reduction step $o \to o'$ which is not erasing is called **non-erasing**. Note that reduction is stable by substitution and replacement.

A **head-context** is a context defined by the following grammar:

$$\begin{array}{lcl} \mathtt{HO} & ::= & \mathtt{HT} \mid \mathtt{HC} \\ \mathtt{HT} & ::= & \Box\, t_1 \dots t_n\ (n \geqslant 0) \mid \lambda x.\mathtt{HT} \mid \mu\alpha.\mathtt{HC} \\ \mathtt{HC} & ::= & [\alpha]\mathtt{HT} \end{array}$$

A **head-normal form** is an object of the form $\mathtt{HO}[x]$, where $x$ is any variable replacing the constant $\Box$. Thus for example $\mu\alpha.[\beta]\lambda y.x(\lambda z.z)$ is a head-normal form. An object $o \in \mathcal{O}_{\lambda\mu}$ is

said to be **head-normalizing**, written $o \in \mathcal{HN}(\lambda\mu)$, if $o \to^*_{\lambda\mu} o'$, for some head-normal form $o'$. Remark that $o \in \mathcal{HN}(\lambda\mu)$ does not imply $o \in \mathcal{SN}(\lambda\mu)$ while the converse necessarily holds. We write $\mathcal{HN}(\lambda)$ and $\mathcal{SN}(\lambda)$ when $t$ is restricted to be a $\lambda$-term and the reduction system is restricted to the $\beta$-reduction rule.

A redex $r$ in an object of the form $o = \mathtt{HO}[r]$ is called the **head-redex** of $t$. The reduction step $o \to_{\lambda\mu} o'$ contracting the head-redex of $o$ is called a **head-reduction step**. The **head reduction strategy** is a deterministic strategy on the set $\mathcal{O}_{\lambda\mu}$ performing always head-steps, so that the head reduction strategy only stops on head normal forms. If the head-strategy starting at $o$ terminates, then $o \in \mathcal{HN}(\lambda\mu)$, while the converse direction is not straightforward (*cf.* Theorem 4.4).

## 3. Quantitative Type Systems for the $\lambda$-Calculus

As mentioned before, our results rely on typability of $\lambda\mu$-terms in suitable systems with non-idempotent types. Since the $\lambda\mu$-calculus embeds the $\lambda$-calculus, we start by recalling the well-known [22, 17, 10] quantitative type systems for $\lambda$-calculus, called here $\mathcal{H}_\lambda$ and $\mathcal{S}_\lambda$. We then reformulate them, using a different syntactical view, resulting in the typing systems $\mathcal{H}'_\lambda$ and $\mathcal{S}'_\lambda$, that are subsumed by the formalisms we adopt in Section 4 for $\lambda\mu$.

We start by fixing a countable set of **base types** $a, b, c \ldots$, then we introduce two different categories of types specified by the following grammars:

$$\begin{array}{lllll} (\textbf{Intersection Types}) & \mathcal{I} & ::= & [\sigma_k]_{k\in K} \\ (\textbf{Types}) & \sigma, \tau & ::= & a \mid \mathcal{I} \Rightarrow \sigma \end{array}$$

An intersection type $[\sigma_k]_{k\in\{1..n\}}$ is a *multiset* that can be understood as a type $\sigma_1 \cap \ldots \cap \sigma_n$, where $\cap$ is associative and commutative, but *non-idempotent*. Thus, $[\,]$ is the empty intersection type. The **non-deterministic choice** *relation* $\mathcal{R}^*$ between intersection types is defined by $[\sigma_k]_{k\in K}\mathcal{R}^*[\sigma_k]_{k\in K}$ when $K \neq \emptyset$ and $[\sigma_k]_{k\in K}\mathcal{R}^*[\tau]$ (for any type $\tau$) when $K = \emptyset$. By making a slight abuse of notation, this choice relation is going to be used as a non-deterministic *operation* $\_^*$ (rather than a relation $\mathcal{R}^*$) as follows:

$$[\sigma_k]^*_{k\in K} := \begin{cases} [\tau] & \text{if } K = \emptyset \text{ and } \tau \text{ is any arbitrary type} \\ [\sigma_k]_{k\in K} & \text{if } K \neq \emptyset \end{cases}$$

**Variable assignments** (written $\Gamma$) are total functions from variables to intersection types. We may write $\emptyset$ to denote the variable assignment that associates the empty intersection type $[\,]$ to every variable. The **domain of** $\Gamma$ is given by $\mathtt{dom}(\Gamma) := \{x \mid \Gamma(x) \neq [\,]\}$, so that when $x \notin \mathtt{dom}(\Gamma)$, then $\Gamma(x)$ stands for $[\,]$. We write $x : \mathcal{I}$ (even when $\mathcal{I} = [\,]$) for the assignment mapping $x$ to $\mathcal{I}$ and all $y \neq x$ to $[\,]$. We write $\Gamma \wedge \Gamma'$ for $x \mapsto \Gamma(x) + \Gamma'(x)$, where $+$ is multiset union, so that $\mathtt{dom}(\Gamma \wedge \Gamma') = \mathtt{dom}(\Gamma) \cup \mathtt{dom}(\Gamma')$. When $\mathtt{dom}(\Gamma) \cap \mathtt{dom}(\Gamma')$, then $\Gamma \wedge \Gamma'$ may also be written as $\Gamma; \Gamma'$. We write $\Gamma \setminus\!\setminus x$ for the assignment defined by $(\Gamma \setminus\!\setminus x)(x) = [\,]$ and $(\Gamma \setminus\!\setminus x)(y) = \Gamma(y)$ if $y \neq x$.

To present/discuss different typing systems, we consider the following derivability notions. A **type judgment** is a triple $\Gamma \vdash t : \sigma$, where $\Gamma$ is a variable assignment, $t$ a term and $\sigma$ a type. A **(type) derivation** in system $\mathcal{X}$ is a tree obtained by applying the (inductive) rules of the type system $\mathcal{X}$, where each node corresponds to the application of some typing rule. We write $\Phi \rhd_{\mathcal{X}} \Gamma \vdash t : \sigma$ if $\Phi$ is a type derivation concluding with the type judgment $\Gamma \vdash t : \sigma$, and just $\rhd_{\mathcal{X}} \Gamma \vdash t : \sigma$ if there exists $\Phi$ such that $\Phi \rhd_{\mathcal{X}} \Gamma \vdash t : \sigma$. A term $t$ is **$\mathcal{X}$-typable** iff there is a derivation in $\mathcal{X}$ typing $t$, *i.e.* if there is $\Phi$ such that $\Phi \rhd_{\mathcal{X}} \Gamma \vdash t : \sigma$. We may omit the index $\mathcal{X}$ if the name of the system is clear from the context.

3.1. **Characterizing Head $\beta$-Normalizing $\lambda$-Terms.** We discuss in this section typing systems being able to characterize head $\beta$-normalizing $\lambda$-terms. We first consider system $\mathcal{H}_\lambda$ in Figure 1, first appearing in [22], then in [17].

$$\frac{}{x : [\tau] \vdash x : \tau}\,(\mathtt{ax}) \qquad \frac{\Gamma \vdash t : \tau}{\Gamma \setminus\!\setminus x \vdash \lambda x.t : \Gamma(x) \Rightarrow \tau}\,(\Rightarrow_{\mathtt{i}}) \qquad \frac{\Gamma \vdash t : [\sigma_k]_{k\in K} \Rightarrow \tau \quad (\Gamma_k \vdash u : \sigma_k)_{k\in K}}{\Gamma \wedge \wedge_{k\in K}\Gamma_k \vdash t\,u : \tau}\,(\Rightarrow_{\mathtt{e}})$$

Figure 1: System $\mathcal{H}_\lambda$

Rule $(\mathtt{ax})$ Rule $(\Rightarrow_{\mathtt{i}})$ $\qquad \dfrac{(\Gamma_k \vdash t : \sigma_k)_{k\in K}}{\wedge_{k\in K}\Gamma_k \Vdash t : [\sigma_k]_{k\in K}}\,(\wedge) \qquad \dfrac{\Gamma \vdash t : \mathcal{I} \Rightarrow \sigma \quad \Gamma' \Vdash u : \mathcal{I}}{\Gamma \wedge \Gamma' \vdash t\,u : \sigma}\,(\Rightarrow_{\mathtt{e}})$

Figure 2: System $\mathcal{H}'_\lambda$

Notice that $K = \emptyset$ in rule $(\Rightarrow_{\mathtt{e}})$ allows to type an application $t\,u$ without necessarily typing the subterm $u$. Thus, if $\Omega = (\lambda x.xx)(\lambda x.xx)$, then from the judgment $x : [\sigma] \vdash x : \sigma$ we can derive for example $x : [\sigma] \vdash (\lambda y.x)\Omega : \sigma$.

System $\mathcal{H}_\lambda$ characterizes head $\beta$-normalization:

**Theorem 3.1.** *Let $t \in \mathcal{T}_\lambda$. Then $t$ is $\mathcal{H}_\lambda$-typable iff $t \in \mathcal{HN}(\lambda)$ iff the head-strategy terminates on $t$.*

Moreover, the implication *typability implies termination of the head-strategy* can be shown by simple arithmetical arguments provided by the *quantitative* flavour of the typing system $\mathcal{H}_\lambda$, in contrast to classical reducibility arguments usually invoked in other cases [24, 35]. Actually, the arithmetical arguments give the following quantitative property:

**Theorem 3.2.** *If $t$ is $\mathcal{H}_\lambda$-typable with tree derivation $\Pi$, then the size (number of nodes) of $\Pi$ gives an upper bound to the length of the head-reduction strategy starting at $t$.*

To reformulate system $\mathcal{H}_\lambda$ in a different way, we now distinguish two sorts of judgments: **regular judgments** of the form $\Gamma \vdash t : \sigma$ assigning *types* to terms, and **auxiliary judgments** of the form $\Gamma \Vdash t : \mathcal{I}$ assigning *intersection types* to terms.

An equivalent formulation of system $\mathcal{H}_\lambda$, called $\mathcal{H}'_\lambda$, is given in Figure 2, (where we always use the name $(\Rightarrow_{\mathtt{e}})$ for the rule typing the application term, even if the rule is different from that in system $\mathcal{H}_\lambda$). There are two inherited forms of type derivations: **regular** (resp. **auxiliary**) **derivations** are those that conclude with regular (resp. auxiliary) judgments. Notice that $K = \emptyset$ in rule $(\wedge)$ gives $\emptyset \Vdash u : [\,]$ for *any term $u$, e.g. $\emptyset \Vdash \Omega : [\,]$, so that one can also derive $x : [\tau] \vdash (\lambda y.x)\Omega : \tau$ in this system.* Notice also that systems $\mathcal{H}_\lambda$ and $\mathcal{H}'_\lambda$ are *relevant*, *i.e.* they lack weakening. Equivalence between $\mathcal{H}_\lambda$ and $\mathcal{H}'_\lambda$ gives the following result:

**Corollary 3.3.** *Let $t \in \mathcal{T}_\lambda$. Then $t$ is $\mathcal{H}'_\lambda$-typable iff $t \in \mathcal{HN}(\lambda)$ iff the head-strategy terminates on $t$.*

Auxiliary judgments turn out to substantially lighten the notations and to make the statements (and their proofs) more readable.

3.2. **Characterizing Strongly $\beta$-Normalizing $\lambda$-Terms.** We now discuss typing systems being able to characterize strongly $\beta$-normalizing $\lambda$-terms. We first consider system $\mathcal{S}_\lambda$ in Figure 3, which appears in [11] (slight variants appear in [16, 9, 29]). Rule $(\Rightarrow_{\mathtt{e}_1})$ forces the *erasable argument* (the subterm $u$) to be typed, even if the type of $u$ (*i.e.* $\sigma$) is not being used in the conclusion of the judgment. Thus, in contrast to system $\mathcal{H}_\lambda$, every subterm of a typed term is now typed. System $\mathcal{S}_\lambda$ characterizes strong $\beta$-normalization:

$$\frac{}{x : [\tau] \vdash x : \tau}\,(\mathtt{ax}) \qquad \frac{\Gamma \vdash t : \tau}{\Gamma \setminus\!\setminus x \vdash \lambda x.t : \Gamma(x) \Rightarrow \tau}\,(\Rightarrow_{\mathtt{i}}) \qquad \frac{\Gamma \vdash t : [\,] \Rightarrow \tau \quad \Delta \vdash u : \sigma}{\Gamma \wedge \Delta \vdash t\,u : \tau}\,(\Rightarrow_{\mathtt{e}_1})$$

$$\frac{\Gamma \vdash t : [\sigma_k]_{k\in K} \Rightarrow \tau \quad (\Delta_k \vdash u : \sigma_k)_{k\in K} \quad K \neq \emptyset}{\Gamma \wedge \wedge_{k\in K}\Delta_k \vdash t\,u : \tau}\,(\Rightarrow_{\mathtt{e}_2})$$

Figure 3: System $\mathcal{S}_\lambda$

Rule $(\mathtt{ax})$ Rule $(\Rightarrow_{\mathtt{i}})$ $\dfrac{(\Gamma_k \vdash t : \sigma_k)_{k\in K}}{\wedge_{k\in K}\Gamma_k \Vdash t : [\sigma_k]_{k\in K}}\,(\wedge)$ $\dfrac{\Gamma \vdash t : \mathcal{I} \Rightarrow \tau \quad \Delta \Vdash u : \mathcal{I}^*}{\Gamma \wedge \Delta \vdash tu : \tau}\,(\Rightarrow_{\mathtt{e}*})$

Figure 4: System $\mathcal{S}'_\lambda$

**Lemma 3.4.** *Let $t \in \mathcal{T}_\lambda$. Then $t$ is $\mathcal{S}_\lambda$-typable iff $t \in \mathcal{SN}(\lambda)$.*

As before, the implication *typability implies normalization* can be show by simple arithmetical arguments provided by the *quantitative* flavour of the typing system $\mathcal{S}_\lambda$. Actually, the *proof* of Lemma 3.4 gives the following bound:

**Theorem 3.5.** *If $t$ is $\mathcal{S}_\lambda$-typable with tree derivation $\Pi$, then the size (number of nodes) of $\Pi$ gives an upper bound to the length of any reduction path starting at $t$.*

An equivalent formulation of system $\mathcal{S}_\lambda$, called $\mathcal{S}'_\lambda$, is given in Figure 4. As before, we use regular as well as auxiliary judgments. Notice that $K = \emptyset$ in rule $(\wedge)$ is still possible, but derivations of the form $\Vdash t : [\,]$, representing untyped terms, will never be used. The choice operation $\_^*$ (defined at the beginning of Section 3) in rule $(\Rightarrow_{\mathtt{e}*})$ is used to impose arbitrary types for erasable terms, *i.e.* when $t$ has type $[\,] \Rightarrow \tau$, then $u$ needs to be typed with an arbitrary type $[\sigma]$, thus the auxiliary judgment typing $u$ on the right premise of $(\Rightarrow_{\mathtt{e}*})$ can never assign $[\,]$ to $u$. This should be understood as a form of *restricted* weakening on the argument $u$, , which is only performed *before* the application of rule $(\Rightarrow_{\mathtt{e}*})$ and only when the functional type is of the form $[\,] \to \mathcal{U}$.

**Example 1.** Here is an example of type derivation in system $\mathcal{S}'_\lambda$.

$$\dfrac{\dfrac{\dfrac{}{x : [\sigma] \vdash x : \sigma}\,(\mathtt{ax})}{x : [\sigma] \vdash \lambda y.x : [\,] \Rightarrow \sigma}\,(\Rightarrow_{\mathtt{i}}) \qquad \dfrac{\dfrac{}{z : [\tau] \vdash z : \tau}\,(\mathtt{ax})}{z : [\tau] \Vdash z : [\tau]}\,(\wedge)}{x : [\sigma], z : [\tau] \vdash (\lambda y.x)z : \sigma}\,(\Rightarrow_{\mathtt{e}*})$$

Since $\mathcal{S}_\lambda$ and $\mathcal{S}'_\lambda$ are equivalent, we also have:

**Corollary 3.6.** *Let $t \in \mathcal{T}_\lambda$. Then $t$ is $\mathcal{S}'_\lambda$-typable iff $t \in \mathcal{SN}(\lambda)$.*

## 4. Quantitative Type Systems for the $\lambda\mu$-Calculus

We present in this section two quantitative systems for the $\lambda\mu$-calculus, systems $\mathcal{H}_{\lambda\mu}$ (Section 4.2) and $\mathcal{S}_{\lambda\mu}$ (Section 4.3), characterizing, respectively, head and strong $\lambda\mu$-normalizing objects. Since $\lambda$-calculus is embedded in the $\lambda\mu$-calculus, then the starting points to design $\mathcal{H}_{\lambda\mu}$ and $\mathcal{S}_{\lambda\mu}$ are, respectively, systems $\mathcal{H}'_\lambda$ and $\mathcal{S}'_\lambda$, introduced in Section 3.

4.1. **Types.** We consider a countable set of **base types** $a, b, c \ldots$ and the following categories of types:

| | | | |
|---|---|---|---|
| (**Object Types**) | $\mathcal{A}$ | := | $\mathcal{C} \mid \mathcal{U}$ |
| (**Command Type**) | $\mathcal{C}$ | := | $\#$ |
| (**Union Types**) | $\mathcal{U}, \mathcal{V}$ | ::= | $\langle \sigma_k \rangle_{k \in K}$ |
| (**Intersection Types**) | $\mathcal{I}, \mathcal{J}$ | ::= | $[\mathcal{U}_k]_{k \in K}$ |
| (**Types**) | $\sigma, \tau$ | ::= | $a \mid \mathcal{I} \Rightarrow \mathcal{U}$ |
| (**Blind Types**) | $\xi$ | ::= | $a \mid [\,] \Rightarrow \langle \xi \rangle$ |

The constant $\#$ is used to type commands, union types to type terms, and intersection types to type variables (thus left-hand sides of arrows). We assume $K$ to be any finite set. Both $[\sigma_k]_{k \in \{1..n\}}$ and $\langle \sigma_k \rangle_{k \in \{1..n\}}$ can be seen as *multisets*, representing, respectively, $\sigma_1 \cap \ldots \cap \sigma_n$ and $\sigma_1 \cup \ldots \cup \sigma_n$, where $\cap$ and $\cup$ are both associative, commutative, but *non-idempotent*. We may omit the indices in the simplest case: thus $[\mathcal{U}]$ and $\langle \sigma \rangle$ denote singleton multisets. We define the operator $\wedge$ (resp. $\vee$) on intersection (resp. union) multiset types by : $[\mathcal{U}_k]_{k \in K} \wedge [\mathcal{V}_\ell]_{\ell \in L} := [\mathcal{U}_k]_{k \in K} + [\mathcal{V}_\ell]_{\ell \in L}$ and $\langle \sigma_k \rangle_{k \in K} \vee \langle \tau_\ell \rangle_{\ell \in L} := \langle \sigma_k \rangle_{k \in K} + \langle \tau_\ell \rangle_{\ell \in L}$, where $+$ always means multiset union. The **non-deterministic choice** operation (resulting from a non-deterministic choice relation, as in Section 3) is now not only defined on intersection but also on union types:

$$[\mathcal{U}_k]^*_{k \in K} := \begin{cases} [\mathcal{U}] & \text{if } K = \emptyset \text{ and } \mathcal{U} \neq \langle\,\rangle \text{ is any arbitrary non-empty union type} \\ [\mathcal{U}_k]_{k \in K} & \text{if } K \neq \emptyset \end{cases}$$

$$\langle \sigma_k \rangle^*_{k \in K} := \begin{cases} \langle \xi \rangle & \text{if } K = \emptyset \text{ and } \xi \text{ is any arbitrary blind type} \\ \langle \sigma_k \rangle_{k \in K} & \text{if } K \neq \emptyset \end{cases}$$

The choice operation for union type is defined so that (1) the empty union cannot be assigned to $\mu$-abstractions (2) subject reduction is guaranteed in system $\mathcal{H}_{\lambda\mu}$ for erasing steps $(\mu\alpha.c)u \to_\mu \mu\alpha.c$, where $\alpha \notin \mathtt{fn}(c)$. We present concrete examples in Section **??** which illustrates the need of non-empty union types and blind types to guarantee subject reduction.

The **arity** of types and union multiset types is defined by induction: for types $\sigma$, if $\sigma = \mathcal{I} \Rightarrow \mathcal{U}$, then $\mathtt{ar}(\sigma) := \mathtt{ar}(\mathcal{U}) + 1$, otherwise, $\mathtt{ar}(\sigma) := 0$; for union multiset types, $\mathtt{ar}(\langle \sigma_k \rangle_{k \in K}) := \Sigma_{k \in K}\, \mathtt{ar}(\sigma_k)$. Thus, the arity of a type counts the number of its *top-level* arrows. The **cardinality of multisets** is defined by $|[\mathcal{U}_k]_{k \in K}| = |\langle \sigma_k \rangle_{k \in K}| := |K|$.

**Variable assignments** (written $\Gamma$), are, as before, total functions from variables to intersection multiset types. Similarly, **name assignments** (written $\Delta$), are total functions from names to union multiset types. The **domain of** $\Delta$ is given by $\mathtt{dom}(\Delta) := \{\alpha \mid \Delta(\alpha) \neq \langle\,\rangle\}$, where $\langle\,\rangle$ is the empty union multiset, so that when $\alpha \notin \mathtt{dom}(\Delta)$, $\Delta(\alpha)$ stands for $\langle\,\rangle$. We may write $\emptyset$ to denote the name assignment that associates the empty union type $\langle\,\rangle$ to every name. We write $\Delta \vee \Delta'$ for $\alpha \mapsto \Delta(\alpha) \vee \Delta'(\alpha)$, so that $\mathtt{dom}(\Delta \vee \Delta') = \mathtt{dom}(\Delta) \cup \mathtt{dom}(\Delta')$.

When $\mathtt{dom}(\Delta) \cap \mathtt{dom}(\Delta') = \emptyset$, we may write $\Delta; \Delta'$ for $\Delta \vee \Delta'$. We write $\alpha : \mathcal{U}$ (even when $\alpha = \langle\,\rangle$) for the name assignment mapping $\alpha$ to $\mathcal{U}$ and all $\beta \neq \alpha$ to $\langle\,\rangle$. We write $\Delta \setminus\!\setminus \alpha$ for the name assignment defined by $(\Delta \setminus\!\setminus \alpha)(\alpha) = \langle\,\rangle$ and $(\Delta \setminus\!\setminus \beta)(\alpha) = \Delta(\beta)$ if $\beta \neq \alpha$.

We now present our typing systems $\mathcal{H}_{\lambda\mu}$ and $\mathcal{S}_{\lambda\mu}$, both having **regular** (resp. **auxiliary**) judgments of the form $\Gamma \vdash t : \mathcal{U} \mid \Delta$ (resp. $\Gamma \Vdash t : \mathcal{I} \mid \Delta$), together with their respective notions of regular and auxiliary derivations. An important syntactical property they enjoy is that both are **syntax directed**, *i.e.* for each (regular/auxiliary) typing judgment $j$ there is a *unique* typing rule whose conclusion matches the judgment $j$. This makes our proofs much simpler than those arising with idempotent types, which are based on long generation lemmas (*e.g.* [9, 47]).

4.2. **System $\mathcal{H}_{\lambda\mu}$.** In this section we present a quantitative typing system for $\lambda\mu$, called $\mathcal{H}_{\lambda\mu}$, characterizing head $\lambda\mu$-normalization. It can be seen as a first intuitive step to understand the typing system $\mathcal{S}_{\lambda\mu}$, introduced later in Section 4.3, and characterizing strong $\lambda\mu$-normalization. However, to avoid redundancy, the properties of the two systems are not described in the same way:

- For $\mathcal{H}_{\lambda\mu}$, we provide informal discussions to explain the main requirements needed to capture quantitative information in the presence of classical feature (names, $\mu$-redexes). We particularly focus on the necessity of banning empty union types. We do not give the proofs of the properties of $\mathcal{H}_{\lambda\mu}$, because they are simpler than those of system $\mathcal{S}_{\lambda\mu}$.
- For $\mathcal{S}_{\lambda\mu}$, we provide a more compact presentation, since the main technical key choices used for $\mathcal{H}_{\lambda\mu}$ are still valid. However, full statements and proofs of the properties of $\mathcal{S}_{\lambda\mu}$ are detailed.

The (syntax directed) rules of the typing system $\mathcal{H}_{\lambda\mu}$ are presented in Figure 5. Rule

$$\frac{\mathcal{U} \neq \langle\,\rangle}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\,(\mathtt{ax}) \qquad \frac{\Gamma \vdash t : \mathcal{U} \mid \Delta}{\Gamma \setminus\!\setminus x \vdash \lambda x.t : \langle \Gamma(x) \Rightarrow \mathcal{U} \rangle \mid \Delta}\,(\Rightarrow_{\mathtt{i}}) \qquad \frac{\Gamma \vdash t : \mathcal{U} \mid \Delta}{\Gamma \vdash [\alpha]t : \# \mid \Delta \vee \{\alpha : \mathcal{U}\}}\,(\#_{\mathtt{i}})$$

$$\frac{\Gamma \vdash c : \# \mid \Delta}{\Gamma \vdash \mu\alpha.c : \Delta(\alpha)^* \mid \Delta \setminus\!\setminus \alpha}\,(\#_{\mathtt{e}}) \qquad \frac{(\Gamma_k \vdash t : \mathcal{U}_k \mid \Delta_k)_{k \in K}}{\wedge_{k\in K}\Gamma_k \Vdash t : [\mathcal{U}_k]_{k\in K} \mid \vee_{k\in K}\Delta_k}\,(\wedge)$$

$$\frac{\Gamma_t \vdash t : \langle \mathcal{I}_k \Rightarrow \mathcal{U}_k \rangle_{k\in K} \mid \Delta_t \quad \Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k \mid \Delta_u}{\Gamma_t \wedge \Gamma_u \vdash tu : \vee_{k\in K}\mathcal{U}_k \mid \Delta_t \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}})$$

Figure 5: System $\mathcal{H}_{\lambda\mu}$

$(\Rightarrow_{\mathtt{e}})$ is to be understood as a *logical admissible* rule: if union (resp. intersection) is interpreted as the $\mathtt{OR}$ (resp. $\mathtt{AND}$) logical connective, then $\mathtt{OR}_{k\in K}\ (\mathcal{I}_k \Rightarrow \mathcal{U}_k)$ and $(\mathtt{AND}_{k\in K}\ \mathcal{I}_k)$ implies $(\mathtt{OR}_{k\in K}\ \mathcal{U}_k)$. As in the simply typed $\lambda\mu$-calculus [42], the $(\#_{\mathtt{i}})$ rule saves a type $\mathcal{U}$ for the name $\alpha$, however, in our system, the corresponding name assignment $\Delta \vee \{\alpha : \mathcal{U}\}$, specified by means of $\vee$, collects *all* the types that $\alpha$ has been assigned during the derivation. Notice that the $(\#_{\mathtt{e}})$-rule is not deterministic since $\Delta(\alpha)^*$ denotes an arbitrary union type when $\Delta(\alpha)$ is $\langle\,\rangle$, a technical requirement which is discussed at the end of the section.

In the simply typed $\lambda\mu$-calculus, **call-cc** $= \lambda y.\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x)$ would be typed with $((a \Rightarrow b) \Rightarrow a) \Rightarrow a$ (Peirce's Law), so that the fact that $\alpha$ is *used* twice in the type derivation would not be explicitly materialized with simple types (same comment applies to *idempotent* intersection/union types). This makes a strong contrast with the derivation in Figure 6, where $\mathcal{A} := \langle a \rangle$, $\mathcal{B} := \langle b \rangle$, $\mathcal{U}_y := \langle [\langle [\mathcal{A}] \Rightarrow \mathcal{B} \rangle] \Rightarrow \mathcal{A} \rangle$, and **call-cc** is typed with the union type $\langle [\langle [\langle [\mathcal{A}] \Rightarrow \mathcal{B} \rangle] \Rightarrow \mathcal{A} \rangle] \Rightarrow (\mathcal{A} \vee \mathcal{A}) \rangle$.

$$\dfrac{\dfrac{}{y : [\mathcal{U}_y] \vdash y : \mathcal{U}_y \mid \emptyset}\,(\mathtt{ax}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : [\mathcal{A}] \vdash x : \mathcal{A} \mid \emptyset}\,(\mathtt{ax})}{x : [\mathcal{A}] \vdash [\alpha]x : \# \mid \alpha : \mathcal{A}}\,(\#_{\mathtt{i}})}{x : [\mathcal{A}] \vdash \mu\beta.[\alpha]x : \mathcal{B} \mid \alpha : \mathcal{A}}\,(\#_{\mathtt{e}})}{\vdash \lambda x.\mu\beta.[\alpha]x : \langle [\mathcal{A}] \Rightarrow \mathcal{B} \rangle \mid \alpha : \mathcal{A}}\,(\Rightarrow_{\mathtt{i}})}{\Vdash \lambda x.\mu\beta.[\alpha]x : [\langle [\mathcal{A}] \Rightarrow \mathcal{B} \rangle] \mid \alpha : \mathcal{A}}\,(\wedge)}{\dfrac{y : [\mathcal{U}_y] \vdash y(\lambda x.\mu\beta.[\alpha]x) : \mathcal{A} \mid \alpha : \mathcal{A}}{\dfrac{y : [\mathcal{U}_y] \vdash [\alpha]y(\lambda x.\mu\beta.[\alpha]x) : \# \mid \alpha : \mathcal{A} \vee \mathcal{A}}{y : [\mathcal{U}_y] \vdash \mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x) : \mathcal{A} \vee \mathcal{A} \mid \emptyset}\,(\#_{\mathtt{e}})}\,(\#_{\mathtt{i}})}\,(\Rightarrow_{\mathtt{e}})$$

$$\vdash \lambda y.\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x) : \langle [\langle [\langle [\mathcal{A}] \Rightarrow \mathcal{B} \rangle] \Rightarrow \mathcal{A} \rangle] \Rightarrow (\mathcal{A} \vee \mathcal{A}) \rangle \mid \emptyset\,(\Rightarrow_{\mathtt{i}})$$

Figure 6: Typing **call-cc**

This example suggests to distinguish two different uses of names:

- The name $\alpha$ is saved twice by a $(\#_{\mathtt{i}})$ rule : once for $x$ and once for $y(\lambda x.\mu\beta.[\alpha]x)$, both times with type $\mathcal{A}$. After that, the abstraction $\mu\alpha.[\alpha]y(\lambda x.\mu\beta.[\alpha]x)$ **restores** the union of the two types that were previously stored by $\alpha$ (by means of the two $(\#_{\mathtt{i}})$-rules). A similar phenomenon occurs with rule $(\Rightarrow_{\mathtt{i}})$, which restores the types of the abstracted variables.
- The name $\beta$ is not free in $[\alpha]x$, so that a new union type $\mathcal{B}$ is introduced to type the abstraction $\mu\beta.[\alpha]x$. From a logical point of view, this corresponds to a *weakening* on the right hand-side of the sequent, which is necessary, for instance, to derive the non-idempotent counterpart of Peirce's Law (*cf.* Fig. 6). Consequently, $\lambda$ and $\mu$-abstractions are not treated symmetrically: when $x$ is not free in $t$, then $\lambda x.t$ will be typed with $\langle [\,] \Rightarrow \sigma \rangle$ (where $\sigma$ is the type of $t$), and no new arbitrary intersection type is introduced for the abstracted variable $x$.

An interesting observation is about the restriction of system $\mathcal{H}_{\lambda\mu}$ to the pure $\lambda$-calculus: union types, name assignments and rules $(\#_{\mathtt{e}})$ and $(\#_{\mathtt{i}})$ are ruled out, so that every union multiset takes the single form $\langle \tau \rangle$, which can be simply identified with $\tau$. Thus, the restricted typing system $\mathcal{H}_{\lambda\mu}$ becomes system $\mathcal{H}'_{\lambda}$ in Figure 2.

Another observation is about the property of *relevance* of assignments. Although there is a *restricted* form of weakening in system $\mathcal{H}_{\lambda\mu}$ (allowing for example to derive the non-idempotent counterpart of Peirce's Law, see the top $(\#_{\mathtt{e}})$-rule in Fig. 6), if $\Gamma \vdash o : \mathcal{A} \mid \Delta$ is derivable, then any $x \in \mathtt{dom}(\Gamma)$ (resp. $\alpha \in \mathtt{dom}(\Delta)$) has at least one free occurrence in $o$. Formally,

**Lemma 4.1 (Relevance for System $\mathcal{H}_{\lambda\mu}$).** *Let $o \in \mathcal{O}_{\lambda\mu}$. If $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$, then $\mathtt{dom}(\Gamma) \subseteq \mathtt{fv}(o)$ and $\mathtt{dom}(\Delta) \subseteq \mathtt{fn}(o)$.*

*Proof.* By induction on $\Phi$. □

We define now our notion of **derivation size**, which is a natural number representing the amount of information in tree derivations. For any type derivation $\Phi$, $\mathtt{sz}(\Phi)$ is inductively defined by the following rules, where we use an abbreviated, self-understanding notation for the premises.

$$\mathtt{sz}\left(\frac{}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\,(\mathtt{ax})\right) := 1$$

$$\mathtt{sz}\left(\frac{\Phi_t \triangleright t}{\Gamma \setminus\!\setminus x \vdash \lambda x.t : \langle \Gamma(x) \Rightarrow \mathcal{U}\rangle \mid \Delta}\,(\Rightarrow_{\mathtt{i}})\right) := \mathtt{sz}(\Phi_t) + 1$$

$$\mathtt{sz}\left(\frac{\Phi_t \triangleright t}{\Gamma \vdash [\alpha]t : \# \mid \Delta \vee \{\alpha : \mathcal{U}\}}\,(\#_{\mathtt{i}})\right) := \mathtt{sz}(\Phi_t) + \mathtt{ar}(\mathcal{U})$$

$$\mathtt{sz}\left(\frac{\Phi_c \triangleright c}{\Gamma \vdash \mu\alpha.c : \Delta(\alpha)^* \mid \Delta \setminus\!\setminus \alpha}\,(\#_{\mathtt{e}})\right) := \mathtt{sz}(\Phi_c) + 1$$

$$\mathtt{sz}\left(\frac{(\Phi_k \triangleright t)_{k\in K}}{\wedge_{k\in K}\Gamma_k \Vdash t : [\mathcal{U}_k]_{k\in K} \mid \vee_{k\in K}\Delta_k}\,(\wedge)\right) := \Sigma_{k\in K}\ \mathtt{sz}(\Phi_k)$$

$$\mathtt{sz}\left(\frac{\Phi_t \triangleright t : \langle \mathcal{I}_k \to \mathcal{U}_k\rangle_{k\in K} \quad \Phi_u \triangleright u}{\Gamma \vdash tu : \vee_{k\in K}\mathcal{V}_k \mid \Delta}\,(\Rightarrow_{\mathtt{e}})\right) := \mathtt{sz}(\Phi_t) + \mathtt{sz}(\Phi_u) + |K|$$

System $\mathcal{H}_{\lambda\mu}$ behaves as expected, in particular, typing is stable by reduction (Subject Reduction) and anti-reduction (Subject Expansion):

**Property 4.2 (Weighted Subject Reduction for $\mathcal{H}_{\lambda\mu}$).** Let $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$. If $o \to o'$, then there exists a derivation $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ such that $\mathtt{sz}(\Phi') \leq \mathtt{sz}(\Phi)$. Moreover, if the reduced redex is typed, then $\mathtt{sz}(\Phi') < \mathtt{sz}(\Phi)$.

An important remark is that, if the arity of the types were not taken into account in the size of the rules $(\#_{\mathtt{i}})$, then we would only have $\mathtt{sz}(\Phi') = \mathtt{sz}(\Phi)$ (and not $\mathtt{sz}(\Phi') < \mathtt{sz}(\Phi)$) for the $\mu$-reduction steps. Intuitively, the $\mu$-reduction $(\mu\alpha.c)u \to_\mu \mu\alpha.c\{\alpha /\!\!/ u\}$ dispatches the $(\Rightarrow_{\mathtt{e}})$-rule typing the root of the $\mu$-redex $(\mu\alpha.c)u$ into several created $(\Rightarrow_{\mathtt{e}})$-rules in the reduct, but neither an increase nor a decrease of the measure is ensured. The solution to recover this key feature (*i.e.* the decrease) is suggested by the effect of $\mu$-reduction on the $(\#_{\mathtt{i}})$-rules associated to $\alpha$ (see Figure 7): indeed, $\mu$-reduction replaces every named term $[\alpha]v$ by $[\alpha]vu$, where $u$ is the argument of the $\mu$-redex, so that the saved types are smaller under the created $(\Rightarrow_{\mathtt{e}})$-rules than the ones in the original derivation.

As expected from an intersection (and union) type system, head normal forms are typable in system $\mathcal{H}_{\lambda\mu}$. Moreover, subject expansion holds for $\mathcal{H}_{\lambda\mu}$, meaning that typing is stable under *anti*-reduction. Note that we do not state a *weighted* subject expansion property (although this would be possible) only because this is not necessary to prove the final characterization property of system $\mathcal{H}_{\lambda\mu}$ (*cf.* Theorem 4.4).

$$\frac{\ldots \vdash v : \langle \mathcal{I}_k \to \mathcal{U}_k \rangle_{k\in K} \mid \ldots}{\ldots \vdash [\alpha]v : \# \mid \ldots \vee \{\alpha : \langle \mathcal{I}_k \to \mathcal{U}_k \rangle_{k\in K}\}} (\#_{\mathtt{i}})$$

becomes

$$\frac{\dfrac{\ldots \vdash v : \langle \mathcal{I}_k \to \mathcal{U}_k \rangle_{k\in K} \mid \ldots \qquad \ldots \Vdash u : \wedge_{k\in K}\mathcal{I}_k \mid \ldots}{\ldots \vdash v\,u : \vee_{k\in K}\mathcal{U}_k \mid \ldots}(\Rightarrow_{\mathtt{e}})}{\ldots \vdash [\alpha]v\,u : \# \mid \ldots \vee \{\alpha : \vee_{k\in K}\mathcal{U}_k\}}(\#_{\mathtt{i}})$$

Figure 7: Effect of $\mu$-reduction on rule $(\#_{\mathtt{i}})$

**Property 4.3 (Subject Expansion for $\mathcal{H}_{\lambda\mu}$).** Let $\Phi' \rhd \Gamma' \vdash o' : \mathcal{A} \mid \Delta'$. If $o \to o'$, then there is $\Phi \rhd \Gamma' \vdash o : \mathcal{A} \mid \Delta'$.

Note in particular that the head strategy only reduces typed redexes (the head redex of a *head reducible* typed term is necessarily typed), so that finally, we can state our type-theoretic characterization of head normalization for the $\lambda\mu$-calculus:

**Theorem 4.4.** *Let $o \in \mathcal{O}_{\lambda\mu}$. Then $o$ is $\mathcal{H}_{\lambda\mu}$-typable iff $o \in \mathcal{HN}(\lambda\mu)$ iff the head-strategy terminates on $o$. Moreover, if $o$ is $\mathcal{H}_{\lambda\mu}$-typable with tree derivation $\Pi$, then $\mathtt{sz}(\Pi)$ gives an upper bound to the length of the head-reduction strategy starting at $o$.*

We do not provide the proofs of these properties and the last theorem, because it uses special cases of the more general technology that we are going to develop later to deal with strong normalization. Notice however that Theorem 4.4 ensures that the head-strategy is complete for head-normalization in $\lambda\mu$.

4.3. **Discussion.** Now that we have stated the main result of this section, based on the key (weighted) subject reduction property, let us come back to the design choices of our development, in particular:

- The choice $\_^*$ operator in rule $(\#_{\mathtt{e}})$ requires the types to be blind, and
- The union types are non-empty.

Blind types raised in rule $(\#_{\mathtt{e}})$ guarantee that terms do not have empty union types, the necessity of which is a delicate constraint discussed later. In the meantime, we start by justifying the particular form they adopt (arrows with empty domains), which is easier to justify. Thus, consider the following example: let $t_1 = \mu\beta.[\gamma]x$ with $\beta \neq \gamma$ and $x \neq y$, so that $t_1\, y \to_\mu t_1$. A typing derivation of $t_1$ necessarily concludes with $x : [\mathcal{V}_x] \vdash t_1 : \langle\,\rangle^* \mid \gamma : \mathcal{V}_x$, for some union type $\mathcal{V}_x$. Let us assume, only temporarily, that a non-blind type can be chosen by the non-deterministic operator in rule $(\#_{\mathtt{e}})$ *e.g.* $\langle\,\rangle^* = \langle [\mathcal{U}] \Rightarrow \mathcal{U}\rangle$ for some union type $\mathcal{U}$. If we then assign $\mathcal{U}$ to $y$, the judgment $y : [\mathcal{U}]; x : [\mathcal{V}_x] \vdash t_1\, y : \mathcal{U} \mid \gamma : \mathcal{V}_x$ is derivable by rule $(\Rightarrow_{\mathtt{e}})$. However, by relevance (Lemma 4.1), the judgment $y : [\mathcal{U}]; x : [\mathcal{V}_x] \vdash t_1 : \mathcal{U} \mid \gamma : \mathcal{V}_x$ cannot be derivable since $y \notin \mathtt{fv}(t_1)$. Thus, subject reduction simply fails. Note that if $\langle\,\rangle^*$ chooses a blind type, for example $\langle [\,] \Rightarrow \langle a\rangle\rangle$, then $y$ is untyped in the derivation of $t_1 y$, *i.e.* $x : [\mathcal{V}_x] \vdash t_1\, y : \mathcal{U} \mid \gamma : \mathcal{V}_x$, so that subject reduction holds.

We now explain why union types are non-empty. In particular, there are two different typing rules requiring union types to be non-empty: rule $(\mathtt{ax})$ and rule $(\#_{\mathtt{e}})$. To illustrate the necessity of non-empty union types for $\mu$-abstractions, *i.e.* rule $(\#_{\mathtt{e}})$, let us assume, again

temporarily, that an empty union type is introduced by the rule $(\#_{\mathtt{e}})$ when $\alpha \notin \mathtt{dom}(\Delta)$, and call such an instance $(\#_{\mathtt{empty}})$ when this happens. Similarly, let us also call the instance of an application rule to be $(\Rightarrow_{\mathtt{empty}})$ when the union type of its left-hand side is empty.

$$\frac{\Gamma \vdash c : \# \mid \Delta \qquad \alpha \notin \mathtt{dom}(\Delta)}{\Gamma \vdash \mu\alpha.c : \langle\,\rangle \mid \Delta}\,(\#_{\mathtt{empty}}) \qquad \frac{\Gamma \vdash t : \langle\,\rangle \mid \Delta \quad \emptyset \Vdash u : [\,] \mid \emptyset}{\Gamma \vdash tu : \langle\,\rangle \mid \Delta}\,(\Rightarrow_{\mathtt{empty}})$$

For instance, given $t_1 := \mu\beta.[\gamma]x$, where $\beta \neq \gamma$, every derivation typing $t_1$ concludes with rule $(\#_{\mathtt{empty}})$ typing a judgment of the form $x : [\mathcal{V}_x] \vdash t_1 : \langle\,\rangle \mid \gamma : \mathcal{V}_x$.

Assume now that a derivation $\Pi_c$ typing a command $c$ contains $k$ empty rules $(\#_{\mathtt{empty}})$ w.r.t. the name $\alpha$, and that $\Pi_c$ is a subderivation of some other derivation typing a $\mu$-redex $(\mu\alpha.c)u$ for some term $u$. Let us give an example with $k = 2$ by setting $c := [\alpha]t_2$, where $t_2 := \mu\beta'.[\alpha]t_1$ and $t_1 := \mu\beta.[\gamma]x$ as before. Then, the (empty union) types of the terms $t_1$ and $t_2$ are saved by $\alpha$, simply because $\beta$ (resp. $\beta'$) does not occur free in the body of $t_1$ (resp. $t_2$). Thus, while typing command $c$, the name $\alpha$ is necessarily typed with $\langle\,\rangle \vee \langle\,\rangle = \langle\,\rangle$.

Formally, let $\mathcal{V}_x$ be any (non-empty) union type. Then,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{x : [\mathcal{V}_x] \vdash [\gamma]x : \# \mid \gamma : \mathcal{V}_x}\,(\#_{\mathtt{i}})}{x : [\mathcal{V}_x] \vdash t_1 : \langle\,\rangle \mid \gamma : \mathcal{V}_x}\,(\#_{\mathtt{empty}} \text{ for } \beta)}{x : [\mathcal{V}_x] \vdash [\alpha]t_1 : \# \mid \gamma : \mathcal{V}_x}\,(\#_{\mathtt{i}})}{x : [\mathcal{V}_x] \vdash \mu\beta'.[\alpha]t_1 : \langle\,\rangle \mid \gamma : \mathcal{V}_x}\,(\#_{\mathtt{empty}} \text{ for } \beta')}{x : [\mathcal{V}_x] \vdash [\alpha](\mu\beta'.[\alpha]t_1) : \# \mid \gamma : \mathcal{V}_x}\,(\#_{\mathtt{i}})$$

Let us now use $c = [\alpha]t_2$ inside the $\mu$-reduction: $t = (\mu\alpha.[\alpha]t_2)u \to_\mu \mu\alpha.[\alpha](t_3u) = t'$, where $t_3 := \mu\beta'.[\alpha](t_1u)$. Then $u$ is necessarily typed as follows:

$$\frac{}{\emptyset \vdash u : [\,] \mid \emptyset}\,(\wedge)$$

Let us call $\Pi$ (resp. $\Pi'$) the type derivation of $t$ (resp $t'$). Note that the $\mu$-reduction transforms each rule $(\#_{\mathtt{i}})$ of $\Pi$ into a rule $(\Rightarrow_{\mathtt{empty}})$ followed by $(\#_{\mathtt{i}})$ in $\Pi'$ (see Figure 8). Thus, if $\Pi_c$ contains $k$ rules $(\#_{\mathtt{i}})$ introducing $\alpha$, then the derivation obtained by subject reduction typing $c\{\alpha /\!\!/ u\}$ contains $k$ *new* rules $(\Rightarrow_{\mathtt{empty}})$, each followed by a rule $(\#_{\mathtt{i}})$. Indeed, one may check in the example above that the derivation $\Pi'$ contains *two* rules $(\Rightarrow_{\mathtt{empty}})$ and *two* rules $(\#_{\mathtt{i}})$ introducing $\alpha$, whereas $\Pi$ contains *two* rules $(\#_{\mathtt{i}})$ introducing $\alpha$, and just one rule $(\Rightarrow_{\mathtt{empty}})$ (the one typing the root of the redex).

In general, one can check that whatever the size definition is for rules $(\Rightarrow_{\mathtt{e}})$, $(\#_{\mathtt{empty}})$ and $(\Rightarrow_{\mathtt{empty}})$, the derivation size cannot decrease for any choice of $k \geqslant 0$: indeed, if rule $(\#_{\mathtt{empty}})$ is used, the (possibly empty) type of a term $\mu\alpha.c$ holds no information capturing the number of free occurrences of $\alpha$ in the command $c$, so that there is no local way to know how many times the argument $u$ should be typed in the whole derivation of the term $(\mu\alpha.c)u$ (compare Figure 8 to Figure 7).

The reader will notice that the same kind of phenomenon occurs, when in the previous example, the term $t_1$ is replaced with a variable $x$ of type $\langle\,\rangle$: then $t = (\mu\alpha.[\alpha](\mu\beta'.[\alpha]x))u \to_\mu \mu\alpha.[\alpha]((\mu\beta'.[\alpha](xu))u) = t'$, the type derivation of $t'$ contains *two* rules $(\Rightarrow_{\mathtt{empty}})$ whereas the type derivation of $t$ contains just one rule $(\Rightarrow_{\mathtt{empty}})$. This explains why one cannot assign the empty union type in the rule $(\mathtt{ax})$.

$$\frac{\ldots \vdash v : \langle\,\rangle \mid \ldots}{\ldots \vdash [\alpha]v : \# \mid \ldots}\,(\#_{\mathtt{i}}) \qquad \text{becomes} \qquad \frac{\dfrac{\ldots \vdash v : \langle\,\rangle \mid \ldots \quad \ldots \Vdash u : [\,] \mid \ldots}{\ldots \vdash vu : \langle\,\rangle \mid \ldots}\,(\Rightarrow_{\mathtt{e}})}{\ldots \vdash [\alpha]v\,u : \# \mid \ldots}\,(\#_{\mathtt{i}})$$

Figure 8: Effect of $\mu$-reduction on empty types

All these arguments to forbid the empty union type in our system are not only valid for system $\mathcal{H}_{\lambda\mu}$, but also apply to the system $\mathcal{S}_{\lambda\mu}$, introduced later in Section. 4.3.

4.4. **System $\mathcal{S}_{\lambda\mu}$.** This section presents a quantitative typing system characterizing strongly normalizing $\lambda\mu$-terms. The (syntax directed) typing rules of the system $\mathcal{S}_{\lambda\mu}$ appear in Figure 9.

$$\frac{\mathcal{U} \neq \langle\,\rangle}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\,(\mathtt{ax}) \qquad \frac{\Gamma \vdash t : \mathcal{U} \mid \Delta}{\Gamma \setminus\!\setminus x \vdash \lambda x.t : \langle \Gamma(x) \Rightarrow \mathcal{U}\rangle \mid \Delta}\,(\Rightarrow_{\mathtt{i}}) \qquad \frac{\Gamma \vdash t : \mathcal{U} \mid \Delta}{\Gamma \vdash [\alpha]t : \# \mid \Delta \vee \{\alpha : \mathcal{U}\}}\,(\#_{\mathtt{i}})$$

$$\frac{\Gamma \vdash c : \# \mid \Delta}{\Gamma \vdash \mu\alpha.c : \Delta(\alpha)^* \mid \Delta \setminus\!\setminus \alpha}\,(\#_{\mathtt{e}}) \qquad \frac{(\Gamma_k \vdash t : \mathcal{U}_k \mid \Delta_k)_{k\in K}}{\wedge_{k\in K}\Gamma_k \Vdash t : [\mathcal{U}_k]_{k\in K} \mid \vee_{k\in K}\Delta_k}\,(\wedge)$$

$$\frac{\Gamma_t \vdash t : \langle \mathcal{I}_k \Rightarrow \mathcal{U}_k\rangle_{k\in K} \mid \Delta_t \qquad \Gamma_u \Vdash u : \wedge_{k\in K}\ \mathcal{I}_k^* \mid \Delta_u}{\Gamma_t \wedge \Gamma_u \vdash tu : \vee_{k\in K}\mathcal{U}_k \mid \Delta_t \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}*})$$

Figure 9: System $\mathcal{S}_{\lambda\mu}$

As in system $\mathcal{S}'_\lambda$, the non-deterministic choice operator $\_^*$ is used to choose arbitrary types for erasable terms, so that no subterm is untyped, thus ensuring strong $\lambda\mu$-normalization. The use of $\_^*$ in the $(\#_{\mathtt{e}})$-rule (raising a non-empty *union* type) can be seen as a *weakening* on the name $\alpha$ (on the right hand-side of the sequent) followed by a $\mu$-abstraction, while its use in rule $(\Rightarrow_{\mathtt{e}*})$ (raising a non-empty *intersection* type) should be seen *weakening* on the left-hand side (domain) of an arrow type. We still consider the definition of size given before, as the choice operator does not play any particular new role here.

As in system $\mathcal{H}_{\lambda\mu}$, every term is typed with a non-empty union type:

**Lemma 4.5.** *If* $\triangleright\Gamma \vdash t : \mathcal{U} \mid \Delta$, *then* $\mathcal{U} \neq \langle\,\rangle$.

As well as in the case of $\mathcal{H}_{\lambda\mu}$, system $\mathcal{S}_{\lambda\mu}$ can be restricted to the pure $\lambda$-calculus. By the same observations made at the end of Section 4.2, we get the typing system $\mathcal{S}'_\lambda$ in Figure 4 that characterizes strong $\beta$-normalization.

Relevance in system $\mathcal{S}_{\lambda\mu}$ is stated as follows:

**Lemma 4.6 (Relevance for System $\mathcal{S}_{\lambda\mu}$).** *Let* $o \in \mathcal{O}_{\lambda\mu}$. *If* $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$, *then* $\mathtt{dom}(\Gamma) = \mathtt{fv}(o)$ *and* $\mathtt{dom}(\Delta) = \mathtt{fn}(o)$.

*Proof.* By induction on $\Phi$. □

Note the difference between Lemma 4.1 (inclusion) and Lemma 4.6 (equality): this is because in $\mathcal{S}_{\lambda\mu}$ we use a choice operator in the $(\Rightarrow_{\mathtt{e}*})$-rule to prevent any subterm of a typed term to be left untyped.

The definition of $\mathtt{sz}\,()$ is extended to system $\mathcal{S}_{\lambda\mu}$ as expected. Hence, $\mathtt{sz}\,(\Phi) \geq 1$ holds for any *regular* derivation $\Phi$, whereas, by definition, the derivation of the empty auxiliary judgment $\emptyset \Vdash t : [\,] \mid \emptyset$ has size 0.

## 5. Typing Properties

This section shows two fundamental properties of reduction (*i.e. forward*) and anti-reduction (*i.e. backward*) of system $\mathcal{S}_{\lambda\mu}$. In Section 5.1, we analyze the *subject reduction (SR)* property, and we prove that reduction preserves typing and decreases the size of type derivations (that is why we call it weighted SR). The proof of this property makes use of two fundamental properties (Lemmas 5.2 and 5.3) guaranteeing well-typedness of the meta-operations of substitution and replacement. Section 5.2 is devoted to *subject expansion (SE)*, which states that non-erasing anti-reduction preserves types. The proof uses the fact that reverse substitution (Lemma 5.5) and reverse replacement (Lemma 5.6) preserve types.

We start by stating an interesting property, to be used in our forthcoming lemmas, that allows us to split and merge auxiliary derivations typing the same term:

**Lemma 5.1.** *Let* $\mathcal{I} = \wedge_{k\in K}\mathcal{I}_k$. *Then* $\Phi \triangleright \Gamma \Vdash t : \mathcal{I} \mid \Delta$ *iff there exist* $(\Gamma_k)_{k\in K}, (\Delta_k)_{k\in K}$ *s.t.* $(\Phi_k \triangleright \Gamma_k \Vdash t : \mathcal{I}_k \mid \Delta_k)_{k\in K}$, $\Gamma = \wedge_{k\in K}\Gamma_k$ *and* $\Delta = \vee_{k\in K}\Delta_k$. *Moreover,* $\mathtt{sz}\,(\Phi) = \Sigma_{k\in K}\mathtt{sz}\,(\Phi_k)$.

### 5.1. Forward Properties.

We first state the substitution lemma, which guarantees that typing is stable by substitution. The lemma also establishes the size of the derivation tree of a substituted object from the sizes of the derivations trees of its components.

**Lemma 5.2 (Substitution).** *Let* $\Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$. *If* $\Phi_o \triangleright \Gamma; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta$, *then there is* $\Phi_{o\{x/u\}}$ *such that*

- $\Phi_{o\{x/u\}} \rhd \Gamma \wedge \Gamma_u \vdash o\{x/u\} : \mathcal{A} \mid \Delta \vee \Delta_u$.
- $\mathtt{sz}\left(\Phi_{o\{x/u\}}\right) = \mathtt{sz}\,(\Phi_o) + \mathtt{sz}\,(\Theta_u) - |\mathcal{I}|$.

*Proof.* By induction on $\Phi_o$ using Lemmas 4.6 and 5.1. See the Appendix for details. □

Typing is also stable by replacement. Moreover, we can specify the exact size of the derivation tree of the replaced object from the sizes of its components.

**Lemma 5.3 (Replacement).** *Let* $\Theta_u \triangleright \Gamma_u \Vdash u : \wedge_{k\in K}\ (\mathcal{I}_k^*) \mid \Delta_u$ *where* $\alpha \notin \mathtt{fn}(u)$. *If* $\Phi_o \triangleright \Gamma \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta$, *then there is* $\Phi_{o\{\alpha /\!\!/ u\}}$ *such that :*

- $\Phi_{o\{\alpha /\!\!/ u\}} \triangleright \Gamma \wedge \Gamma_u \vdash o\{\alpha /\!\!/ u\} : \mathcal{A} \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta \vee \Delta_u$.
- $\mathtt{sz}\left(\Phi_{o\{\alpha /\!\!/ u\}}\right) = \mathtt{sz}\,(\Phi_o) + \mathtt{sz}\,(\Theta_u)$.

*Proof.* By induction on $\Phi$ using Lemmas 4.6 and 5.1. See the Appendix for details. □

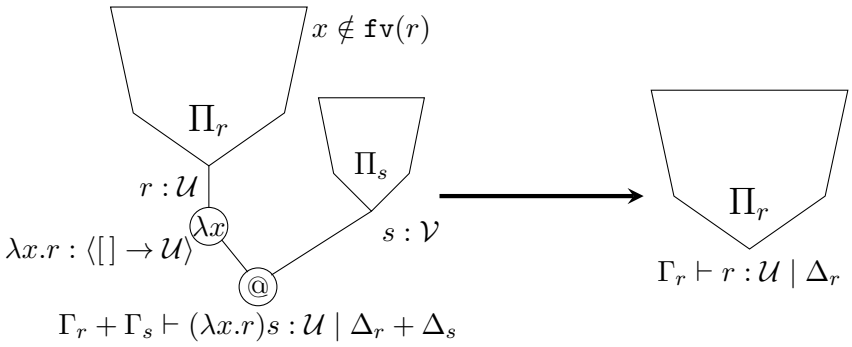


Figure 10: Proof reduction (erasing $\beta$-redex)

Notice that the type of $\alpha$ in the conclusion of the derivation $\Phi_{o\{\alpha/\!/u\}}$ (which is $\vee_{k\in K}\mathcal{V}_k$) is strictly smaller than that of the conclusion of the derivation $\Phi_o$ (which is $\langle \mathcal{I}_k \Rightarrow \mathcal{V}_k \rangle_{k\in K}$) if and only if $K \neq \emptyset$. Lemmas 5.2 and 5.3 are used in the proof of the following key property.

**Property 5.4 (Weighted Subject Reduction for $\mathcal{S}_{\lambda\mu}$).** Let $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$. If $o \to o'$ is a non-erasing step, then there exists a derivation $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ such that $\mathtt{sz}(\Phi) > \mathtt{sz}(\Phi')$.

*Proof.* By induction on $o \to o'$ using Lemmas 4.6, 5.2 and 5.3. See the Appendix for details. □

It is now worth discussing the erasing cases. Note that variable and name assignments are not necessarily preserved by erasing reductions. For example, consider $t = (\lambda y.x)z \to x = t'$. The term $t$ is typed with a variable assignment whose domain is $\{x, z\}$, while $t'$ can only be typed with an assignment whose domain is $\{x\}$. Concretely, starting from a derivation of $x : [\langle a \rangle], z : [\langle b \rangle] \vdash (\lambda y.x)z : \langle a \rangle$ (see Example **??** on page **??**), we can derive $x : [\langle a \rangle] \vdash x : \langle a \rangle$ but not $x : [\langle a \rangle], z : [\langle b \rangle] \vdash x : \langle a \rangle$, so that the type is preserved while the variable assignment is not.

Type systems lacking (full) subject reduction are unusual, but (1) our restricted form of subject reduction, for non-erasing steps only, is sufficient for our purpose (see how we deal with the erasing steps in the proof of Lemma 6.2), (2) strong normalization differs from head normalization in that, whereas "$(\lambda x.r)s$ is HN" is equivalent to "$r\{x/r\}$ is HN", "$(\lambda x.r)s$ is SN" not equivalent to "$r\{x/s\}$ is SN" (except in the non-erasing cases). In other words, contrary to the HN case, erasing reductions lose information about the fact that terms are or are not SN. This loss of information (occurring in the erasing case) is what makes subject reduction in system $\mathcal{S}_{\lambda\mu}$ fail in general. This is illustrated in Fig. **??**: there are two derivations $\Pi_r \triangleright \Gamma_r \vdash r : \mathcal{U} \mid$ and $\Pi_s \triangleright \Gamma_s \vdash s : \mathcal{V} \mid \Delta_s$ with $x \notin \mathtt{fv}(r)$, which are subderivations of $\Pi$ typing the redex $(\lambda x.r)s$, which concludes with $\Gamma_r + \Gamma_s \vdash (\lambda x.s)r \mid \Delta_r + \Delta_s$ (this is represented on the left-hand side of Fig. **??**). The type $\mathcal{V}$ can be seen as an instance of the choice operator $[]^*$. The derivation typing the reduct $r$ is just $\Pi_r$ (on the right-hand side). All typing information pertaining to $s$ has disappeared (*i.e.* has been lost), which explains why the variable/name assignments are modified ($\Gamma_r/\Delta_r$ instead of $\Gamma_r + \Gamma_s/\Delta_r + \Delta_s$). We will come back to the semantics of strong normalization in Section **??**.

5.2. **Backward Properties.** Subject expansion is based on two technical properties: the first one, called reverse substitution, allows us to extract type information for an object $o$ and a term $u$ from the type derivation of $o\{x/u\}$; similarly, the second one, called reverse replacement, gives type information for a command $c$ and a term $u$ from the type derivation of $c\{\alpha /\!\!/ u\}$. Both of them are proved by induction on derivations using Lemmas 4.6 and 5.1. Formally,

**Lemma 5.5** (**Reverse Substitution**). *Let* $\Phi' \triangleright \Gamma' \vdash o\{x/u\} : \mathcal{A} \mid \Delta'$ *Then there exist* $\Gamma, \Delta, \mathcal{I}, \Gamma_u, \Delta_u$ *such that:*

- $\Gamma' = \Gamma \wedge \Gamma_u$,
- $\Delta' = \Delta \vee \Delta_u$,
- $\triangleright\Gamma; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta$
- $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$.

*Proof.* By induction on $\Phi'$ using Lemmas 4.6 and 5.1. See the Appendix for details. □

**Lemma 5.6** (**Reverse Replacement**). *Let* $\Phi' \triangleright \Gamma' \vdash o\{\alpha /\!\!/ u\} : \mathcal{A} \mid \alpha : \mathcal{V}; \Delta'$, *where* $\alpha \notin \mathtt{fn}(u)$. *Then there exist* $\Gamma, \Delta, \Gamma_u, \Delta_u, (\mathcal{I}_k)_{k\in K}, (\mathcal{V}_k)_{k\in K}$ *such that:*

- $\Gamma' = \Gamma \wedge \Gamma_u$,
- $\Delta' = \Delta \vee \Delta_u$,
- $\mathcal{V} = \vee_{k\in K}\mathcal{V}_k$,
- $\triangleright\Gamma \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta$, *and*
- $\triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$

*Proof.* By induction on $\Phi'$ using Lemmas 4.6 and 5.1. See the Appendix for details. □

The following property will be used in Section 6 to show that normalization implies typability.

**Property 5.7** (**Subject Expansion for $\mathcal{S}_{\lambda\mu}$**). Assume $\Phi' \triangleright \Gamma' \vdash o' : \mathcal{A} \mid \Delta'$. If $o \to o'$ is a non-erasing step, then there is $\Phi \triangleright \Gamma' \vdash o : \mathcal{A} \mid \Delta'$.

*Proof.* By induction on $\to$ using Lemma 4.6, 5.5 and 5.6. See the Appendix for details. □

## 6. Strongly Normalizing $\lambda\mu$-Objects

6.1. **Type-theoretic characterization, with quantitative bounds.** In this section, we show the characterization of strongly-normalizing terms of the $\lambda\mu$-calculus by means of the typing system introduced in Section 4, *i.e.* we show that an $\lambda\mu$-object $o$ is strongly-normalizing iff $t$ is $\mathcal{S}_{\lambda\mu}$-typable.

As defined in Section 2, for any $\lambda\mu$-object $o$, we write $\eta(o)$ for the length of the maximal reduction sequence starting at $o$. Remark that $\eta(o) < \infty$ iff $o \in \mathcal{SN}(\lambda\mu)$. The following equations will play a key role in our proof of Lemma 6.2.

**Lemma 6.1.**

$$\begin{array}{lcll}
\eta(x\,t_1\dots t_n) & = & +_{i=1\dots n}\eta(t_i) & \\
\eta(\lambda x.t) & = & \eta(t) & \\
\eta(\mu\alpha.c) & = & \eta(c) & \\
\eta([\alpha]t) & = & \eta(t) & \\
\eta((\lambda x.t)u\,\overline{v}) & = & \eta(t\{x/u\}\overline{v})+1 & \textit{if } x\in \mathtt{fv}(t)\\
\eta((\lambda x.t)u\,\overline{v}) & = & \eta(u)+\eta(t\overline{v})+1 & \textit{if } x\notin \mathtt{fv}(t)\\
\eta((\mu\alpha.c)u\overline{v}) & = & \eta((\mu\alpha.c\{\alpha/\!\!/u\})\overline{v})+1 & \textit{if } \alpha\in \mathtt{fv}(c)\\
\eta((\mu\alpha.c)u\overline{v}) & = & \eta(u)+\eta((\mu\alpha.c)\overline{v})+1 & \textit{if } \alpha\notin \mathtt{fv}(c)
\end{array}$$

*Proof.* The proof follows the same lines of that of the $\lambda$-calculus (see [50], Fundamental Lemma of Maximality 3.18). We only briefly discuss the two last cases.

- Let $o=(\mu\alpha.c)u\,\overline{v}$, where $\alpha\in\mathtt{fv}(c)$. Take any reduction sequence $\rho$ to normal form starting at $o$, it is straightforward to see that $\rho$ must reduce the head redex $(\mu\alpha.c)u$, so that $\rho$ has necessarily the form $o\to^*_{\lambda\mu}(\mu\alpha.c')u'\,\overline{v'}\to_{\lambda\mu}(\mu\alpha.c'\{\alpha/\!\!/u'\})\overline{v'}\to_{\lambda\mu}\dots$, where $c\to^*_{\lambda\mu}c'$, $u\to^*_{\lambda\mu}u$ and $\overline{v}\to^*_{\lambda\mu}\overline{v'}$. Then $\rho$ can be transformed into another (potentially longer) reduction sequence $\rho'$ of the form $o\to_{\lambda\mu}(\mu\alpha.c\{\alpha/\!\!/u\})\overline{v}\to^*_{\lambda\mu}\mu\alpha.c'\{\alpha/\!\!/u'\})\overline{v'}\to_{\lambda\mu}\dots$. Thus $\eta((\mu\alpha.c)u\,\overline{v})\le\eta((\mu\alpha.c\{\alpha/\!\!/u\})\overline{v})+1$ holds. The converse inequality is easy.
- Let $o=(\mu\alpha.c)u\,\overline{v}$, where $\alpha\notin\mathtt{fv}(c)$. Again, take any reduction sequence $\rho$ to normal form starting at $o$, it is again straightforward to see that $\rho$ must reduce the head redex $(\mu\alpha.c)u$, so that $\rho$ has necessarily the form $o\to^*_{\lambda\mu}(\mu\alpha.c')u'\,\overline{v'}\to_{\lambda\mu}(\mu\alpha.c')\overline{v'}\to_{\lambda\mu}\dots$, where $c\to^*_{\lambda\mu}c'$, $u\to^*_{\lambda\mu}u$ and $\overline{v}\to^*_{\lambda\mu}\overline{v'}$. Then $\rho$ can be transformed into another (potentially longer) reduction sequence $\rho'$ of the form $o\to^*_{\lambda\mu}(\mu\alpha.c)\mathtt{nf}(u)\,\overline{v}\to_{\lambda\mu}(\mu\alpha.c)\overline{v}\to^*_{\lambda\mu}(\mu\alpha.c')\overline{v'}\to_{\lambda\mu}\dots$, where $\mathtt{nf}(u)$ denotes the normal form of $u$. Thus, $\eta((\mu\alpha.c)u\,\overline{v})\le\eta(u)+\eta((\mu\alpha.c)\overline{v})+1$. The converse inequality is easy.

□

The proof of our main result (Theorem 6.4) relies on the following two ingredients:

- Every $\mathcal{S}_{\lambda\mu}$-typable object is in $\mathcal{SN}(\lambda\mu)$ (Lemma 6.2).
- Every object in $\mathcal{SN}(\lambda\mu)$ is $\mathcal{S}_{\lambda\mu}$-typable (Lemma 6.3).

We first show that any typable object $o$ belongs to $\mathcal{SN}(\lambda\mu)$.

**Lemma 6.2.** *If $o$ is $\mathcal{S}_{\lambda\mu}$-typable, i.e. $\Phi\triangleright\Gamma\vdash o:\mathcal{A}\mid\Delta$, then $\eta(o)\le\mathtt{sz}(\Phi)$. Thus $o\in\mathcal{SN}(\lambda\mu)$.*

*Proof.* We show $\eta(o)\le\mathtt{sz}(\Phi)$ by induction on $\mathtt{sz}(\Phi)$, where $\Phi\triangleright\Gamma\vdash o:\mathcal{A}\mid\Delta$. When $\Phi$ does not end with the rule $(\Rightarrow_{\mathtt{e}})$ the proof holds straightforwardly by the *i.h.* so we consider that $\Phi$ ends with $(\Rightarrow_{\mathtt{e}})$, where $\mathcal{A}=\mathcal{U}$ and $o=x\,t_1\dots t_n$ or $o=(\mu\alpha.c)t_1\dots t_n$ or $o=(\lambda x.u)t_1\dots t_n$, with $n\ge1$.

In all the three cases for $o$, there are subderivations $(\Phi_i)_{i\in\{1\dots n\}}$ such that $\mathtt{sz}(\Phi_i)<\mathtt{sz}(\Phi)$ so that the *i.h.* gives $\eta(t_i)\le\mathtt{sz}(\Phi_i)$. Now, there are three different cases to consider:

(1) If $o=x\,t_1\dots t_n$, then there are non-empty subderivations $\Phi_1\dots\Phi_n$ of $\Phi$ typing $t_1\dots t_n$ respectively. Since $+_{i=1\dots n}\mathtt{sz}(\Phi_i)+n+1\le\mathtt{sz}(\Phi)$, the *i.h.* gives $\eta(t_i)\le\mathtt{sz}(\Phi_i)$ for $1\le i\le n$. We conclude since $\eta(x\,t_1\dots t_n)=+_{i=1\dots n}\eta(t_i)$.

(2) If $o = (\mu\alpha.c)t_1 \dots t_n$, there are two cases:
- $\alpha \in \mathtt{fn}(c)$. Using Property 5.4, we get $\Phi' \triangleright \Gamma \vdash (\mu\alpha.c\{\alpha /\!\!/ t_1\})t_2 \dots t_n : \mathcal{U} \mid \Delta$ and $\mathtt{sz}(\Phi') < \mathtt{sz}(\Phi)$. Then the *i.h.* gives $\eta((\mu\alpha.c\{\alpha /\!\!/ t_1\})t_2 \dots t_n) \leq \mathtt{sz}(\Phi')$. We conclude since $\eta(o) = \eta((\mu\alpha.c\{\alpha /\!\!/ t_1\})t_2 \dots t_n) + 1 \leq \mathtt{sz}(\Phi') + 1 \leq \mathtt{sz}(\Phi)$.
- $\alpha \notin \mathtt{fn}(c)$. Then $\Phi$ is of the form:

$$\dfrac{\dfrac{\dfrac{\Phi_{\mu\alpha.c} \triangleright \dots \vdash \mu\alpha.c : \langle\xi_0\rangle \mid \dots \qquad \dfrac{\Phi_1 \triangleright \dots \vdash t_1 : \mathcal{V}_1 \mid \dots}{\dots \vdash t_1 : [\mathcal{V}_1] \mid \dots}}{\dots \vdash (\mu\alpha.c)t_1 : \langle\xi_1\rangle \mid \dots}}{\vdots} \qquad \dfrac{\Phi_n \triangleright \dots \vdash t_n : \mathcal{V}_n \mid \dots}{\dots \vdash t_n : [\mathcal{V}_n] \mid \dots}}{\dots \vdash o : \langle\xi_n\rangle \mid \dots}$$

with $\mathcal{U} = \langle\xi_n\rangle$. The type $\langle\xi_0\rangle$ is obtained by choice because Lemma 4.6 guarantees that no $\alpha$ is typed in the name assignments of the left derivations. Note that $\mathtt{sz}(\Phi) = +_{i=1\dots n}\mathtt{sz}(\Phi_i) + \mathtt{sz}(\Phi_{\mu\alpha.c}) + n$ since the $\xi_i$ are all blind. We then build the following derivation:

$$\Phi_{o'} = \dfrac{\dfrac{\dfrac{\Phi'_{\mu\alpha.c} \triangleright \dots \vdash \mu\alpha.c : \langle\xi_1\rangle \mid \dots \qquad \dfrac{\Phi_2 \triangleright \dots \vdash t_2 : \mathcal{V}_2 \mid \dots}{\dots \vdash t_2 : [\mathcal{V}_2] \mid \dots}}{\vdots} \qquad \dfrac{\Phi_n \triangleright \dots \vdash t_n : \mathcal{V}_n \mid \dots}{\dots \vdash t_n : [\mathcal{V}_n] \mid \dots}}{\dots \vdash o' : \langle\xi_n\rangle \mid \dots}$$

with $o' = (\mu\alpha.c)t_2 \dots t_n$ and $\Phi'_{\mu\alpha.c}$ the derivation $\Phi_{\mu\alpha.c}$ where the choice operator raises $\xi_1$ instead of $\xi_0$ (actually, any blind type of arity $\geq n-1$ would do). In particular, $\mathtt{sz}\left(\Phi'_{\mu\alpha.c}\right) = \mathtt{sz}(\Phi_{\mu\alpha.c})$, so that $\mathtt{sz}(\Phi_{o'}) = +_{i=2\dots n}\mathtt{sz}(\Phi_i) + \mathtt{sz}(\Phi_{\mu\alpha.c}) + n - 1 < \mathtt{sz}(\Phi)$. We also have $\mathtt{sz}(\Phi_1) < \mathtt{sz}(\Phi)$. The *i.h.* then gives $\eta(o') \leq \mathtt{sz}(\Phi_{o'})$ and $\eta(t_1) \leq \mathtt{sz}(\Phi_1)$. We conclude since:

$$\begin{array}{rll} \eta((\mu\alpha.c)t_1 \dots t_n) = & \eta((\mu\alpha.c)t_2 \dots t_n) + \eta(t_1) + 1 \\ \leq_{i.h.} & \mathtt{sz}(\Phi_{o'}) + \mathtt{sz}(\Phi_1) + 1 \\ = & +_{i=2\dots n}\mathtt{sz}(\Phi_i) + \mathtt{sz}(\Phi_{\mu\alpha.c}) + n - 1 + \mathtt{sz}(\Phi_1) + 1 \\ = & \mathtt{sz}(\Phi) \end{array}$$

(3) If $o = (\lambda x.u)t_1 \dots t_n$, we reason similarly to the previous case.

□

**Lemma 6.3.** *If $o \in \mathcal{SN}(\lambda\mu)$, then $o$ is $\mathcal{S}_{\lambda\mu}$-typable.*

*Proof.* By induction on $\langle \eta(o), |o| \rangle$. If $o$ is not an application $t\,u$, the inductive step is straightforward. We only detail the case when $o$ is an application.

- If $o = x\,t_1 \dots t_n$, the property is straightforward.
- If $o = (\mu\alpha.c)t_1 \dots t_n$, we set $o' = (\mu\alpha.c\{\alpha /\!\!/ t_1\})t_2 \dots t_n$. There are two cases:
  - $\alpha \in \mathtt{fn}(c)$. Then $\eta(o) = \eta(o') + 1$. By the *i.h.*, there is $\Phi_{o'} \triangleright \Gamma' \vdash o' : \mathcal{U} \mid \Delta'$. By Property 5.7, there is $\Phi \triangleright \Gamma' \vdash o : \mathcal{U} \mid \Delta'$ and we are done.
  - $\alpha \notin \mathtt{fn}(c)$. Then $o' = (\mu\alpha.c)t_2 \dots t_n$ and $\eta(o) = \eta(o') + \eta(t_1) + 1$. By the *i.h.*, there are typing derivations $\Phi_{o'} \triangleright \Gamma' \vdash o' : \mathcal{U} \mid \Delta'$ and $\Phi_1 \triangleright \Gamma_1 \vdash t_1 : \mathcal{V}_1 \mid \Delta_1$. Since $\alpha \notin \mathtt{fn}(c)$,

Lemma 4.6 entails that $\Phi_{o'}$ is of the following form, where $\xi_1$ is a blind type:

$$\dfrac{\dfrac{\Phi'_{\mu\alpha.c} \rhd \mu\alpha.c : \langle \xi_1 \rangle \qquad \dfrac{\Phi_2 \rhd t_2 : \mathcal{V}_2}{t_2 : [\mathcal{V}_2]}}{\vdots} \qquad \dfrac{\Phi_n \rhd t_n : \mathcal{V}_n}{t_n : [\mathcal{V}_n]}}{\Gamma' \vdash o : \langle \xi_n \rangle \mid \Delta'}$$

We then set $\Phi$ as follows:

$$\dfrac{\dfrac{\dfrac{\Phi_{\mu\alpha.c} \rhd \mu\alpha.c : \langle \xi_0 \rangle \qquad \dfrac{\Phi_1 \rhd \Gamma_1 \vdash t_1 : \mathcal{V}_1 \mid \Delta_1}{t_1 : [\mathcal{V}_1]}}{(\mu\alpha.c)t_1 : \langle \xi_1 \rangle} \qquad \dfrac{\Phi_2 \rhd t_2 : \mathcal{V}_2}{t_2 : [\mathcal{V}_2]}}{\vdots} \qquad \dfrac{\Phi_n \rhd t_n : \mathcal{V}_n}{t_n : [\mathcal{V}_n]}}{\Gamma' + \Gamma_1 \vdash o : \langle \xi_n \rangle \mid \Delta' + \Delta_1}$$

where $\Phi_{\mu\alpha.c}$ is exactly as $\Phi'_{\mu\alpha.c}$ except that we raise the blind type $\xi_0 = [\,] \to \langle \xi_1 \rangle$ instead of $\xi_1$ (actually, any blind type of arity $\geq n$ would do).

- If $o = (\lambda x.t)u\,\overline{v}$, we reason similarly to the previous case.

□

Lemmas 6.2 and 6.3 allow us to conclude with the main result of this paper which is the equivalence between typability and strong-normalization for the $\lambda\mu$-calculus. Notice that no reducibility argument was used in the whole proof.

**Theorem 6.4.** *Let $o \in \mathcal{O}_{\lambda\mu}$. Then $o$ is typable in system $\mathcal{S}_{\lambda\mu}$ iff $o \in \mathcal{SN}(\lambda\mu)$. Moreover, if $o$ is $\mathcal{S}_{\lambda\mu}$-typable with tree derivation $\Pi$, then $\mathtt{sz}(\Pi)$ gives an upper bound to the maximal length of a reduction sequence starting at $o$.*

6.2. **Discussion.** As we have observed in the end of Section 5.1, subject reduction for erasing steps fails in system $\mathcal{S}_{\lambda\mu}$. The same is true for subject expansion. This naturally rise the question: why subject reduction and expansion should hold in system $\mathcal{H}_{\lambda\mu}$ (related to HN) and not in $\mathcal{S}_{\lambda\mu}$ (related to SN)? The difference of treatment lies in the semantics of what these two systems are designed to capture:

- $\mathcal{H}_{\lambda\mu}$ captures head normalization (Theorem 4.4) *i.e.* a $\lambda\mu$-object $o$ reduces to a HNF iff $o$ is $\mathcal{H}_{\lambda\mu}$-typable. The first crucial observation about these first results is that no "semantic information" with respect to head normalization is lost during (arbitrary) reduction. We make this notion more concrete below. Indeed, if $o \to o'$ (whether this step is erasing or not), then, by confluence, $o$ is HN iff $o'$ is HN, and this is why we want both subject reduction and subject expansion for $\mathcal{H}_{\lambda\mu}$ to hold. Moreover, $o$ and $o'$ have head normal forms of the same shape. Intuitively, it is no more difficult to prove that $o'$ is HN than to prove that $o$ is HN, and vice-versa. Of course, the step $o \to o'$ may erase some reduction paths that exist in $o$, but these (erased) reduction paths are irrelevant to the fact that $o$ is HN or not: no semantic information pertaining to HN has been lost. This situation is very different in system $\mathcal{S}_{\lambda\mu}$.

- $\mathcal{S}_{\lambda\mu}$ captures strong normalization (Theorem 6.4), which is different from weak and head normalization because it is a property about the finiteness of *all* reduction paths, and not about the existence of *at least one* reduction path to a normal form. This difference is materialized by the following three observations.
  - If $o \to o'$, then "$o$ is SN" and "$o'$ is SN" are not equivalent propositions, *e.g.*$(\lambda x.y)\Omega \to y$, $(\lambda x.y)\Omega$ is not SN whereas $y$ is. In particular, no type system characterizing SN satisfies subject expansion.
  - If $o \to o'$ and "$o$ is SN", then "$o'$ is always SN", but some important semantic information may be lost with respect to strong normalization! For instance, if $u$ is SN but the normal form of $u$ cannot be reached in less than 1000 reductions steps, then $(\lambda x.y)u$ is also SN, but the (erasing) reduction step $(\lambda x.y)u \to y$ loses information regarding the reduction paths starting at $(\lambda x.y)u$: the reduction obliterates reduction paths in $u$ (intuitively, on may not know if $y$ originates from $(\lambda x.y)u$, or $(\lambda x.y)x$, or $(\lambda x.y)\Omega\dots$). This is a loss, which explains why full subject reduction does not hold for system $\mathcal{S}_{\lambda\mu}$. Moreover, this also suggests that full subject reduction would arguably be less *faithful* to the semantics of strong normalization. Note that (full) subject reduction could be obtained by just allowing weakening in the typing system, while preserving the characterization theorem. But weakening does not restore subject expansion, since SN is not stable under expansion.
  - If $o \to o'$ and "$o'$ is SN implies $o$ is SN", then the reduction step is called *perpetual*: perpetual strategies are precisely those that are used to study strong normalization.

  Again, it is interesting to note that, when $o \to o'$ is *non-erasing*, then $o$ is SN iff $o'$ is SN, and any reduction path in $o$ has residuals in $o'$. This explains why a typing system for strong normalization should satisfy subject reduction and subject expansion for *non-erasing* steps (which is sufficient to prove the characterization theorem), while this is not necessary for erasing steps.

These observations summarize why our typing systems enjoy full subject reduction and subject expansion in one case ($\mathcal{H}_{\lambda\mu}$) and only subject reduction and subject expansion for non-erasing steps in the other ($\mathcal{S}_{\lambda\mu}$).

## 7. The $\lambda\mu_{\mathtt{s}}$-calculus

This section introduces the syntax (Section 7.1) and the operational semantics (Section 7.2) of the $\lambda\mu_{\mathtt{s}}$-calculus, a small-step refinement of $\lambda\mu$, for which the typing system $\mathcal{S}_{\lambda\mu}$ naturally extends. The restriction of the $\lambda\mu_{\mathtt{s}}$-calculus to intuitionistic logic is known as the *linear substitution calculus* [1], deeply studied in rewriting theory and complexity analysis.

7.1. **Syntax.** The set of **objects** ($\mathcal{O}_{\lambda\mu_{\mathtt{s}}}$), **terms** ($\mathcal{T}_{\lambda\mu_{\mathtt{s}}}$) and **commands** ($\mathcal{C}_{\lambda\mu_{\mathtt{s}}}$) of the $\lambda\mu_{\mathtt{s}}$-calculus are given by the following grammars

$$\begin{array}{llll} (\textbf{objects}) & o & ::= & t \mid c \\ (\textbf{terms}) & t, u & ::= & x \mid \lambda x.t \mid tu \mid \mu\alpha.c \mid t[x/u] \\ (\textbf{commands}) & c & ::= & [\alpha]t \mid c\langle\alpha /\!\!/ \beta.u\rangle \end{array}$$

The construction $[x/u]$ (resp. $\langle\alpha /\!\!/ \beta.u\rangle$) is called an **explicit substitution (ES)** (resp. **explicit replacement (ER)**). Remark that ES do not apply to commands and ER do not apply to terms. An ES $[x/u]$ implements the *meta-substitution* operator $\{x/u\}$ while an ER $\langle\alpha /\!\!/ \beta.u\rangle$ implements the *fresh* replacement meta-operator $\{\alpha /\!\!/ \beta.u\}$ introduced in

Section 2.2, *i.e.* the small step computation of $c\langle\alpha /\!\!/ \beta.u\rangle$ replaces only one occurrence of $[\alpha]t$ inside $c$ by $[\beta]t\langle\alpha /\!\!/ \beta.u\rangle u$. As in Section 2.1, the **size of an object** $o$ is denoted by $|o|$.

The notions of **free** and **bound variables** and **names** are extended as expected, in particular $\mathtt{fv}(t[x/u]) := (\mathtt{fv}(t)\setminus\{x\})\cup\mathtt{fv}(u)$ and $\mathtt{fn}(c\langle\alpha /\!\!/ \beta.u\rangle) := (\mathtt{fn}(c)\setminus\{\alpha\})\cup\{\beta\}\cup\mathtt{fn}(u)$. The derived notion of $\alpha$-conversion (*i.e.* renaming of bound variables and names) will be assumed in the rest of the paper. Thus *e.g.* $([\gamma]x[x/y])\langle\gamma /\!\!/ \beta.z\rangle =_\alpha ([\gamma']x'[x'/y])\langle\gamma' /\!\!/ \beta.z\rangle$. The **number of free occurrences** of the variable $x$ (resp. the name $\alpha$) in $o$ is denoted by $|o|_x$ (resp. $|o|_\alpha$).

**List** ($\mathtt{L}$), **term** ($\mathtt{TT}, \mathtt{CT}, \mathtt{OT}$), and **command** ($\mathtt{TC}, \mathtt{CC}, \mathtt{OC}$) **contexts** are respectively defined by the following grammars:

$$\begin{array}{l}
\mathtt{L} ::= \Box \mid \mathtt{L}[x/u] \\
\mathtt{TT} ::= \Box \mid \lambda x.\mathtt{TT} \mid \mathtt{TT}\, t \mid t\, \mathtt{TT} \mid \mu\alpha.\mathtt{CT} \mid \mathtt{TT}[x/t] \mid t[x/\mathtt{TT}] \\
\mathtt{CT} ::= [\alpha]\mathtt{TT} \mid \mathtt{CT}\langle\alpha /\!\!/ \beta.u\rangle \mid c\langle\alpha /\!\!/ \beta.\mathtt{TT}\rangle \\
\mathtt{OT} ::= \mathtt{TT} \mid \mathtt{CT} \\
\mathtt{TC} ::= \lambda x.\mathtt{TC} \mid \mathtt{TC}\, t \mid t\, \mathtt{TC} \mid \mu\alpha.\mathtt{CC} \mid \mathtt{TC}[x/t] \mid t[x/\mathtt{TC}] \\
\mathtt{CC} ::= \boxdot \mid [\alpha]\mathtt{TC} \mid \mathtt{CC}\langle\alpha /\!\!/ \beta.u\rangle \mid c\langle\alpha /\!\!/ \beta.\mathtt{TC}\rangle \\
\mathtt{OC} ::= \mathtt{TC} \mid \mathtt{CC}
\end{array}$$

The hole $\Box$ (resp. $\boxdot$) can be replaced by a term (resp. a command). Indeed, $\mathtt{L}[t]$ denotes the replacement of $\Box$ in $\mathtt{L}$ by the term $t$ (similarly for $\mathtt{TT}[t]$, $\mathtt{CT}[t]$ and $\mathtt{OT}$), while $\mathtt{CC}[c]$ denotes the replacement of $\boxdot$ in $\mathtt{CC}$ by the command $c$ (similarly for $\mathtt{TC}$ and $\mathtt{OC}$). Every meta-expression $\mathtt{XY}$ with $\mathtt{X} \in \{\mathtt{T}, \mathtt{C}\}$ and $\mathtt{Y} \in \{\mathtt{T}, \mathtt{C}\}$ must be interpreted as a context taking an object $\mathtt{Y}$ and yielding an object $\mathtt{X}$: *e.g.* $\mathtt{TC}$ denotes a context that takes a command ($\mathtt{C}$ on the right) and outputs a term ($\mathtt{T}$ on the left).

We write $\mathtt{OT}^{\mathcal{S}}$ for a term context $\mathtt{OT}$ which does not capture the free variables and names in the set $\mathcal{S}$, *i.e.* there are no abstractions and substitutions in the context that bind the symbols in $\mathcal{S}$. For instance $\mathtt{TT} = \lambda y.\Box$ can be specified as $\mathtt{TT}^x$ while $\mathtt{TT} = \lambda x.\Box$ cannot. In order to emphasize this particular property we may write $\mathtt{TT}^{\mathcal{S}}[\![t]\!]$ instead of $\mathtt{TT}^{\mathcal{S}}[t]$, and we may omit $\mathcal{S}$ when it is clear from the context. Same concepts apply to command contexts, *i.e.* $\mathtt{OC}^{\mathcal{S}}$ does not capture the variables and names in $\mathcal{S}$ and the notation used for that is $\mathtt{OC}^{\mathcal{S}}[\![c]\!]$.

7.2. **Operational Semantics.** The reduction rules of the $\lambda\mu_{\mathtt{s}}$-calculus aim to give a small-step semantics to the $\lambda\mu$-calculus, based on the *substitution/replacement at a distance* paradigm [2, 1]. The reduction relation $\lambda\mu_{\mathtt{s}}$ of the calculus is given by the context closure of the following rewriting rules.

$$\begin{array}{lll}
\mathtt{L}[\lambda x.t]\, u & \mapsto_{\mathtt{B}} \mathtt{L}[t[x/u]] & \\
\mathtt{TT}[\![x]\!][x/u] & \mapsto_{\mathtt{c}_v} \mathtt{TT}[\![u]\!][x/u] & \text{if } |\mathtt{TT}[\![x]\!]|_x > 1 \\
\mathtt{TT}[\![x]\!][x/u] & \mapsto_{\mathtt{d}_v} \mathtt{TT}[\![u]\!] & \text{if } |\mathtt{TT}[\![x]\!]|_x = 1 \\
t[x/u] & \mapsto_{\mathtt{w}_v} t & \text{if } x \notin \mathtt{fv}(t) \\
\mathtt{L}[\mu\alpha.c]\, u & \mapsto_{\mathtt{M}} \mathtt{L}[\mu\gamma.c\langle\alpha /\!\!/ \gamma.u\rangle] & \text{if } \gamma \text{ is fresh} \\
\mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \gamma.u\rangle & \mapsto_{\mathtt{c}_n} \mathtt{CC}[\![[\gamma]tu]\!]\langle\alpha /\!\!/ \gamma.u\rangle & \text{if } |\mathtt{CC}[\![[\alpha]t]\!]|_\alpha > 1 \\
\mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \gamma.u\rangle & \mapsto_{\mathtt{d}_n} \mathtt{CC}[\![[\gamma]tu]\!] & \text{if } |\mathtt{CC}[\![[\alpha]t]\!]|_\alpha = 1 \\
c\langle\alpha /\!\!/ \gamma.u\rangle & \mapsto_{\mathtt{w}_n} c & \text{if } \alpha \notin \mathtt{fn}(c)
\end{array}$$

where $\mathtt{TT}$ is to be understood as $\mathtt{TT}^x$ and $\mathtt{CC}$ as $\mathtt{CC}^{\alpha,\gamma}$.

We use $\to_{\mathtt{w}}$ for the reduction relation generated by the set of rules $\{\mapsto_{\mathtt{w}_v}, \mapsto_{\mathtt{w}_n}\}$ and $\to_{\lambda\overline{\mu\mathtt{s}}}$ for the **non-erasing reduction relation** $\to_{\lambda\mu_{\mathtt{s}}} \setminus \to_{\mathtt{w}}$. For instance, the big step reduction

$$(\mu\alpha.[\alpha]x(\mu\beta.[\alpha]\lambda x.xx))u \to_{\mu} \mu\gamma.[\gamma]x(\mu\beta.[\gamma](\lambda x.xx)u))u$$

where $\alpha$ has been alpha-renamed to $\gamma$, can be now emulated by 3 small steps :

$$\begin{array}{l}(\mu\alpha.[\alpha]x(\mu\beta.[\alpha]\lambda x.xx))u\\ \to_{\mathtt{M}} \mu\gamma.([\alpha]x(\mu\beta.[\alpha]\lambda x.xx))\langle\alpha /\!\!/ \gamma.u\rangle\\ \to_{\mathtt{c}_n} \mu\gamma.([\alpha]x(\mu\beta.[\gamma](\lambda x.xx)u))\langle\alpha /\!\!/ \gamma.u\rangle\\ \to_{\mathtt{d}_n} \mu\gamma.[\gamma]x(\mu\beta.[\gamma](\lambda x.xx)u))u\end{array}$$

Notice that the occurrences of $\alpha$ are (arbitrarily) replaced by $\gamma$ one after another, thus replacement is *linearly* processed. When there is just one occurrence of $\alpha$ left, the small reduction step $\mathtt{d}_n$ performs the last replacement and erases the remaining ER $\langle\alpha /\!\!/ \gamma.u\rangle$ to complete the operation.

More generally, not only the syntax of the $\lambda\mu_{\mathtt{s}}$-calculus can be seen as a refinement of the $\lambda\mu$-calculus, but also its operational semantics. Formally,

**Lemma 7.1.** *If $o \in \mathcal{O}_{\lambda\mu}$, then $o \to_{\lambda\mu} o'$ implies $o \to^{+}_{\lambda\mu_{\mathtt{s}}} o'$.*

*Proof.* By induction on the reduction relation $\to_{\lambda\mu}$. □

Moreover, we can project $\lambda\mu_{\mathtt{s}}$-reduction sequences into $\lambda\mu$-reduction sequences. Indeed, consider the projection function $\mathtt{P}(\_)$ computing all the explicit substitutions and replacements of an object, thus in particular $\mathtt{P}(t[x/u]) := \mathtt{P}(t)\{x/\mathtt{P}(u)\}$ and $\mathtt{P}(c\langle\alpha /\!\!/ \alpha'.u\rangle) := \mathtt{P}(c)\{\alpha /\!\!/ \alpha'.\mathtt{P}(u)\}$. Then,

**Lemma 7.2.** *If $o \in \mathcal{O}_{\lambda\mu_{\mathtt{s}}}$, then $o \to_{\lambda\mu_{\mathtt{s}}} o'$ implies $\mathtt{P}(o) \to^{*}_{\lambda\mu} \mathtt{P}(o')$.*

*Proof.* By induction on the reduction relation $\to_{\lambda\mu_{\mathtt{s}}}$. □

7.3. **Typing System.** In this section we extend the (quantitative) typing system $\mathcal{S}_{\lambda\mu}$ in order to capture the $\lambda\mu_{\mathtt{s}}$-calculus, the aim being to characterize the set of strongly $\lambda\mu_{\mathtt{s}}$-normalizing objects by using quantitative arguments.

More precisely, system $\mathcal{S}_{\lambda\mu}$ is enriched with the two typing rules in Figure 10. Rule

$$\frac{\Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t \qquad \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma_t \wedge \Gamma_u \vdash t[x/u] : \mathcal{U} \mid \Delta_t \vee \Delta_u}\ (\mathtt{s})$$

$$\frac{\Gamma_c \vdash c : \# \mid \Delta_c; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K} \quad \Gamma_u \Vdash u : (\wedge_{k\in K}\mathcal{I}_k^*)^* \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash c\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta_c \vee \Delta_u \vee \alpha' : \vee_{k\in K}\mathcal{V}_k}\ (\mathtt{r})$$

Figure 11: Additional Rules for System $\mathcal{S}_{\lambda\mu_{\mathtt{s}}}$

$(\mathtt{s})$ is inspired by the derivation tree typing the term $(\lambda x.t)u$: indeed, any derivation $\triangleright\Gamma \vdash (\lambda x.t)u : \mathcal{V} \mid \Delta$ induces two derivations $\triangleright\Gamma_t, x : \mathcal{I} \vdash t : \mathcal{V} \mid \Delta_t$ and $\triangleright\Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u$, from which we can type $t[x/u]$. Likewise, the rule $(\mathtt{r})$ is motivated by the derivation tree

typing a $\mu$-redex. In particular, when $K = \emptyset$ (*i.e.* when $\alpha \notin \mathtt{fn}(c)$), then $(\wedge_{k\in\emptyset}\mathcal{I}_k^*)^* = [\,]^*$, so that the outer star in $(\wedge_{k\in K}\mathcal{I}_k^*)^*$ gives an arbitrary multiset $[\sigma]$ ensuring the typing (and thus the $SN$ property) of the replacement argument $u$. Notice that Lemma 4.5 still holds for $\mathcal{S}_{\lambda\mu_{\mathtt{s}}}$ *i.e.* if $\rhd_{\mathcal{S}_{\lambda\mu_{\mathtt{s}}}}\Gamma \vdash t : \mathcal{U} \mid \Delta$, then $\mathcal{U} \neq \langle\,\rangle$.

As one may expect, system $\mathcal{S}_{\lambda\mu_{\mathtt{s}}}$ encodes a non-idempotent and relevant system for intuitionistic logic with ES [29]. More precisely, restricting rule ($\mathtt{s}$) to $\lambda$-terms with ES gives the following rule:

$$\frac{\Gamma; x : \mathcal{I} \vdash t : \sigma \qquad \Gamma' \Vdash u : \mathcal{I}^*}{\Gamma \wedge \Gamma' \vdash t[x/u] : \sigma}$$

Relevance also holds for $\lambda\mu_{\mathtt{s}}$:

**Lemma 7.3 (Relevance).** *Let $o \in \mathcal{O}_{\lambda\mu_{\mathtt{s}}}$. If $\Phi \rhd \Gamma \vdash o : \mathcal{A} \mid \Delta$ (resp. $\Phi \rhd \Gamma \Vdash t : \mathcal{I} \mid \Delta$ with $\mathcal{I} \neq [\,]$), then $\mathtt{fv}(o) = \mathtt{dom}(\Gamma)$ and $\mathtt{fn}(o) = \mathtt{dom}(\Delta)$ (resp. $\mathtt{fv}(t) = \mathtt{dom}(\Gamma)$ and $\mathtt{fn}(t) = \mathtt{dom}(\Delta)$).*

*Proof.* By induction on $\Phi$. □

We now extend the function $\mathtt{sz}(\_)$ introduced in Section 4.3 by adding the following cases:

$$\mathtt{sz}\left(\frac{\Phi_t \rhd t \qquad \Phi_u \rhd u}{\Gamma_t \wedge \Gamma_u \vdash t[x/u] : \mathcal{U} \mid \Delta \vee \Delta_u}\,(\mathtt{s})\right) \quad := \quad \mathtt{sz}(\Phi_t) + \mathtt{sz}(\Phi_u)$$

$$\mathtt{sz}\left(\frac{\Phi_c \rhd \Gamma \vdash c : \# \mid \Delta, \alpha : \langle \mathcal{I}_k \to \mathcal{U}\rangle_{k\in K} \qquad \Phi_u \rhd u}{\Gamma_c \wedge \Gamma_u \vdash c\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta_c \vee \Delta_u}\,(\mathtt{r})\right) \quad := \quad \mathtt{sz}(\Phi_c) + \mathtt{sz}(\Phi_u) + |K| - \tfrac{1}{2}$$

Notice that $\mathtt{sz}(\Phi) \geqslant 1$ still holds for any *regular* derivation $\Phi$.

As explained in Section 5.1, weighted subject reduction holds for $\mu$-reduction steps like $t = (\mu\alpha.c)u \to_\mu \mu\gamma.c\{\alpha /\!\!/ \gamma.u\} = t'$ because $\gamma$ is typed in $t'$ with smaller arity than that of $\alpha$ in $t$. The (big) step above is emulated in the $\lambda\mu_{\mathtt{s}}$-calculus by the (small) steps $t \to_{\mathtt{M}} \mu\gamma.c\langle\alpha /\!\!/ \gamma.u\rangle \to^+_{\mathtt{c}_n,\mathtt{d}_n,\mathtt{w}_n} t'$, where $\mathtt{c}_n$ and $\mathtt{d}_n$ perform linear replacements, so they are also naturally expected to decrease the size of type derivations. However, for the first step $t = (\mu\alpha.c)u \to_{\mathtt{M}} \mu\gamma.c\langle\alpha /\!\!/ \gamma.u\rangle = t'$, even if no real replacement has taken place yet, we should still have a quantifiable decrease of the form $\mathtt{sz}(\Phi_t) > \mathtt{sz}(\Phi_{t'})$. This is the reason we use "$-\frac{1}{2}$" when defining the size of explicit replacements, which does not compromise the forthcoming weighted subject reduction property.

One may naively think that the "$-\frac{1}{2}$" component in the size definition of an ER can compromise the decrease of the size for a step $t = \mu\gamma.c\langle\alpha /\!\!/ \gamma.u\rangle \to_{\mathtt{d}_n} \mu\alpha.c\{\alpha /\!\!/ \gamma.u\} = t'$, when $c$ holds exactly one occurrence of $\alpha$ : indeed, removing the ER $\langle\alpha /\!\!/ \gamma.u\rangle$ induces an *increase* of the measure equal to $\frac{1}{2}$. However, the arity contribution of (the unique occurrence of) $\alpha$ in $t$ is greater than that of the new occurrence of $\gamma$ in $t'$: the replacement operation then induces a decrease of the measure which is equal to some $k \geqslant 1$; and thus the overall decrease of the measure is in the worst case $k - \frac{1}{2} > 0$, which still grants $\mathtt{sz}(\Phi) > \mathtt{sz}(\Phi')$. The decrease of the measure for a $\mathtt{w}_n$-step is more evident. Last, but not least, the fact that $\mathtt{sz}(\Phi)$ is a half-integer greater or equal to one ensures that the measure is still well-founded.

## 8. Typing Properties

As in the case of the $\lambda\mu$-calculus, we show that the refined $\lambda\mu_{\mathtt{s}}$-calculus is well-behaved w.r.t. the extended typing system $\mathcal{S}_{\lambda\mu_{\mathtt{s}}}$. This is done by means of forward (Section 8.1) and backward (Section 8.2) properties.

8.1. **Forward Properties.** Weighted Subject Reduction for the $\lambda\mu_{\mathtt{s}}$-calculus (Property 1) is based on two key properties, called respectively the **Linear Substitution** and the **Linear Replacement** Lemmas. These properties may simply be understood as a refinement of the Substitution Lemma 5.2 and the Replacement Lemma 5.3 to the case of the small-step $\lambda\mu_{\mathtt{s}}$-calculus. Their precise statements and proofs can be found in the Appendix (Lemmas 10.1 and 10.2).

**Property 1** (**Weighted Subject Reduction for $\lambda\mu_{\mathtt{s}}$**). *Let $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$. If $o \to o'$ is a non-erasing step, then $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ and $\mathtt{sz}(\Phi) > \mathtt{sz}(\Phi')$.*

*Proof.* By induction on the relation $\to$ using Lemma 4.5, Lemma 7.3 and the Linear Substitution and Replacement Lemmas mentioned above. See the Appendix for details. □

8.2. **Backward Properties.** As in the implicit case (Section 5.2), subject expansion for non-erasing $\lambda\mu_{\mathtt{s}}$-step relies on **(Linear) Reverse Substitution** and **(Linear) Reverse Replacement** Lemmas: if $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ and $o'$ has been obtained from $o$ by substituting one occurrence of $x$ by $u$ (or one subcommand $[\alpha]t$ by $[\alpha']tu$), then, informally speaking, it is possible to decompose $\Phi'$ into a regular derivation $\Phi_0$ typing $o$ and an auxiliary derivation $\Theta_u$ typing $u$. The precise statements and proofs can be found in the Appendix (Lemma 10.3 and 10.4).

**Property 2** (Subject Expansion for $\lambda\mu_{\mathtt{s}}$). *Let $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$. If $o \to_{\lambda\overline{\mu\mathtt{s}}} o'$ (i.e. a non-erasing $\lambda\mu_{\mathtt{s}}$-step), then $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$.*

*Proof.* By induction on $\to_{\lambda\overline{\mu\mathtt{s}}}$ using the Linear Reverse Lemmas mentioned above. See the Appendix for details. □

## 9. Strongly Normalizing $\lambda\mu_{\mathtt{s}}$-Objects

In this section we show a characterization of the set of strongly $\lambda\mu_{\mathtt{s}}$-normalizing terms by means of typability. The proof is done in several steps. The first key point is the characterization of the set of strongly $\lambda\overline{\mu\mathtt{s}}$-normalizing terms (instead of strongly normalizing $\lambda\mu_{\mathtt{s}}$-terms). For that, SR and SE lemmas for the type system are used. The second key point is the equivalence between strongly $\lambda\overline{\mu\mathtt{s}}$ and $\lambda\mu_{\mathtt{s}}$-normalizing terms. While the inclusion $\mathcal{SN}(\lambda\mu_{\mathtt{s}}) \subseteq \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ is straightforward, the fact that every $\mathtt{w}$-reduction step can be *postponed* w.r.t. any $\lambda\overline{\mu\mathtt{s}}$-step (Lemma 9.2) turns out to be crucial to show $\mathcal{SN}(\lambda\overline{\mu\mathtt{s}}) \subseteq \mathcal{SN}(\lambda\mu_{\mathtt{s}})$.

These technical tools are now used to prove that $\mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ coincides exactly with the set of typable terms. To close the picture, *i.e.* to show that also $\mathcal{SN}(\lambda\mu_{\mathtt{s}})$ coincides with the set of typable terms, we establish an equivalence between $\mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ and $\mathcal{SN}(\lambda\mu_{\mathtt{s}})$.

As defined in Section 2, for any $\lambda\mu_{\mathtt{s}}$-object $o$, we write now $\eta(o)$ for the length of the maximal reduction sequence starting at $o$. The following equations will play a key role in our proof of Theorem 9.4.

**Lemma 9.1.**

$$
\begin{array}{lcll}
\eta(x\,t_1\dots t_n) & = & +_{i=1\dots n}\eta(t_i) & \\
\eta(\lambda x.t) & = & \eta(t) & \\
\eta(\mu\alpha.c) & = & \eta(c) & \\
\eta([\alpha]t) & = & \eta(t) & \\
\eta((\lambda x.u)v\bar{t}) & = & \eta(u[x/v]\bar{t}) & \\
\eta((\mu\alpha.c)v\bar{t}) & = & \eta((\mu\alpha'.c\langle\alpha\mathbin{/\!\!/}\alpha'.v\rangle)\bar{t}) & \\
\eta(t[x/s]) & = & \eta(t)+\eta(s)+1 & \text{if } |t|_x = 0 \\
\eta(c\langle\alpha\mathbin{/\!\!/}\alpha'.s\rangle) & = & \eta(c)+\eta(s)+1 & \text{if } |c|_\alpha = 0 \\
\eta(\mathtt{TT})[\![x]\!][x/u] & = & \eta(\mathtt{TT}[\![u]\!]) & \text{if } |\mathtt{TT}[\![x]\!]|_x = 1 \\
\eta(\mathtt{CC}[\![[\alpha]t]\!]\langle\alpha\mathbin{/\!\!/}\alpha'.v\rangle) & = & \eta(\mathtt{CC}[\![[\alpha']tv]\!]) & \text{if } |\mathtt{CC}[\![[\alpha]t]\!]|_\alpha = 1 \\
\eta(\mathtt{TT}[\![x]\!][x/u]) & = & \eta(\mathtt{TT}[\![u]\!][x/u]) & \text{if } |\mathtt{TT}[\![x]\!]|_x > 1 \\
\eta(\mathtt{CC}[\![[\alpha]t]\!]\langle\alpha\mathbin{/\!\!/}\alpha'.v\rangle) & = & \eta(\mathtt{CC}[\![[\alpha']tv]\!]\langle\alpha\mathbin{/\!\!/}\alpha'.v\rangle) & \text{if } |\mathtt{CC}[\![[\alpha]t]\!]|_\alpha > 1 \\
\eta((tu)[x/s]) & = & \eta(t[x/s]u) & \text{if } |u|_x = 0
\end{array}
$$

In order to infer $\mathcal{SN}(\lambda\overline{\mu\mathtt{s}}) \subseteq \mathcal{SN}(\lambda\mu_{\mathtt{s}})$, the following postponement property is crucial.

**Lemma 9.2** (Postponement)**.** *Let $o \in \mathcal{O}_{\lambda\mu_{\mathtt{s}}}$. If $o{\to}^+_{\mathtt{w}}{\to}_{\lambda\overline{\mu\mathtt{s}}}o'$ then $o{\to}_{\lambda\overline{\mu\mathtt{s}}}{\to}^+_{\mathtt{w}}o'$.*

*Proof.* We first show by cases $o \to_{\mathtt{w}}{\to}_{\lambda\overline{\mu\mathtt{s}}} o'$ implies $o \to_{\lambda\overline{\mu\mathtt{s}}}{\to}^+_{\mathtt{w}} o'$. Then, the statement holds by induction on the number of $\mathtt{w}$-steps from $o$. □

**Lemma 9.3** (From $\lambda\overline{\mu\mathtt{s}}$ to $\lambda\mu_{\mathtt{s}}$)**.** *Let $o \in \mathcal{O}_{\lambda\mu_{\mathtt{s}}}$. If $o \in \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$, then $o \in \mathcal{SN}(\lambda\mu_{\mathtt{s}})$.*

*Proof.* We show that any reduction sequence $\rho : o \to_{\lambda\mu_{\mathtt{s}}} \dots$ is finite by induction on the pair $\langle o, n\rangle$, where $n$ is the maximal integer such that $\rho$ can be decomposed as $\rho : o \to^n_{\mathtt{w}} o' \to_{\lambda\overline{\mu\mathtt{s}}} o'' \to \dots$ (this is well-defined since $\to_{\mathtt{w}}$ is trivially terminating). We compare the pair $\langle o, n\rangle$ using $\to_{\lambda\overline{\mu\mathtt{s}}}$ for the first component (this is well-founded since $o \in \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ by hypothesis) and the standard order on natural numbers for the second one. When the reduction sequence starts with at least one $\mathtt{w}$-step we conclude by Lemma 9.2. All the other cases are straightforward. □

We conclude with the main theorem of this section:

**Theorem 9.4.** *Let $o \in \mathcal{O}_{\lambda\mu_{\mathtt{s}}}$. Then $o \in \mathcal{SN}(\lambda\mu_{\mathtt{s}})$ iff $o$ is typable.*

*Proof.* Let $\Phi \triangleright \Gamma \vdash o : \tau \mid \Delta$. Assume $o \notin \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ so that there exists an infinite sequence $o = o_0 \to_{\lambda\overline{\mu\mathtt{s}}} o_1 \to_{\lambda\overline{\mu\mathtt{s}}} o_2 \to_{\lambda\overline{\mu\mathtt{s}}} \cdots$. By Lemma 1 $\Phi_i \triangleright \Gamma \vdash o_i : \tau \mid \Delta$ for every $i$, and there exists an infinite sequence $\mathtt{sz}(\Phi_0) > \mathtt{sz}(\Phi_1) > \mathtt{sz}(\Phi_2) > \dots$, which leads to a contradiction because $\mathtt{sz}(\_)$ is a half-integer $\geqslant 1$. Therefore, $o \in \mathcal{SN}(\lambda\overline{\mu\mathtt{s}}) \subseteq_{\text{Lemma 9.3}} \mathcal{SN}(\lambda\mu_{\mathtt{s}})$.

For the converse, $o \in \mathcal{SN}(\lambda\mu_{\mathtt{s}}) \subseteq \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ because $\to_{\lambda\overline{\mu\mathtt{s}}} \subseteq \to_{\lambda\mu_{\mathtt{s}}}$. We then show that $o \in \mathcal{SN}(\lambda\overline{\mu\mathtt{s}})$ implies $o$ is typable. For that, we use the equalities in Lemma **??** to reason by induction on $\eta(t)$. The cases (1)-(6) and (13) are straightforward while the cases (7)-(12) use Lemma 2 (Partial Subject Expansion). □

It is worth noticing that the proof of Theorem 9.4 is self-contained: we do not use at all the previous characterization of strongly normalizing objects in the $\lambda\mu$-calculus that we have developed in Section 6. We remark however that an alternative proof of this theorem can be given in terms of the projection function defined in Section 7.2, an appropriate preservation of strong normalization-like property [27], and Theorem 6.4.

## 10. Conclusion

This paper provides non-idempotent type assignment systems $\mathcal{H}_{\lambda\mu}$ and $\mathcal{S}_{\lambda\mu}$ for the $\lambda\mu$-calculus, characterizing, respectively, head and strongly normalizing terms. These systems feature intersection and union types and can be used to get quantitative information of $\lambda\mu$-reduction sequences in the following sense:

- Whenever $o$ is typable in system $\mathcal{H}_{\lambda\mu}$, then its type derivation gives a measure providing an upper bound to the length of the head-reduction strategy starting at $o$.
- The same happens with system $\mathcal{S}_{\lambda\mu}$ with respect to the maximal length of a reduction sequence starting at $o$.
- Systems $\mathcal{H}_{\lambda\mu}$ and $\mathcal{S}_{\lambda\mu}$ have suggested the definition of the calculus $\lambda\mu_{\mathtt{s}}$, which implements a small-step operational semantics for classical natural deduction that is an extension of the *substitution at a distance paradigm* to the classical case.
- The calculus $\lambda\mu_{\mathtt{s}}$ was endowed with an extension of the typing system $\mathcal{S}_{\lambda\mu}$ presented for the $\lambda\mu$-calculus. The resulting system does not only characterize strong-normalization of small-step reduction but also gives quantitative information about it.

Following Chapter 3 of [35] (resp. [11]) in the framework of idempotent (resp. non-idempotent) intersection types for the $\lambda$-calculus, it is also possible to use system $\mathcal{H}_{\lambda\mu}$ to characterize *weak* normalization of $\lambda\mu$-terms. This can be done by considering a restricted class of judgments based on positive/negative occurrences of the empty type $[\,]$. This characterization also gives a certification of the fact that the leftmost-outermost strategy is complete for weak normalization in the $\lambda\mu$-calculus.

This work suggests many perspectives in the close future, including:

- Quantitative types are a powerful tool to provide *relational models* for $\lambda$-calculus [17, 3]. The construction of such models for $\lambda\mu$ should be investigated, particularly to understand in the classical case the collapse relation between quantitative and qualitative models [21].
- We expect to be able to transfer the ideas in this paper to a *classical sequent calculus* system, as was already done for focused intuitionistic logic [30]. In particular, the relational model proposed for the $\bar{\lambda}\mu$-calculus [51] could be useful for this purpose.
- The fact that idempotent types were already used to show *observational equivalence* between call-by-name and call-by-need [28] in intuitionistic logic suggests that our typing system $\mathcal{S}_{\lambda\mu_{\mathtt{s}}}$ could be used in the future to provide a type-theoretical view of the fact that classical call-by-name and classical call-by-need are *not* observationally equivalent [43].
- Moreover, as in [8], it should be possible to obtain *exact* bounds (and not only *upper* bounds) for the lengths of the head-reduction and the maximal reduction sequences. Although this result remains as future work, we remark that the difficult and conceptual part of the technique relies on a decreasing measure for $\lambda\mu$-reduction, which is precisely one of the contributions of this paper.
- The *inhabitation problem* for $\lambda$-calculus is known to be undecidable for idempotent intersection types [46], but decidable for the non-idempotent ones [10]. We may conjecture that inhabitation is also decidable for $\mathcal{H}_{\lambda\mu}$.

**Acknowledgment:** We would like to thank Vincent Guisse, who started a reflexion on quantitative types for the $\lambda\mu$-calculus during his M1 internship in Univ. Paris-Diderot.

## References


[1] B. Accattoli, E. Bonelli, D. Kesner, and C. Lombardi. A nonstandard standardization theorem. In P. Sewell, editor, *Proceedings of the 41st Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 659–670. ACM Press, 2014.

[2] B. Accattoli and D. Kesner. The structural lambda-calculus. In A. Dawar and H. Veith, editors, *Proceedings of 24th EACSL Conference on Computer Science Logic*, volume 6247 of *Lecture Notes in Computer Science*, pages 381–395. Springer-Verlag, Aug. 2010.

[3] S. Amini and T. Ehrhard. On Classical PCF, Linear Logic and the MIX Rule. In S. Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic (CSL 2015)*, volume 41 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 582–596. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

[4] Y. Andou. Church-Rosser property of a simple reduction for full first-order classical natural deduction. *Annals of Pure Applied Logic*, 119(1-3):225–237, 2003.

[5] Z. M. Ariola, H. Herbelin, and A. Saurin. Classical call-by-need and duality. In Ong [39], pages 27–44.

[6] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Bulletin of Symbolic Logic*, 48:931–940, 1983.

[7] P. Battyányi and K. Nour. An estimation for the lengths of reduction sequences of the $\lambda\mu\rho\theta$-calculus. *Logical Methods in Computer Science*, 14(2), 2018.

[8] A. Bernadet and S. Lengrand. Complexity of strongly normalising $\lambda$-terms via non-idempotent intersection types. In M. Hofmann, editor, *Foundations of Software Science and Computation Structures (FOSSACS)*, volume 6604 of *Lecture Notes in Computer Science*, pages 88–107. Springer-Verlag, 2011.

[9] A. Bernadet and S. Lengrand. Non-idempotent intersection types and strong normalisation. *Logical Methods in Computer Science*, 9(4), 2013.

[10] A. Bucciarelli, D. Kesner, and S. Ronchi Della Rocca. The inhabitation problem for non-idempotent intersection types. In Díaz et al. [19], pages 341–354.

[11] A. Bucciarelli, D. Kesner, and D. Ventura. Non-idempotent intersection types for the lambda-calculus. *Logic Journal of the IGPL*, 2017.

[12] M. Coppo and M. Dezani-Ciancaglini. A new type assignment for lambda-terms. *Archive for Mathematical Logic*, 19:139–156, 1978.

[13] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the $\lambda$-calculus. *Notre Dame Journal of Formal Logic*, 4:685–693, 1980.

[14] P. Curien and H. Herbelin. The duality of computation. In M. Odersky and P. Wadler, editors, *Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP '00), Montreal, Canada, September 18-21, 2000.*, pages 233–243. ACM, 2000.

[15] R. David and K. Nour. A short proof of the strong normalization of classical natural deduction with disjunction. *J. Symb. Log.*, 68(4):1277–1288, 2003.

[16] E. De Benedetti and S. Ronchi Della Rocca. Bounding normalization time through intersection types. In Graham-Lengrand and Paolini [25], pages 48–57.

[17] D. de Carvalho. *Sémantiques de la logique linéaire et temps de calcul*. These de doctorat, Université Aix-Marseille II, 2007.

[18] D. de Carvalho. Execution time of $\lambda$-terms via denotational semantics and intersection types. *Mathematical Structures in Computer Science*, 28(7):1169–1203, 2018.

[19] J. Díaz, I. Lanese, and D. Sangiorgi, editors. *Proceedings of the 8th International Conference on Theoretical Computer Science (TCS)*, volume 8705 of *Lecture Notes in Computer Science*. Springer-Verlag, 2014.

[20] D. J. Dougherty, S. Ghilezan, and P. Lescanne. Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: Extending the coppo-dezani heritage. *Theoretical Computer Science*, 398(1-3):114–128, 2008.

[21] T. Ehrhard. Collapsing non-idempotent intersection types. In P. Cégielski and A. Durand, editors, *Proceedings of 26th EACSL Conference on Computer Science Logic*, volume 16 of *LIPIcs*, pages 259–273. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[22] P. Gardner. Discovering needed reductions using type theory. In M. Hagiya and J. C. Mitchell, editors, *Theoretical Aspects of Computer Software, International Conference TACS '94, Sendai, Japan, April 19-22, 1994, Proceedings*, volume 789 of *Lecture Notes in Computer Science*, pages 555–574. Springer, 1994.

[23] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[24] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1990.
[25] S. Graham-Lengrand and L. Paolini, editors. *Proceedings of the Sixth Workshop on Intersection Types and Related Systems (ITRS), Dubrovnik, Croatia, 2012*, volume 121 of *Electronic Proceedings in Theoretical Computer Science*, 2013.
[26] T. Griffin. A formulae-as-types notion of control. In *17th Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 47–58. ACM Press, 1990.
[27] D. Kesner. A theory of explicit substitutions with safe and full composition. *Logical Methods in Computer Science*, 5(3:1):1–29, 2009.
[28] D. Kesner. Reasoning about call-by-need by means of types. In B. Jacobs and C. Löding, editors, *Foundations of Software Science and Computation Structures - 19th International Conference, FOSSACS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9634 of *Lecture Notes in Computer Science*, pages 424–441. Springer-Verlag, 2016.
[29] D. Kesner and D. Ventura. Quantitative types for the linear substitution calculus. In Díaz et al. [19], pages 296–310.
[30] D. Kesner and D. Ventura. A resource aware computational interpretation for Herbelin's syntax. In M. Leucker, C. Rueda, and F. D. Valencia, editors, *Theoretical Aspects of Computing - ICTAC 2015 - 12th International Colloquium Cali, Colombia, October 29-31, 2015, Proceedings*, volume 9399 of *Lecture Notes in Computer Science*, pages 388–403. Springer-Verlag, 2015.
[31] D. Kesner and P. Vial. Types as resources for classical natural deduction. In D. Miller, editor, *Proceedings of the 2nd International Conference on Formal Structures for Computation and Deduction, FSCD 2017*, volume 84 of *LIPIcs*, pages 24:1–24:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Sept. 2017.
[32] A. Kfoury. A linearization of the lambda-calculus and consequences. Technical report, Boston Universsity, 1996.
[33] A. Kfoury and J. Wells. Principality and type inference for intersection types using expansion variables. *Theoretical Computer Science*, 311(1-3):1–70, 2004.
[34] K. Kikuchi and T. Sakurai. A translation of intersection and union types for the $\lambda\mu$-calculus. In J. Garrigue, editor, *Programming Languages and Systems - 12th Asian Symposium, APLAS 2014, Singapore, November 17-19, 2014, Proceedings*, volume 8858 of *Lecture Notes in Computer Science*, pages 120–139. Springer-Verlag, 2014.
[35] J.-L. Krivine. *Lambda-calculus, types and models*. Ellis Horwood, 1993.
[36] O. Laurent. On the denotational semantics of the untyped lambda-mu calculus, 2004. Unpublished note.
[37] B. V. Mario Coppo, Mariangiola Dezani-Ciancaglini. Functional characters of solvable terms. *Mathematical Logic Quarterly*, 27:45–58, 1981.
[38] P. M. Neergaard and H. G. Mairson. Types, potency, and idempotency: why nonlinearity and amnesia make a type system work. In C. Okasaki and K. Fisher, editors, *Proceedings of the Ninth ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 138–149. ACM Press, 2004.
[39] L. Ong, editor. *Typed Lambda Calculi and Applications - 10th International Conference, TLCA 2011, Novi Sad, Serbia, June 1-3, 2011. Proceedings*, volume 6690 of *Lecture Notes in Computer Science*. Springer-Verlag, 2011.
[40] L. Ong and S. J. Ramsay. Verifying higher-order functional programs with pattern matching algebraic data type s. In T. Ball and M. Sagiv, editors, *Proceedings of the 38th Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 587–598. ACM Press, 2011.
[41] M. Pagani and S. Ronchi Della Rocca. Solvability in resource lambda-calculus. In L. Ong, editor, *Foundations of Software Science and Computation Structures*, volume 6014 of *Lecture Notes in Computer Science*, pages 358–373. Springer-Verlag, 2010.
[42] M. Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *International Conference on Logic Programming and Automated Reasoning*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer-Verlag, July 1992.
[43] P. Pédrot and A. Saurin. Classical by-need. In P. Thiemann, editor, *Programming Languages and Systems - 25th European Symposium on Programming, ESOP 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9632 of *Lecture Notes in Computer Science*, pages 616–643. Springer-Verlag, 2016.

[44] E. Polonovski. *Substitutions explicites, logique et normalisation.* Thèse de doctorat, Université Paris 7, 2004.
[45] P. Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(2):207–260, 2001.
[46] P. Urzyczyn. The emptiness problem for intersection types. *Journal of Symbolic Logic*, 64(3):1195–1215, 1999.
[47] S. van Bakel. Sound and complete typing for lambda-mu. In E. Pimentel, B. Venneri, and J. B. Wells, editors, *Proceedings Fifth Workshop on Intersection Types and Related Systems, ITRS 2010, Edinburgh, U.K., 9th July 2010.*, volume 45 of *EPTCS*, pages 31–44, 2010.
[48] S. van Bakel, F. Barbanera, and U. de'Liguoro. A filter model for the $\lambda\mu$-calculus - (extended abstract). In Ong [39], pages 213–228.
[49] S. van Bakel, F. Barbanera, and U. de'Liguoro. Characterisation of strongly normalising lambda-mu-terms. In Graham-Lengrand and Paolini [25], pages 1–17.
[50] F. van Raamsdonk, P. Severi, M. H. Sørensen, and H. Xi. Perpetual reductions in lambda-calculus. *Inf. Comput.*, 149(2):173–225, 1999.
[51] L. Vaux. Convolution lambda-bar-mu-calculus. In S. Ronchi Della Rocca, editor, *Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007, Paris, France, June 26-28, 2007, Proceedings*, volume 4583 of *Lecture Notes in Computer Science*, pages 381–395. Springer-Verlag, 2007.

## Appendix

**Lemma 5.2 (Substitution).** Let $\Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$. If $\Phi_o \rhd \Gamma; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta$, then there is $\Phi_{o\{x/u\}}$ such that

- $\Phi_{o\{x/u\}} \rhd \Gamma \wedge \Gamma_u \vdash o\{x/u\} : \mathcal{A} \mid \Delta \vee \Delta_u$.
- $\mathtt{sz}\left(\Phi_{o\{x/u\}}\right) = \mathtt{sz}\left(\Phi_o\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$.

*Proof.* We prove a more general statement, namely:

Let $\Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$.

- If $\Phi_o \rhd \Gamma_o; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta_o$, then there is $\Phi_{o\{x/u\}}$ such that
$$\Phi_{o\{x/u\}} \rhd \Gamma_o \wedge \Gamma_u \vdash o\{x/u\} : \mathcal{A} \mid \Delta_o \vee \Delta_u$$
- If $\Phi_o \rhd \Gamma_o; x : \mathcal{I} \Vdash t : \mathcal{J} \mid \Delta_o$, then there is $\Phi_{o\{x/u\}}$ such that
$$\Phi_{o\{x/u\}} \rhd \Gamma_o \wedge \Gamma_u \Vdash t\{x/u\} : \mathcal{J} \mid \Delta_o \vee \Delta_u$$

In both cases $\mathtt{sz}\left(\Phi_{o\{x/u\}}\right) = \mathtt{sz}\left(\Phi_o\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$.

We proceed by induction on the structure of $\Phi_o$.

- ($\mathtt{ax}$):
  - If $o = x$, then $\mathcal{I} = [\mathcal{U}]$ is a singleton, $\mathcal{A} = \mathcal{U}$, $\Gamma_o = \Delta_o = \emptyset$ and $o\{x/u\} = u$. The derivation $\Theta_u$ is necessarily of the following form
$$\frac{\Phi'_u \rhd \Gamma_u \vdash u : \mathcal{U} \mid \Delta_u}{\Gamma_u \Vdash u : [\mathcal{U}] \mid \Delta_u}\,(\wedge)$$
  We then set $\Phi_{x\{x/u\}} = \Phi'_u$. Then $\mathtt{sz}\left(\Phi_{x\{x/u\}}\right) = \mathtt{sz}\left(\Phi_x\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$, since $\mathtt{sz}\left(\Phi_x\right) = 1 = |\mathcal{I}|$ and $\mathtt{sz}\left(\Theta_u\right) = \mathtt{sz}\left(\Phi'_u\right)$.
  - If $o = y \neq x$, then $\mathcal{I} = [\,]$ and $o\{x/u\} = y$. Moreover, $\Theta_u$ is necessarily :
$$\frac{}{\emptyset \Vdash u : [\,] \mid \emptyset}\,(\wedge)$$
  We set $\Phi_{y\{x/u\}} = \Phi_y$. Then $\mathtt{sz}\left(\Phi_{y\{x/u\}}\right) = \mathtt{sz}\left(\Phi_y\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$ since $|\mathcal{I}| = 0$ and $\mathtt{sz}\left(\Theta_u\right) = 0$.
- ($\Rightarrow_{\mathtt{i}}$) : then $o = \lambda x.t$ and the derivation $\Phi_o$ has the following form

$$\frac{\Phi_t \rhd \Gamma_o; x : \mathcal{I}; y : \mathcal{J} \vdash t : \mathcal{U}_t \mid \Delta_o}{\Gamma_o; x : \mathcal{I} \vdash \lambda y.t : \langle \mathcal{J} \Rightarrow \mathcal{U}_t \rangle \mid \Delta_o} (\Rightarrow_{\mathtt{i}})$$

By the *i.h.* we have $\Phi_{t\{x/u\}} \rhd (\Gamma_o; y : \mathcal{J}) \wedge \Gamma_u \vdash t\{x/u\} : \mathcal{U} \mid \Delta_o \vee \Delta_u$ with $\mathtt{sz}\left(\Phi_{t\{x/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$. By $\alpha$-conversion $y \notin \mathtt{fv}(u)$ so that $y \notin \mathtt{dom}(\Gamma_u)$ by Lemma 4.6, thus $(\Gamma_o; y : \mathcal{J}) \wedge \Gamma_u = (\Gamma_o \wedge \Gamma_u); y : \mathcal{J}$. We then set $\Phi_{(\lambda y.t)\{x/u\}}$ equal to

$$\frac{\Phi_{t\{x/u\}}}{\Gamma_o \wedge \Gamma_u \vdash \lambda y.t\{x/u\} : \langle \mathcal{J} \Rightarrow \mathcal{U}_t \rangle \mid \Delta_o \vee \Delta_u} (\Rightarrow_{\mathtt{i}})$$

We have $\mathtt{sz}\left(\Phi_{(\lambda y.t)\{x/u\}}\right) = \mathtt{sz}\left(\Phi_{t\{x/u\}}\right) + 1 =_{i.h.} \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}| + 1 = \mathtt{sz}\left(\Phi\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}|$.

- $(\wedge)$: then $o$ is a term $t$ and $\Phi_o$ has the following form

$$\frac{(\Gamma_k; x : \mathcal{I}_k \vdash t : \mathcal{U}_k \mid \Delta_k)_{k \in K}}{\Gamma_o =; x : \mathcal{I} \Vdash t : [\mathcal{U}_k]_{k \in K} \mid \Delta_o} (\wedge)$$

where $\mathcal{I} = \wedge_{k \in K} \mathcal{I}_k$, $\Gamma_o = \wedge_{k \in K} \Gamma_k$ and $\Delta_o = \vee_{k \in K} \Delta_k$. By Lemma 5.1 there are auxiliary derivations $(\rhd \Gamma_u^k \Vdash u : \mathcal{I}_k \mid \Delta_u^k)_{k \in K}$ such that $\Gamma_u = \wedge_{k \in K} \Gamma_u^k$ and $\Delta_u = \vee_{k \in K} \Delta_u^k$. The *i.h.* gives derivations $(\rhd \Gamma_k \wedge \Gamma_u^k \vdash t\{x/u\} : \mathcal{U}_k \mid \Delta_k \wedge \Delta_u^k)_{k \in K}$ and we construct the following auxiliary derivation to conclude

$$\frac{(\Gamma_k \wedge \Gamma_u^k \vdash t\{x/u\} : \mathcal{U}_k \mid \Delta_k \wedge \Delta_u^k)_{k \in K}}{\wedge_{k \in K} \Gamma_k \wedge \Gamma_u^k \Vdash t\{x/u\} : [\mathcal{U}_k]_{k \in K} \mid \vee_{k \in K} \Delta_k \wedge \Delta_u^k} (\wedge)$$

We have $\wedge_{k \in K} \Gamma_k \wedge \Gamma_u^k = \Gamma_o \wedge \Gamma_u$ and $\vee_{k \in K} \Delta_k \wedge \Delta_u^k = \Delta_o \vee \Delta_u$ as desired. The size statement trivially holds by the *i.h.*

- $(\Rightarrow_{\mathtt{e}*})$: then $o = tv$ and the derivation $\Phi_o$ has the following form

$$\frac{\Phi_t \rhd \Gamma_t; x : \mathcal{I}_t \vdash t : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k \rangle_{k \in K} \mid \Delta_t \qquad \Phi_v \rhd \Gamma_v; x : \mathcal{I}_v \Vdash v : \wedge_{k \in K} \mathcal{I}_k^* \mid \Delta_v}{\Gamma_o; x : \mathcal{I} \vdash tv : \vee_{k \in K} \mathcal{V}_k \mid \Delta_o} (\Rightarrow_{\mathtt{e}*})$$

where $\Gamma_o = \Gamma_t \wedge \Gamma_v$, $\Delta_o = \Delta_t \vee \Delta_v$ and $\mathcal{I} = \mathcal{I}_t \wedge \mathcal{I}_v$.

Moreover, by Lemma 5.1 we can split $\Theta_u$ in $\Theta_u^t \rhd \Gamma_u^t \Vdash u : \mathcal{I}_t \mid \Delta_u^t$ and $\Theta_u^v \rhd \Gamma_u^v \Vdash u : \mathcal{I}_v \mid \Delta_u^v$ s.t. $\mathtt{sz}\left(\Theta_u\right) = \mathtt{sz}\left(\Theta_u^t\right) + \mathtt{sz}\left(\Theta_u^v\right)$.

By the *i.h.* there is $\Phi_{t\{x/u\}} \rhd \Gamma_t' \vdash t\{x/u\} : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k \rangle_{k \in K} \mid \Delta_t'$, where $\Gamma_t' = \Gamma_t \wedge \Gamma_u^t$ and $\Delta_t' = \Delta_t \vee \Delta_u^t$ and $\mathtt{sz}\left(\Phi_{t\{x/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}_t|$.

Also by the *i.h.* there is $\Phi_{v\{x/u\}} \rhd \Gamma_v' \Vdash v\{x/u\} : \wedge_{k \in K} \mathcal{I}_k^* \mid \Delta_v'$, where $\Gamma_v' = \Gamma_v \wedge \Gamma_u^v$ and $\Delta_v' = \Delta_v \vee \Delta_u^v$ and $\mathtt{sz}\left(\Phi_{v\{x/u\}}\right) = \mathtt{sz}\left(\Phi_v\right) + \mathtt{sz}\left(\Theta_u^v\right) - |\mathcal{I}_v|$.

We set then

$$\Phi_{o\{x/u\}} = \frac{\Phi_{t\{x/u\}} \qquad \Phi_{v\{x/u\}}}{\Gamma' \vdash (tv)\{x/u\} : \vee_{k \in K} \mathcal{V}_k \mid \Delta'} (\Rightarrow_{\mathtt{e}*})$$

where $\Gamma' = (\Gamma_t \wedge \Gamma_u^t) \wedge (\Gamma_v \wedge \Gamma_u^v) = \Gamma_o \wedge \Gamma_u$ and $\Delta' = (\Delta_t \vee \Delta_u^t) \vee (\Delta_v \vee \Delta_u^v) = \Delta_o \vee \Delta_u$ as desired. We conclude since

$$\begin{array}{l} \mathtt{sz}\left(\Phi_{o\{x/u\}}\right) = \mathtt{sz}\left(\Phi_{t\{x/u\}}\right) + \mathtt{sz}\left(\Phi^{v\{x/u\}}\right) + |K| \\ =_{i.h.} (\mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^t\right) - |\mathcal{I}_t|) + (\mathtt{sz}\left(\Phi_v\right) + \mathtt{sz}\left(\Theta_u^v\right) - |\mathcal{I}_v|) + |K| \\ = \mathtt{sz}\left(\Phi\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}| \end{array}$$

- All the other cases are straightforward.

□

**Lemma 5.3 (Replacement).** Let $\Theta_u \rhd \Gamma_u \Vdash u : \wedge_{k\in K}\ (\mathcal{I}_k^*) \mid \Delta_u$ where $\alpha \notin \mathtt{fn}(u)$. If $\Phi_o \rhd \Gamma_o \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta_o$, then there is $\Phi_{o\{\alpha/\!/u\}}$ such that :

- $\Phi_{o\{\alpha/\!/u\}} \rhd \Gamma_o \wedge \Gamma_u \vdash o\{\alpha/\!/u\} : \mathcal{A} \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta_o \vee \Delta_u$.
- $\mathtt{sz}\left(\Phi_{o\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_o\right) + \mathtt{sz}\left(\Theta_u\right)$.

*Proof.* We prove a more general statement, namely:
Let $\Theta_u \rhd \Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$ where $\alpha \notin \mathtt{fn}(u)$.

- If $\Phi_o \rhd \Gamma_o \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta_o$, then there is $\Phi_{o\{\alpha/\!/u\}}$ such that $\Phi_{o\{\alpha/\!/u\}} \rhd \Gamma_o \wedge \Gamma_u \vdash o\{\alpha/\!/u\} : \mathcal{A} \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta_o \vee \Delta_u$.
- If $\Phi_o \rhd \Gamma_o \Vdash t : \mathcal{J} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta_o$, then there is $\Phi_{o\{\alpha/\!/u\}}$ such that $\Phi_{o\{\alpha/\!/u\}} \rhd \Gamma_o \wedge \Gamma_u \Vdash t\{\alpha/\!/u\} : \mathcal{J} \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta_o \vee \Delta_u$

In both cases, $\mathtt{sz}\left(\Phi_{o\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_o\right) + \mathtt{sz}\left(\Theta_u\right)$.

We reason by induction on $\Phi_o$. Let us call $\mathcal{U}_\alpha = \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}$ and $\mathcal{U}'_\alpha = \vee_{k\in K}\mathcal{V}_k$.

- ($\mathtt{ax}$): $o = x$, thus we have by construction
$$\Phi_o \rhd \frac{}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\ (\mathtt{ax})$$
so that $K = \emptyset$. Thus, $\wedge_{k\in K}\mathcal{I}_k^* = [\,]$ and $\Gamma_u = \Delta_u = \emptyset$, then $\Theta_u$ is :
$$\frac{}{\emptyset \Vdash u : [\,] \mid \emptyset}\ (\wedge)$$
Thus $\mathtt{sz}\left(\Theta_u\right) = 0$.
We set $\Phi_{o\{\alpha/\!/u\}} = \Phi_o$ and the first result holds because the derivation has the desired form. We conclude since $\mathtt{sz}\left(\Phi_{o\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_o\right) + \mathtt{sz}\left(\Theta_u\right)$ as desired.
- ($\Rightarrow_{\mathtt{i}}$): then $o = \lambda x.t$, $o\{\alpha/\!/u\} = \lambda x.(t\{\alpha/\!/u\})$ and by construction we have
$$\Phi_{\lambda x.t} = \frac{\Phi_t \rhd x : \mathcal{I}; \Gamma_o \vdash t : \mathcal{U} \mid \alpha : \mathcal{U}_\alpha; \Delta_o}{\Gamma_o \vdash \lambda x.t : \langle \mathcal{I} \Rightarrow \mathcal{U}\rangle \mid \alpha : \mathcal{U}_\alpha; \Delta_o}\ (\Rightarrow_{\mathtt{i}})$$
By *i.h.* it follows that
$$\Phi_{t\{\alpha/\!/u\}} \rhd (x : \mathcal{I}; \Gamma_o) \wedge \Gamma_u \vdash t\{\alpha/\!/u\} : \mathcal{U} \mid \alpha : \mathcal{U}'_\alpha; \Delta_o \vee \Delta_u$$
with $\mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right)$. By $\alpha$-conversion we can assume that $x \notin \mathtt{fv}(u)$, thus by Lemma 4.6 $x \notin \mathtt{dom}(\Gamma_u)$, so that $(x : \mathcal{I}; \Gamma_o) \wedge \Gamma_u = x : \mathcal{I}; \Gamma_o \wedge \Gamma_u$.
We thus obtain $\Phi_{\lambda x.t\{\alpha/\!/u\}}$ of the form:
$$\frac{\Phi_{t\{\alpha/\!/u\}}}{\Gamma_o \wedge \Gamma_u \vdash \lambda x.t\{\alpha/\!/u\} : \langle \mathcal{I} \Rightarrow \mathcal{U}\rangle \mid \alpha : \mathcal{U}'_\alpha; \Delta_o \vee \Delta_u}\ (\Rightarrow_{\mathtt{i}})$$
We conclude since
$$\mathtt{sz}\left(\Phi_{\lambda x.t\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) + 1 =_{i.h.} \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) + 1 = \mathtt{sz}\left(\Phi_{\lambda x.t}\right) + \mathtt{sz}\left(\Theta_u\right)$$
- ($\Rightarrow_{\mathtt{e}*}$): then $o = tv$, $o\{\alpha/\!/u\} = t\{\alpha/\!/u\}v\{\alpha/\!/u\}$ and by construction we have $\Phi_o =$
$$\frac{\Phi_t \rhd \Gamma_t \vdash t : \mathcal{U}_t \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_t}; \Delta_t \qquad \Phi_v \rhd \Gamma_v \Vdash v : \mathcal{J}_v \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_v}; \Delta_v}{\Gamma_o \vdash o : \mathcal{U} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta_o}\ (\Rightarrow_{\mathtt{e}*})$$
where $\mathcal{U}_t = \langle \mathcal{J}_\ell \Rightarrow \mathcal{U}_\ell\rangle_{\ell\in L}$, $\mathcal{J}_v = \wedge_{\ell\in L}\mathcal{J}_\ell^*$, $\mathcal{U} = \vee_{\ell\in L}\mathcal{U}_\ell$ (those types are of no matter here, except they satisfy the typing constraint of $\Rightarrow_{\mathtt{e}*}$), $\Gamma_o = \Gamma_t \wedge \Gamma_v$, $\Delta_o = \Delta_t \vee \Delta_v$, $K = K_t \uplus K_v$.

Moreover, by Lemma 5.1, we can split $\Theta_u$ in $\Theta_u^t \triangleright \Gamma_u^t \Vdash u : \wedge_{k\in K_t}\mathcal{I}_k^* \mid \Delta_u^t$ and $\Theta_u^v \triangleright \Gamma_u^v \Vdash u : \wedge_{k\in K_v}\mathcal{I}_k^* \mid \Delta_u^v$ s.t. $\mathtt{sz}(\Theta_u) = \mathtt{sz}\left(\Theta_u^t\right) + \mathtt{sz}\left(\Theta_u^v\right)$.

By *i.h.* we have $\Phi_{t\{\alpha/\!/u\}} \triangleright \Gamma_t \wedge \Gamma_u^t \vdash t\{\alpha/\!/u\} : \mathcal{U}_t \mid \alpha : \vee_{k\in K_t}\mathcal{V}_k; \Delta_t \vee \Delta_u^t$ (since $\alpha \notin \mathtt{fn}(u)$) with $\mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^t\right)$.

Also by *i.h.* we have $\Phi_{v\{\alpha/\!/u\}} \triangleright \Gamma_v \wedge \Gamma_u^v \Vdash v\{\alpha/\!/u\} : \mathcal{J}_v \mid \alpha : \vee_{k\in K_v}\mathcal{V}_k, \Delta_v \vee \Delta_u^v$ with $\mathtt{sz}\left(\Phi_{v\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_v\right) + \mathtt{sz}\left(\Theta_u^v\right)$.

We can now construct the following derivation

$$\frac{\Phi_{t\{\alpha/\!/u\}} \qquad \Phi_{v\{\alpha/\!/u\}}}{\Gamma' \vdash o\{\alpha/\!/u\} : \mathcal{U} \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta'}\,(\Rightarrow_{\mathtt{e}*})$$

where $\Gamma' = (\Gamma_t \wedge \Gamma_u^t) \wedge (\Gamma_v \wedge \Gamma_u^v) = (\Gamma_t \wedge \Gamma_v) \wedge (\Gamma_u^t \wedge \Gamma_u^v) = \Gamma_o \wedge \Gamma_u$ and likewise, $\Delta' = \Delta_o \vee \Delta_u$ as desired. Moreover,

$$\begin{array}{l}\mathtt{sz}\left(\Phi_{(tv)\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) + \mathtt{sz}\left(\Phi_{v\{\alpha/\!/u\}}\right) + |L| \\ =_{i.h.} \left(\mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^t\right)\right) + \left(\mathtt{sz}\left(\Phi_v\right) + \mathtt{sz}\left(\Theta_u^v\right)\right) + |L| \\ = \left(\mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Phi_v\right) + |L|\right) + \left(\mathtt{sz}\left(\Theta_u^t\right) + \mathtt{sz}\left(\Theta_u^v\right)\right) = \mathtt{sz}\left(\Phi_{tv}\right) + \mathtt{sz}\left(\Theta_u\right)\end{array}$$

- If $o = [\alpha]t$, then $o\{\alpha/\!/u\} = [\alpha]t\{\alpha/\!/u\}u$ and by construction we have a derivation $\Phi_{[\alpha]t}$ of the form:

$$\frac{\Phi_t \triangleright \Gamma_o \vdash t : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_t} \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_\alpha}; \Delta_o}{\Gamma_o \vdash [\alpha]t : \# \mid \alpha : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K}; \Delta_o}\,(\#_{\mathtt{i}})$$

where $K = K_t \uplus K_\alpha$.

Moreover, by Lemma 5.1, we can split $\Theta_u$ in $\Theta_u^t \triangleright \Gamma_u^t \Vdash u : \wedge_{k\in K_t}\mathcal{I}_k^* \mid \Delta_u^t$ and $\Theta_u^\alpha \triangleright \Gamma_u^\alpha \Vdash u : \wedge_{k\in K_\alpha}\mathcal{I}_k^* \mid \Delta_u^\alpha$ s.t. $\mathtt{sz}(\Theta_u) = \mathtt{sz}\left(\Theta_u^t\right) + \mathtt{sz}\left(\Theta_u^\alpha\right)$.

By the *i.h.* we have $\Phi_{t\{\alpha/\!/u\}} \triangleright \Gamma_o \wedge \Gamma_u^\alpha \vdash t\{\alpha/\!/u\} : \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_t} \mid \alpha : \vee_{k\in K_\alpha}\mathcal{V}_k; \Delta_o \vee \Delta_u^\alpha$ with $\mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^\alpha\right)$.

We can then construct the following derivation $\Phi_{[\alpha]t\{\alpha/\!/u\}u}$:

$$\dfrac{\dfrac{\Phi_{t\{\alpha/\!/u\}} \qquad \Theta_u^t}{\Gamma' \vdash t\{\alpha/\!/u\}u : \vee_{k\in K_t}\mathcal{V}_k \mid \alpha : \vee_{k\in K_\alpha}\mathcal{V}_k; \Delta'}\,(\Rightarrow_{\mathtt{e}*})}{\Gamma' \vdash [\alpha]t\{\alpha/\!/u\}u : \# \mid \alpha : \vee_{k\in K}\mathcal{V}_k; \Delta'}\,(\#_{\mathtt{i}})$$

with $\Gamma' = \Gamma_o \wedge \Gamma_u^\alpha \wedge \Gamma_u^t = \Gamma_o \wedge \Gamma_u$ and likewise $\Delta' = \Delta_o \vee \Delta_u$ (since $\alpha \notin \mathtt{fn}(u)$) as expected. We conclude since

$$\begin{array}{l}\mathtt{sz}\left(\Phi_{[\alpha]t\{\alpha/\!/u\}u}\right) = \mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}u}\right) + \mathtt{ar}(\vee_{k\in K_t}\mathcal{V}_k) \\ = \mathtt{sz}\left(\Phi_{t\{\alpha/\!/u\}}\right) + \mathtt{sz}\left(\Theta_u^t\right) + |K_t| + \mathtt{ar}(\vee_{k\in K_t}\mathcal{V}_k) \\ =_{i.h.} \left(\mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^\alpha\right)\right) + \mathtt{sz}\left(\Theta_u^t\right) + \mathtt{ar}(\langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_t}) \\ = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) + \mathtt{ar}(\langle \mathcal{I}_k \Rightarrow \mathcal{V}_k\rangle_{k\in K_t}) = \mathtt{sz}\left(\Phi_{[\alpha]t}\right) + \mathtt{sz}\left(\Theta_u\right)\end{array}$$

- All the other cases are straightforward.

□

**Property 5.4 (Weighted Subject Reduction for $\mathcal{S}_{\lambda\mu}$).** Let $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$. If $o \to o'$ is a non-erasing step, then there exists a derivation $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ such that $\mathtt{sz}(\Phi) > \mathtt{sz}(\Phi')$.

*Proof.* By induction on the relation $\to$. We only show the main cases of reduction at the root, the other ones being straightforward.

- If $o = (\lambda x.t)u$, then $o' = t\{x/u\}$ and $x \in \mathtt{fv}(t)$. The derivation $\Phi$ has the following form:

$$\dfrac{\dfrac{\Phi_t \rhd \Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t}{\Gamma_t \vdash \lambda x.t : \langle \mathcal{I} \Rightarrow \mathcal{U} \rangle \mid \Delta_t}\,(\Rightarrow_{\mathtt{i}}) \qquad \Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma \vdash o : \mathcal{U} \mid \Delta}\,(\Rightarrow_{\mathtt{e}*})$$

  where $\Gamma = \Gamma_t \wedge \Gamma_u$, $\Delta = \Delta_t \vee \Delta_u$. Indeed, $x \in \mathtt{fv}(t)$ implies by Lemma 4.6 that $\mathcal{I} \neq [\,]$ so that $\mathcal{I}^* = \mathcal{I} = [\mathcal{U}_k]_{k \in K}$ for some $K \neq \emptyset$ and some $(\mathcal{U}_k)_{k \in K}$.

  Lemma 5.2 yields a derivation $\Phi'_{t\{x/u\}} \rhd \Gamma_t \wedge \Gamma_u \vdash t\{x/u\} : \mathcal{U} \mid \Delta_t \vee \Delta_u$ with $\mathtt{sz}\left(\Phi'_{t\{x/u\}}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u\right) - |K|$ ($|\mathcal{I}| = |K|$). We set $\Phi' = \Phi'_{t\{x/u\}}$ so that $\mathtt{sz}\left(\Phi\right) = \mathtt{sz}\left(\Phi_t\right) + 1 + \mathtt{sz}\left(\Theta_u\right) + 1 > \mathtt{sz}\left(\Phi'\right)$.

- If $o = (\mu\alpha.c)u$, then $o' = \mu\alpha.c\{\alpha /\!\!/ u\}$ and $\alpha \in \mathtt{fn}(c)$.

  The derivation $\Phi$ has the following form:

$$\dfrac{\dfrac{\Phi_c \rhd \Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_c; \Delta_c}{\Gamma_c \vdash \mu\alpha.c : \mathcal{V}_c^* \mid \Delta_c}\,(\#_{\mathtt{e}}) \qquad \Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I}_u^* \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash (\mu\alpha.c)u : \mathcal{U} \mid \Delta_c \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}*})$$

  where $\mathcal{V}_c^* = \mathcal{V}_c = \langle \mathcal{I}_k \Rightarrow \mathcal{V}_k \rangle_{k \in K}$, $\mathcal{I}_u^* = \mathcal{I}_u = \wedge_{k \in K} \mathcal{I}_k^*$, $\mathcal{U} = \vee_{k \in K} \mathcal{V}_k$, $\Gamma = \Gamma_c \wedge \Gamma_u$ and $\Delta = \Delta_c \vee \Delta_u$. Indeed, the hypothesis $\alpha \in \mathtt{fn}(c)$ implies $K \neq \emptyset$ by Lemma 4.6, and thus $\mathcal{V}_c^* = \mathcal{V}_c$ and $\mathcal{I}_u^* = \mathcal{I}_u$. Lemma 5.3 then gives the derivation $\Phi_{c\{\alpha /\!\!/ u\}} \rhd \Gamma_c \wedge \Gamma_u \vdash c\{\alpha /\!\!/ u\} : \# \mid \alpha : \vee_{k \in K} \mathcal{V}_k; \Delta_c \vee \Delta_u$. We can then construct the following derivation $\Phi'$:

$$\dfrac{\Phi_{c\{\alpha /\!\!/ u\}}}{\Gamma_c \wedge \Gamma_u \vdash \mu\alpha.c\{\alpha /\!\!/ u\} : \vee_{k \in K} \mathcal{V}_k \mid \Delta_c \vee \Delta_u}\,(\#_{\mathtt{e}})$$

  We conclude since

$$\mathtt{sz}\left(\Phi'\right) = \mathtt{sz}\left(\phi_{c\{\alpha /\!\!/ u\}}\right) + 1 =_{Lemma 5.3} \mathtt{sz}\left(\Phi_c\right) + \mathtt{sz}\left(\Theta_u\right) + 1 <$$
$$\mathtt{sz}\left(\Phi_c\right) + 1 + \mathtt{sz}\left(\Theta_u\right) + |K| = \mathtt{sz}\left(\Phi_{\mu\alpha.c}\right) + \mathtt{sz}\left(\Theta_u\right) + |K| = \mathtt{sz}\left(\Phi\right)$$

  The step $<$ is justified by $K \neq \emptyset$.

□

The reader should notice that the fact that the choice operator produces a *blind type* for union types is not used in the proof of Property 5.4. Indeed, by Lemma 4.6, the variable (resp. name) of a $\beta$-redex (resp. $\mu$-redex) has an empty intersection (resp. union) type in system $\mathcal{S}_{\lambda\mu}$ only when this redex is erasing, a case that is not in the scope of Property 5.4. However, note that blind types are involved in the proof of the subject reduction property in system $\mathcal{H}_{\lambda\mu}$, which can easily be adapted from that of Property 5.4.

**Lemma 5.5 (Reverse Substitution).** Let $\Phi' \rhd \Gamma' \vdash o\{x/u\} : \mathcal{A} \mid \Delta'$ Then there exist $\Gamma_o, \Delta_o, \mathcal{I}, \Gamma_u, \Delta_u$ such that:

- $\Gamma' = \Gamma \wedge \Gamma_u$,
- $\Delta' = \Delta \vee \Delta_u$,
- $\rhd\Gamma; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta$
- $\rhd\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$.

*Proof.* We prove a more general statement, namely:

- If $\Phi' \triangleright \Gamma' \vdash o\{x/u\} : \mathcal{A} \mid \Delta'$, then $\triangleright\Gamma_o; x : \mathcal{I} \vdash o : \mathcal{A} \mid \Delta_o$, $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$, where $\Gamma' = \Gamma_o \wedge \Gamma_u,\ \Delta' = \Delta_o \vee \Delta_u$ for some $\mathcal{I},\ \Gamma_o,\ \Gamma_u,\ \Delta_o,\ \Delta_u$.
- If $\Phi' \triangleright \Gamma' \Vdash t\{x/u\} : \mathcal{J} \mid \Delta'$, then $\triangleright\Gamma_o; x : \mathcal{I} \Vdash t : \mathcal{J} \mid \Delta_o$, $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$, where $\Gamma' = \Gamma_o \wedge \Gamma_u,\ \Delta' = \Delta_o \vee \Delta_u$ for some $\mathcal{I},\ \Gamma_o,\ \Gamma_u,\ \Delta_o,\ \Delta_u$.

We proceed by induction on the structure of $\Phi'$.

- ($\mathtt{ax}$)
  - If $o = y \neq x$, then $y\{x/u\} = y$. By construction one has that $\Gamma' = y : [\mathcal{U}]$ and $\mathcal{A} = \mathcal{U}$. The result thus holds for $\mathcal{I} = [\,],\ \Gamma_o = \Gamma',\ \Delta_o = \Delta',\ \Gamma_u = \emptyset$ and $\Delta_u = \emptyset$ as $\emptyset \Vdash u : [\,] \mid \emptyset$ is derivable by the ($\wedge$) rule.
  - If $o = x$, then $x\{x/u\} = u$. By construction one has that $\mathcal{A} = \mathcal{U}$. We type $x$ with the axiom rule:
$$\frac{}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\ (\mathtt{ax})$$
  so that the property holds for $\Gamma_o = \Delta_o = \emptyset,\ \mathcal{I} = [\mathcal{U}],\ \Gamma_u = \Gamma',\ \Delta_u = \Delta'$, where $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$ is obtained by the rule ($\wedge$) from $\Gamma' \vdash u : \mathcal{U} \mid \Delta'$.
- ($\Rightarrow_{\mathtt{i}}$) $o = \lambda y.t$ and $(\lambda y.t)\{x/u\} = \lambda y.t\{x/u\}$. Then $\Phi'$ is of the form
$$\frac{\Phi'_t \triangleright \Gamma'; y : \mathcal{J} \vdash t\{x/u\} : \mathcal{V} \mid \Delta'}{\Gamma' \vdash \lambda y.t\{x/u\} : \langle \mathcal{J} \Rightarrow \mathcal{V} \rangle \mid \Delta'}\ (\Rightarrow_{\mathtt{i}})$$
  where $\mathcal{U} = \langle \mathcal{J} \Rightarrow \mathcal{V} \rangle$.

  By the *i.h.* $\Gamma'; y : \mathcal{I} = \Gamma_t \wedge \Gamma_u$ and $\Delta' = \Delta_t \vee \Delta_u$, $\triangleright\Gamma_t; x : \mathcal{I} \vdash t : \mathcal{V} \mid \Delta_t$ and $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$. By $\alpha$-conversion we can assume that $y \notin \mathtt{fv}(u)$, so that $y \notin \mathtt{dom}(\Gamma_u)$ by Lemma 4.6 and thus $\Gamma_t = \Gamma'_t; y : \mathcal{J}$ and $\Gamma' = \Gamma'_t \wedge \Gamma_u$. Hence, we obtain $\Gamma'_t; x : \mathcal{I} \vdash \lambda y.t : \mathcal{U} \mid \Delta_t$ by the rule ($\Rightarrow_{\mathtt{i}}$). We conclude by setting $\Gamma_o = Gam'_t$ and $\Delta_o = \Delta_t$.
- ($\Rightarrow_{\mathtt{e}*}$) $o = tv$ and $(tv)\{x/u\} = t\{x/u\}v\{x/u\}$. By construction we have that $\Gamma' = \Gamma'_t \wedge \Gamma'_v$ and $\Delta' = \Delta'_t \vee \Delta'_v$ and $\triangleright\Gamma'_t \vdash t\{x/u\} : \mathcal{U}_t \mid \Delta'_t$, $\triangleright\Gamma'_v \Vdash v : \mathcal{J}_v \mid \Delta'_v$ with $\mathcal{U}_t = \langle \mathcal{J}_k \Rightarrow \mathcal{U}_k \rangle_{k \in K},\ \mathcal{J}_v = \wedge_{k \in K} \mathcal{J}_k^*$ (those types are of no matter here, except they satisfy the typing constraint of ($\Rightarrow_{\mathtt{e}*}$)). By the *i.h.* there are:
  - $\Gamma_t,\ \mathcal{I}_t,\ \Delta_t,\ \Gamma_u^t,\ \Delta_u^t$ s.t. $\Gamma'_t = \Gamma_t \wedge \Gamma_u^t,\ \Delta'_t = \Delta_t \vee \Delta_u^t,\ \triangleright\Gamma_t; x : \mathcal{I}_t \vdash t : \mathcal{U}_t \mid \Delta_t$ and $\triangleright\Gamma_u^t \Vdash u : \mathcal{I}_t \mid \Delta_u^t$.
  - $\Gamma_v,\ \mathcal{I}_v,\ \Delta_v,\ \Gamma_u^v,\ \Delta_u^v$ s.t. $\Gamma'_v = \Gamma_v \wedge \Gamma_u^v,\ \Delta'_v = \Delta_v \vee \Delta_u^v,\ \triangleright\Gamma_v; x : \mathcal{I}_v \Vdash v : \mathcal{J}_v \mid \Delta_v$ and $\triangleright\Gamma_u^v \Vdash u : \mathcal{I}_v \mid \Delta_u^v$.

  Thus, we can type $tv$ with :
$$\frac{\triangleright\Gamma_t; x : \mathcal{I}_t \vdash t : \mathcal{U}_t \mid \Delta_t \triangleright \Gamma_v; x : \mathcal{I}_v \Vdash v : \mathcal{J}_v \mid \Delta_v}{\Gamma_o; x : \mathcal{I} \vdash tv : \mathcal{U} \mid \Delta_o}\ (\Rightarrow_{\mathtt{e}*})$$
  where $\Gamma_o = \Gamma_t \wedge \Gamma_u,\ \Delta_o = \Delta_t \vee \Delta_u,\ \mathcal{I} = \mathcal{I}_t \vee \mathcal{I}_v$.

  We obtain $\triangleright\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$ with $\Gamma_u = \Gamma_u^t \wedge \Gamma_u^v,\ \Delta_u = \Delta_u^t \vee \Delta_u^v$ by Lemma 5.1.
- The other cases are similar.

□

**Lemma 5.6 (Reverse Replacement).** Let $\Phi' \triangleright \Gamma' \vdash o\{\alpha /\!\!/ u\} : \mathcal{A} \mid \alpha : \mathcal{V}; \Delta'$, where $\alpha \notin \mathtt{fn}(u)$. Then there exist $\Gamma_o, \Delta_o, \Gamma_u, \Delta_u, (\mathcal{I}_k)_{k \in K}, (\mathcal{V}_k)_{k \in K}$ such that:

- $\Gamma' = \Gamma \wedge \Gamma_u$,
- $\Delta' = \Delta \vee \Delta_u$,

- $\mathcal{V} = \vee_{k\in K}\mathcal{V}_k$,
- $\triangleright\Gamma \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta$, and
- $\triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$

*Proof.* We prove a more general statement, namely :

- If $\Phi' \rhd \Gamma' \vdash o\{\alpha /\!\!/ u\} : \mathcal{A} \mid \alpha : \mathcal{V}; \Delta'$, then $\triangleright\Gamma_o \vdash o : \mathcal{A} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_o,\ \triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$ where $\Gamma' = \Gamma_o \wedge \Gamma_u,\ \Delta' = \Delta_o \vee \Delta_u,\ \mathcal{V} = \vee_{k\in K}\mathcal{V}_k$ for some $\Gamma_o,\ \Gamma_u,\ \Delta_o,\ \Delta_u, (\mathcal{V}_k)_{k\in K},\ (\mathcal{I}_k)_{k\in K}$.
- If $\Phi' \rhd \Gamma' \Vdash t\{\alpha /\!\!/ u\} : \mathcal{J} \mid \alpha : \mathcal{V}; \Delta'$, then $\triangleright\Gamma_o \Vdash t : \mathcal{J} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_o,\ \triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$ where $\Gamma' = \Gamma_o \wedge \Gamma_u,\ \Delta' = \Delta_o \vee \Delta_u,\ \mathcal{V} = \vee_{k\in K}\mathcal{V}_k$ for some $\Gamma_o,\ \Gamma_u,\ \Delta_o,\ \Delta_u, (\mathcal{V}_k)_{k\in K},\ (\mathcal{I}_k)_{k\in K}$.

We proceed by induction on the structure of $\Phi$'.

- ($\mathtt{ax}$) $o = x$ and $o\{\alpha /\!\!/ u\} = x$. Then $\Phi'$ is of the form $x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset$ and we have $\mathcal{V} = \langle\,\rangle$ so that we set $\Gamma_o = x : [\mathcal{U}]$, $\Gamma_u = \Delta_u = \Delta_o = \emptyset$, $K = \emptyset$. Notice that $\emptyset \Vdash u : [\,] \mid \emptyset$ always holds.

- ($\Rightarrow_{\mathtt{i}}$) $o = \lambda y.t$ and $(\lambda y.t)\{\alpha /\!\!/ u\} = \lambda y.t\{\alpha /\!\!/ u\}$. Then $\Phi'$ is of the form
$$\frac{\Gamma'; y : \mathcal{J} \vdash t\{\alpha /\!\!/ u\} : \mathcal{U}_t \mid \alpha : \mathcal{V}; \Delta'}{\Gamma' \vdash \lambda y.t\{\alpha /\!\!/ u\} : \langle \mathcal{I} \Rightarrow \mathcal{U}_t\rangle \mid \alpha : \mathcal{V}; \Delta'}\,(\Rightarrow_{\mathtt{i}})$$
The *i.h.* gives $\Gamma'; y : \mathcal{J} = \Gamma_t \wedge \Gamma_u,\ \mathcal{V} = \vee_{k\in K}\mathcal{V}_k,\ \Delta' = \Delta_t \vee \Delta_u,\ \triangleright\Gamma_t \vdash t : \mathcal{U}_t \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_t$ and $\triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$. By $\alpha$-conversion we can assume that $y \notin \mathtt{fv}(u)$, so that $y \notin \mathtt{dom}(\Gamma_u)$ holds by Lemma 4.6 and thus $\Gamma_t = \Gamma_t'; y : \mathcal{J}$. Hence, we obtain
$$\frac{\triangleright\Gamma_t'; y : \mathcal{J} \vdash t : \mathcal{U}_t \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_t}{\triangleright\Gamma_t' \vdash \lambda y.t : \langle \mathcal{J} \Rightarrow \mathcal{U}_t\rangle \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_t}$$
From that, the desired conclusion is straightforward by setting $\Gamma_o = \Gamma_t'$ and $\Delta_o = \Delta_t$.
- $o = [\alpha]t$ and $o\{\alpha /\!\!/ u\} = [\alpha]t\{\alpha /\!\!/ u\}u$. Then $\Phi'$ has the following form
$$\frac{\dfrac{\Gamma_t' \vdash t\{\alpha /\!\!/ u\} : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K_t} \mid \alpha : \mathcal{V}_\alpha; \Delta_t' \qquad \Gamma_u^t \Vdash u : \wedge_{k\in K_t}\mathcal{I}_k^* \mid \alpha : \mathcal{V}_u; \Delta_u^t}{\Gamma_t' \wedge \Gamma_u^t \vdash t\{\alpha /\!\!/ u\}u : \vee_{k\in K_t}\mathcal{V}_k \mid \alpha : \mathcal{V}_\alpha; \Delta'}\,(\Rightarrow_{\mathtt{e}*})}{\Gamma' \vdash [\alpha]t\{\alpha /\!\!/ u\}u : \# \mid \alpha : \vee_{k\in K_t}\mathcal{V}_k \vee \mathcal{V}_\alpha; \Delta'}\,(\#_{\mathtt{e}})$$
where $\Gamma' = \Gamma_t' \wedge \Gamma_u^t,\ \Delta' = \Delta_t \vee \Delta_u^t$ and $\mathcal{V} = \vee_{k\in K_t}\mathcal{V}_k \vee \mathcal{V}_\alpha \vee \mathcal{V}_u$. Moreover, the hypothesis $\alpha \notin \mathtt{fn}(u)$ implies $\mathcal{V}_u = \langle\,\rangle$ by Lemma 4.6.

The *i.h.* gives $\triangleright\Gamma_t \vdash t : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K_t} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K_\alpha}; \Delta_t,\ \triangleright\Gamma_u^\alpha \Vdash u : \wedge_{k\in K_\alpha}\mathcal{I}_k^* \mid \Delta_u^\alpha$ where $\Gamma_t' = \Gamma_t \wedge \Gamma_u^\alpha,\ \Delta_t' = \Delta_t \vee \Delta_u^\alpha$, and $\mathcal{V}_\alpha = \vee_{k\in K_\alpha}\mathcal{V}_k$. W.l.o.g we can assume $K_\alpha \cap K_t = \emptyset$. We then set $K = K_\alpha \uplus K_t$ and we define :
$$\frac{\triangleright\Gamma_t \vdash t : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K_t} \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K_\alpha}; \Delta_t}{\Gamma_t \vdash [\alpha]t : \# \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K}; \Delta_t}\,(\#_{\mathtt{e}})$$
By Lemma 5.1, we also have $\triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$ with $\Gamma_u = \Gamma_u^t \wedge \Gamma_u^\alpha,\ \Delta_u = \Delta_u^t \vee \Delta_u^\alpha$. We can then conclude by setting $\Gamma_o = \Gamma_t$ and $\Delta_o = \Delta_t$ since $\Gamma_t \wedge \Gamma_u = \Gamma_t \wedge (\Gamma_u^\alpha \wedge \Gamma_u^t) = \Gamma_t' \wedge \Gamma_u^t = \Gamma'$ and likewise $\Delta_t \vee \Delta_u = \Delta'$.

- $o = tv$ so that $o\{\alpha /\!\!/ u\} = t\{\alpha /\!\!/ u\}v\{\alpha /\!\!/ u\}$. Then $\Phi$ has the following form:
$$\frac{\triangleright\Gamma_t' \vdash t\{\alpha /\!\!/ u\} : \mathcal{U}_t \mid \alpha : \mathcal{V}_t; \Delta_t' \qquad \triangleright\Gamma_v' \Vdash v\{\alpha /\!\!/ u\} : \mathcal{J}_v \mid \alpha : \mathcal{V}_v; \Delta_v'}{\Gamma' \vdash t\{\alpha /\!\!/ u\}v\{\alpha /\!\!/ u\} : \mathcal{U} \mid \alpha : \mathcal{V}; \Delta'}\,(\Rightarrow_{\mathtt{e}*})$$

where $\mathcal{V} = \mathcal{V}_t \vee \mathcal{V}_\alpha$, $\Gamma' = \Gamma'_t \wedge \Gamma'_v$, $\Delta' = \Delta'_t \vee \Delta'_v$, $\mathcal{U}_t = \langle \mathcal{J}_k \Rightarrow \mathcal{U}_k \rangle_{k\in K}$ and $\mathcal{J}_v = \wedge_{k\in K} \mathcal{J}_k^*$ (those types are of no matter here, except they satisfy the typing constraint of $(\Rightarrow_{\mathtt{e}*})$).

The property then trivially holds by the *i.h.* (we proceed as in the complete proof of Lemma 5.5, case $(\Rightarrow_{\mathtt{e}*})$).

- The other cases are similar.

□

**Property 5.7 (Subject Expansion for $\mathcal{S}_{\lambda\mu}$).** Assume $\Phi' \rhd \Gamma' \vdash o' : \mathcal{A} \mid \Delta'$. If $o \to o'$ is a non-erasing step, then there is $\Phi \rhd \Gamma' \vdash o : \mathcal{A} \mid \Delta'$.

*Proof.* By induction on the reduction relation. We only show the main cases of reduction at the root, the other ones being straightforward by induction. We can then assume $\mathcal{A} = \mathcal{U}$ for some union type $\mathcal{U}$.

- If $o = (\lambda x.t)u$, then $o' = t\{x/u\}$ with $x \in \mathtt{fv}(t)$. The Reverse Substitution Lemma 5.5 yields
  - $\Gamma' = \Gamma_o \wedge \Gamma_u$,
  - $\Delta' = \Delta_o \wedge \Delta_u$,
  - $\rhd\Gamma_o; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_o$, and
  - $\rhd\Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$.

  Moreover, $x \in \mathtt{fv}(t)$ implies by Lemma 4.6 that $\mathcal{I} \neq [\,]$, so that $\mathcal{I}^* = \mathcal{I}$. We can then set :
$$\Phi = \dfrac{\dfrac{\rhd\Gamma_o; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_o}{\Gamma_o \vdash \lambda x.t : \langle \mathcal{I} \Rightarrow \mathcal{U} \rangle \mid \Delta_o}\,(\Rightarrow_{\mathtt{i}}) \qquad \rhd \Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u}{\Gamma' \vdash (\lambda x.t)u : \mathcal{U} \mid \Delta'}\,(\Rightarrow_{\mathtt{e}*})$$
- If $o = (\mu\alpha.c)u$, then $o' = \mu\alpha.c\{\alpha /\!\!/ u\}$ with $\alpha \in \mathtt{fn}(c)$. Moreover, $\alpha \in \mathtt{fn}(c\{\alpha /\!\!/ u\})$ and $\Phi'$ has the following form:
$$\dfrac{\Gamma' \vdash c\{\alpha /\!\!/ u\} : \# \mid \alpha : \mathcal{U}; \Delta'}{\Gamma' \vdash \mu\alpha.c\{\alpha /\!\!/ u\} : \mathcal{U} \mid \Delta'}\,(\#_{\mathtt{e}})$$
  where $\mathcal{U} \neq \langle\,\rangle$ holds by Lemma 4.6, since $\alpha \in \mathtt{fn}(c\{\alpha /\!\!/ u\})$, so that the $\#_{\mathtt{e}}$ rule is correctly applied. Then the Reverse Replacement Lemma 5.6 yields:
  - $\Gamma' = \Gamma_c \wedge \Gamma_u$,
  - $\Delta' = \Delta_c \vee \Delta_u$,
  - $\mathcal{U} = \vee_{k\in K} \mathcal{V}_k$,
  - $\rhd\Gamma_c \vdash c : \# \mid \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k\in K}; \Delta_c$, and
  - $\rhd\Gamma_u \Vdash u : \wedge_{k\in K} \mathcal{I}_k^* \mid \Delta_u$.

  Moreover, $\mathcal{U} \neq \langle\,\rangle$ implies $K \neq \emptyset$, thus $\langle \mathcal{I}_k \to \mathcal{V}_k \rangle^*_{k\in K} = \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k\in K}$ and we conclude by constructing the following derivation:
$$\dfrac{\dfrac{\rhd\Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_c; \Delta_c}{\Gamma_c \Vdash \mu\alpha.c : \mathcal{V}_c \mid \Delta_c}\,(\#_{\mathtt{e}}) \qquad \rhd \Gamma_u \Vdash u : \mathcal{I}_u \mid \Delta_u}{\Gamma' \Vdash (\mu\alpha.c)u : \mathcal{U} \mid \Delta'}\,(\Rightarrow_{\mathtt{e}*})$$
  where $\mathcal{V}_c = \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k\in K}$, $\mathcal{I}_u = \wedge_{k\in K} \mathcal{I}_k^*$

□

Weighted Subject reduction for the $\lambda\mu_{\mathtt{s}}$-calculus (Lemma 1) is based on the fact that linear substitution (Lemma 10.1) and linear replacement (Lemma 10.2) preserve types.

**Lemma 10.1** (**Linear Substitution**). *Let* $\Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I} \mid \Delta_u$. *If* $\Phi_{\mathtt{OT}[\![x]\!]} \rhd \Gamma; x : \mathcal{I} \vdash \mathtt{OT}[\![x]\!] : \mathcal{A} \mid \Delta$, *then there exist* $\mathcal{I}_1, \mathcal{I}_2, \Gamma_u^1, \Gamma_u^2, \Delta_u^1, \Delta_u^2$ *s.t.*
- $\mathcal{I} = \mathcal{I}_1 \wedge \mathcal{I}_2$, *where* $\mathcal{I}_1 \neq [\,]$,
- $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ *and* $\Delta_u = \Delta_u^1 \vee \Delta_u^2$,
- $\Theta_u^1 \rhd \Gamma_u^1 \Vdash u : \mathcal{I}_1 \mid \Delta_u^1$,
- $\Theta_u^2 \rhd \Gamma_u^2 \Vdash u : \mathcal{I}_2 \mid \Delta_u^2$,
- $\Phi_{\mathtt{T}[\![u]\!]} \rhd \Gamma \wedge \Gamma_u^1; x : \mathcal{I}_2 \vdash \mathtt{OT}[\![u]\!] : \mathcal{A} \mid \Delta \vee \Delta_u^1$, *and*
- $\mathtt{sz}\left(\Phi_{\mathtt{OT}[\![u]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{OT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right) - |\mathcal{I}_1|$.

*Proof.* The proof is by induction on the context $\mathtt{OT}$ so we need to prove the statement of the lemma for regular derivations simultaneously with the following one for *non-empty* auxiliary derivations: if $\Phi_{\mathtt{TT}[\![x]\!]} \rhd \Gamma; x : \mathcal{I} \Vdash \mathtt{TT}[\![x]\!] : \mathcal{J} \mid \Delta$ and $\mathcal{J} \neq [\,]$, then there exist $\mathcal{I}_1, \mathcal{I}_2, \Gamma_u^1, \Gamma_u^2, \Delta_u^1, \Delta_u^2$ s.t.
- $\mathcal{I} = \mathcal{I}_1 \wedge \mathcal{I}_2$, where $\mathcal{I}_1 \neq [\,]$,
- $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ and $\Delta_u = \Delta_u^1 \vee \Delta_u^2$,
- $\Theta_u^1 \rhd \Gamma_u^1 \Vdash u : \mathcal{I}_1 \mid \Delta_u^1$,
- $\Theta_u^2 \rhd \Gamma_u^2 \Vdash u : \mathcal{I}_2 \mid \Delta_u^2$,
- $\Phi_{\mathtt{TT}[\![u]\!]} \rhd \Gamma \wedge \Gamma_u^1; x : \mathcal{I}_2 \Vdash \mathtt{TT}[\![u]\!] : \mathcal{J} \mid \Delta \vee \Delta_u^1$, and
- $\mathtt{sz}\left(\Phi_{\mathtt{TT}[\![u]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right) - |\mathcal{I}_1|$.

Now, we can start the proof. Notice that $\mathcal{I} \neq [\,]$ by Lemma 7.3, since $x \in \mathtt{fv}(\mathtt{OT}[\![x]\!])$ (resp. $x \in \mathtt{fv}(\mathtt{TT}[\![x]\!])$). We only show the case $\mathtt{OT} = \Box$ since all the other ones are straightforward. So assume $\mathtt{OT} = \Box$. Then $\mathcal{I} = [\mathcal{U}]$ for some $\mathcal{U}$ and the derivation $\Phi_x$ has the following form :

$$\Phi_x = \frac{}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\,(\mathtt{ax})$$

Thus, $\mathtt{sz}\left(\Phi_x\right) = 1$. We set then $\Theta_u^1 = \Theta_u$ and $\Theta_u^2 = \dfrac{}{\Vdash u : [\,] \mid}\,(\wedge)$
We have $\mathtt{sz}\left(\Phi_u\right) = \mathtt{sz}\left(\Theta_u^1\right) = \mathtt{sz}\left(\Phi_x\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}_1|$ since $|\mathcal{I}_1| = 1$. □

**Lemma 10.2** (**Linear Replacement**). *Let* $\Theta_u \rhd \Gamma_u \Vdash u : \wedge_{\ell \in L} \mathcal{I}_\ell^* \mid \Delta_u$ *s.t.* $\alpha \notin \mathtt{fv}(u)$. *If* $\Phi_{\mathtt{OC}[\![[\alpha]t]\!]} \rhd \Gamma \vdash \mathtt{OC}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}; \Delta$, *then there exist* $L_1, L_2, \Gamma_u^1, \Gamma_u^2, \Delta_u^1, \Delta_u^2, \Phi_{\mathtt{OC}[\![[\alpha']tu]\!]}$ *s.t.*
- $L = L_1 \uplus L_2$, *where* $L_1 \neq \emptyset$.
- $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ *and* $\Delta_u = \Delta_u^1 \vee \Delta_u^2$,
- $\Theta_u^1 \rhd \Gamma_u^1 \Vdash u : \wedge_{\ell \in L_1} \mathcal{I}_\ell^* \mid \Delta_u^1$,
- $\Theta_u^2 \rhd \Gamma_u^2 \Vdash u : \wedge_{\ell \in L_2} \mathcal{I}_\ell^* \mid \Delta_u^2$,
- $\Phi_{\mathtt{OC}[\![[\alpha']tu]\!]} \rhd \Gamma \wedge \Gamma_u^1 \vdash \mathtt{OC}[\![[\alpha']tu]\!] : \mathcal{A} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L_2}; \alpha' : \vee_{\ell \in L_1} \mathcal{V}_\ell \vee \Delta \vee \Delta_u^1$, *and*
- $\mathtt{sz}\left(\Phi_{\mathtt{OC}[\![[\alpha']tu]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{OC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right)$.

*Proof.* The proof is by induction on the context $\mathtt{OC}$ so we need to prove the statement of the lemma for regular derivations simultaneously with the following one for *non-empty* auxiliary derivations: if $\Phi_{\mathtt{TC}[\![[\alpha]t]\!]} \rhd \Gamma \Vdash \mathtt{TC}[\![[\alpha]t]\!] : \mathcal{J} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}; \Delta$ and $\mathcal{J} \neq [\,]$, then there exist $L_1, L_2, \Gamma_u^1, \Gamma_u^2, \Delta_u^1, \Delta_u^2, \Phi_{\mathtt{TC}[\![[al']tu]\!]}$ s.t.

- $L = L_1 \uplus L_2$, where $L_1 \neq \emptyset$.
- $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ and $\Delta_u = \Delta_u^1 \vee \Delta_u^2$,
- $\Theta_u^1 \triangleright \Gamma_u^1 \Vdash u : \wedge_{\ell \in L_1} \mathcal{I}_\ell^* \mid \Delta_u^1$,
- $\Theta_u^2 \triangleright \Gamma_u^2 \Vdash u : \wedge_{\ell \in L_2} \mathcal{I}_\ell^* \mid \Delta_u^2$,
- $\Phi_{\mathtt{TC}[\![[\alpha']tu]\!]} \triangleright \Gamma \wedge \Gamma_u^1 \Vdash \mathtt{TC}[\![[\alpha']tu]\!] : \mathcal{J} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L_2}; \alpha' : \vee_{\ell \in L_1} \mathcal{V}_\ell \vee \Delta \vee \Delta_u^1$, and
- $\mathtt{sz}\left(\Phi_{\mathtt{TC}[\![[\alpha']tu]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{TC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right)$.

Now, we can start the proof. Notice that $L \neq \emptyset$ by Lemma 7.3, since $\alpha \in \mathtt{fn}(\mathtt{OC}[\![[\alpha]t]\!])$ (resp. $\alpha \in \mathtt{fn}(\mathtt{TC}[\![[\alpha]t]\!])$). We only show the case $\mathtt{OC} = \boxdot$ since all the other ones are straightforward.

So assume $\mathtt{OC} = \boxdot$. Then the derivation $\Phi_{[\alpha]t}$ has the following form, where $K \neq \emptyset$ holds by Lemma 4.5:

$$\frac{\Phi_t \triangleright \Gamma \vdash t : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L \setminus K}; \Delta}{\Gamma \vdash [\alpha]t : \# \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}; \Delta} (\#_{\mathtt{i}})$$

Thus, $\mathtt{sz}\left(\Phi_{[\alpha]t}\right) = \mathtt{sz}\left(\Phi_t\right) + \mathtt{ar}(\langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K}) = \mathtt{sz}\left(\Phi_t\right) + |K| + \mathtt{ar}(\vee_{k \in K} \mathcal{V}_k)$. We set $L_1 = K$ and $L_2 = L \setminus K$ and we write $\wedge_{\ell \in L} \mathcal{I}_\ell^*$ as $(\wedge_{\ell \in L_1} \mathcal{I}_\ell^*) \wedge (\wedge_{\ell \in L_2} \mathcal{I}_\ell^*)$. Then by Lemma 5.1 there are $\Theta_u^1 \triangleright \Gamma_u^1 \Vdash u : \wedge_{\ell \in L_1} \mathcal{I}_\ell^* \mid \Delta_u^1$, $\Theta_u^2 \triangleright \Gamma_u^2 \Vdash u : \wedge_{\ell \in L_2} \mathcal{I}_\ell^* \mid \Delta_u^2$ s.t. $\Gamma_u^1 \wedge \Gamma_u^2 = \Gamma_u$, $\Delta_u^1 \vee \Delta_u^2 = \Delta_u$. We set $\mathcal{V} = \vee_{\ell \in L_1} \mathcal{V}_\ell$ and then construct the following derivation $\Phi_{[\alpha']tu}$:

$$\dfrac{\dfrac{\Phi_t \qquad \Theta_u^1}{\Gamma \wedge \Gamma_u^1 \vdash t\, u : \vee_{\ell \in L_1} \mathcal{V}_\ell \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L_2}; \Delta \vee \Delta_u^1} (\Rightarrow_{\mathtt{e}*})}{\Gamma \wedge \Gamma_u^1 \vdash [\alpha']tu : \# \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L_2}; \alpha' : \mathcal{V} \vee \Delta \vee \Delta_u^1} (\#_{\mathtt{i}})$$

We have:

$$\begin{array}{l} \mathtt{sz}\left(\Phi_{[\alpha']tu}\right) = \mathtt{sz}\left(\Phi_{tu}\right) + \mathtt{ar}(\vee_{k \in K} \mathcal{V}_k) \\ = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^1\right) + |K| + \mathtt{ar}(\vee_{k \in K} \mathcal{V}_k) \\ = \mathtt{sz}\left(\Phi_t\right) + \mathtt{sz}\left(\Theta_u^1\right) + \mathtt{ar}(\langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K}) \\ = \mathtt{sz}\left(\Phi_{[\alpha]t}\right) + \mathtt{sz}\left(\Theta_u^1\right) \end{array}$$

□

**Property 1 (Weighted Subject Reduction for $\lambda\mu_{\mathtt{s}}$).** Let $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$. If $o \to o'$ is a non-erasing step, then $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$ and $\mathtt{sz}\left(\Phi\right) > \mathtt{sz}\left(\Phi'\right)$.

*Proof.* By induction on the reduction relation $\to$. We only show the main cases of reduction at the root, the other ones being straightforward.

- If $o = \mathtt{L}[(\lambda x.t)]u \to \mathtt{L}[t[x/u]] = o'$: we proceed by induction on $\mathtt{L}$, by detailing only the case $\mathtt{L} = \Box$ as the other one is straightforward.

  The derivation $\Phi$ has the following form:

$$\Phi = \dfrac{\dfrac{\Phi_t \triangleright \Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t}{\Gamma_t \vdash \lambda x.t : \langle \mathcal{I} \to \mathcal{U} \rangle \mid \Delta_t} (\Rightarrow_{\mathtt{i}}) \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta}{\Gamma \vdash (\lambda x.t)u : \mathcal{U} \mid \Delta} (\Rightarrow_{\mathtt{e}*})$$

where $\Gamma = \Gamma_t \wedge \Gamma_u$, $\Delta = \Delta_t \vee \Delta_u$ and $\mathcal{A} = \mathcal{U}$. We then construct the following derivation $\Phi'$:

$$\dfrac{\Phi_t \rhd \Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t \qquad \Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma_t \wedge \Gamma_u \vdash t[x/u] : \mathcal{U} \mid \Delta_t \vee \Delta_u}\,(\mathtt{s})$$

We conclude since $\mathtt{sz}(\Phi) = \mathtt{sz}(\Phi_t) + \mathtt{sz}(\Theta_u) + 2 > \mathtt{sz}(\Phi_t) + \mathtt{sz}(\Theta_u) = \mathtt{sz}(\Phi')$.

- If $o = \mathtt{L}[\mu\alpha.c]u \to \mathtt{L}[\mu\alpha'.c\langle\alpha /\!\!/ \alpha'.u\rangle] = o'$: we proceed by induction on $\mathtt{L}$, by detailing only the case $\mathtt{L} = \Box$ as the other one is straightforward. The derivation $\Phi$ has the following form:

$$\Phi = \dfrac{\dfrac{\Phi_c \rhd \Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_c; \Delta_c}{\Gamma_c \vdash \mu\alpha.c : \mathcal{V}_c^* \mid \Delta_c}\,(\#_{\mathtt{e}}) \qquad \Theta_u \rhd \Gamma_u \Vdash u : \mathcal{I}_u^* \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash (\mu\alpha.c)u : \mathcal{U} \mid \Delta_u}\,(\Rightarrow_{\mathtt{e}*})$$

where $\mathcal{V}_c^* = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}$, $\mathcal{I}_u^* = \wedge_{\ell \in L} \mathcal{I}_\ell^*$, $\mathcal{U} = \vee_{\ell \in L} \mathcal{V}_\ell$, $\Gamma = \Gamma_c \wedge \Gamma_u$ and $\Delta = \Delta_c \vee \Delta_u$.

Moreover, Lemma 4.5 implies $L \neq \emptyset$, so that $\wedge_{\ell \in L} \mathcal{I}_\ell^* = (\wedge_{\ell \in L} \mathcal{I}_\ell^*)^*$.

We then construct the following derivation $\Phi'$:

$$\dfrac{\dfrac{\Phi_c \qquad \Theta_u}{\Gamma' \wedge \Gamma_u \vdash c\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta' \vee \Delta_u; \alpha' : \mathcal{U}}\,(\mathtt{r})}{\Gamma' \wedge \Gamma_u \vdash \mu\alpha'.c\langle\alpha /\!\!/ \alpha'.u\rangle : \mathcal{U} \mid \Delta' \vee \Delta_u}\,(\#_{\mathtt{e}})$$

We conclude since $|L| \geq 1$ in the following equation:

$$\begin{aligned} &\mathtt{sz}(\Phi') = \mathtt{sz}\left(\Phi_{c\langle\alpha /\!\!/ \alpha'.u\rangle}\right) + 1 \\ &= \mathtt{sz}(\Phi_c) + \mathtt{sz}(\Theta_u) + |L| - \tfrac{1}{2} + 1 \\ &= \mathtt{sz}(\Phi_{\mu\alpha.c}) + \mathtt{sz}(\Theta_u) + |L| - \tfrac{1}{2} \\ &< \mathtt{sz}(\Phi_{\mu\alpha.c}) + \mathtt{sz}(\Theta_u) + |L| = \mathtt{sz}(\Phi) \end{aligned}$$

- If $o = \mathtt{TT}[\![x]\!][x/u] \to \mathtt{TT}[\![u]\!][x/u] = o'$, with $|\mathtt{TT}[\![x]\!]|_x > 1$. The derivation $\Phi$ has the following form:

$$\dfrac{\Phi'_{\mathtt{TT}[\![x]\!]} \rhd \Gamma'; x : \mathcal{I} \vdash \mathtt{TT}[\![x]\!] : \mathcal{U} \mid \Delta' \qquad \Theta_u \rhd \Gamma_u \vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma' \wedge \Gamma_u \vdash \mathtt{TT}[\![x]\!][x/u] : \mathcal{U} \mid \Delta' \vee \Delta_u}\,(\mathtt{s})$$

Moreover, $|\mathtt{TT}[\![x]\!]|_x > 1$ so that Lemma 7.3 applied to $\Phi_{\mathtt{TT}[\![x]\!]}$ gives $\mathcal{I} \neq [\,]$ and thus $\mathcal{I}^* = \mathcal{I}$. We can then apply Lemma 10.1 which gives a derivation

$$\Phi_{\mathtt{TT}[\![u]\!]} \rhd \Gamma' \wedge \Gamma_u^1; x : \mathcal{I}_2 \vdash \mathtt{TT}[\![u]\!] : \mathcal{U} \mid \Delta' \vee \Delta_u^1$$

where $\mathcal{I} = \mathcal{I}_1 \wedge \mathcal{I}_2$ and $\mathcal{I}_1 \neq [\,]$ and $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ and $\Delta_u = \Delta_u^1 \vee \Delta_u^2$. Moreover $\Theta_u^1 \rhd \Gamma_u^1 \Vdash u : \mathcal{I}_1 \mid \Delta_u^1$, $\Theta_u^2 \rhd \Gamma_u^2 \Vdash u : \mathcal{I}_2 \mid \Delta_u^2$, and $\mathtt{sz}\left(\Phi_{\mathtt{TT}[\![u]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right) - |\mathcal{I}_1|$.

The hypothesis $|\mathtt{T}[\![x]\!]|_x > 1$ implies $|\mathtt{T}[\![u]\!]|_x > 0$, then $\mathcal{I}_2 \neq [\,]$ by Lemma 7.3 applied to $\Phi_{\mathtt{TT}[\![u]\!]}$ so that $\mathcal{I}_2^* = \mathcal{I}_2$. We can then construct the derivation $\Phi'$ as follows:

$$\dfrac{\Phi_{\mathtt{TT}[\![u]\!]} \qquad \Theta_u^2}{\Gamma' \wedge \Gamma \vdash \mathtt{TT}[\![u]\!][x/u] : \mathcal{U} \mid \Delta' \wedge \Delta}\,(\mathtt{s})$$

We conclude since $\mathtt{sz}(\Phi') = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![u]\!]}\right) + \mathtt{sz}\left(\Theta_u^2\right) =_{Lemma\ 10.1} \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right) - |\mathcal{I}_1| + \mathtt{sz}\left(\Theta_u^2\right) = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}(\Theta_u) - |\mathcal{I}_1| < \mathtt{sz}(\Phi)$.

The step $<$ is justified by $\mathcal{I}_1 \neq [\,]$.

- If $o = \mathtt{TT}[\![x]\!][x/u] \to \mathtt{TT}[\![u]\!] = o'$, with $|\mathtt{TT}[\![x]\!]|_x = 1$. The derivation $\Phi$ has the following form:
$$\frac{\Phi_{\mathtt{TT}[\![x]\!]} \triangleright \Gamma'; x : \mathcal{I} \vdash \mathtt{TT}[\![x]\!] : \mathcal{U} \mid \Delta' \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma' \wedge \Gamma_u \vdash \mathtt{TT}[\![x]\!][x/u] : \mathcal{U} \mid \Delta' \vee \Delta_u}\,(\mathtt{s})$$
Lemma 7.3 applied to $\Phi_{\mathtt{TT}[\![x]\!]}$ gives $\mathcal{I} \neq [\,]$ and thus $\mathcal{I}^* = \mathcal{I}$. We can then apply Lemma 10.1 which gives a derivation
$$\Phi_{\mathtt{TT}[\![u]\!]} \triangleright \Gamma' \wedge \Gamma^1_u; x : \mathcal{I}_2 \vdash \mathtt{TT}[\![u]\!] : \mathcal{U} \mid \Delta' \vee \Delta^1_u$$
where $\mathcal{I} = \mathcal{I}_1 \wedge \mathcal{I}_2$ and $\mathcal{I}_1 \neq [\,]$ and $\Gamma_u = \Gamma^1_u \wedge \Gamma^2_u$ and $\Delta_u = \Delta^1_u \vee \Delta^2_u$. Moreover $\Theta^1_u \triangleright \Gamma^1_u \Vdash u : \mathcal{I}_1 \mid \Delta^1_u$, $\Theta^2_u \triangleright \Gamma^2_u \Vdash u : \mathcal{I}_2 \mid \Delta^2_u$, and $\mathtt{sz}\left(\Phi_{\mathtt{TT}[\![u]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta^1_u\right) - |\mathcal{I}_1|$. By hypothesis $|\mathtt{TT}[\![x]\!]|_x = 1$ so that $|\mathtt{TT}[\![u]\!]|_x = 0$, then $\mathcal{I}_2 = \emptyset$ by Lemma 7.3 applied to $\Phi_{\mathtt{TT}[\![u]\!]}$. Thus $\mathcal{I} = \mathcal{I}_1$. We then set $\Phi' = \Phi_{\mathtt{TT}[\![u]\!]}$ and conclude since
$$\mathtt{sz}\left(\Phi'\right) = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![u]\!]}\right) =_{Lemma 10.1} \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta^1_u\right) - |\mathcal{I}_1| = \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u\right) - |\mathcal{I}| <$$
$$= \mathtt{sz}\left(\Phi_{\mathtt{TT}[\![x]\!]}\right) + \mathtt{sz}\left(\Theta_u\right) = \mathtt{sz}\left(\Phi\right)$$
The step $<$ is justified by $\mathcal{I} = \mathcal{I}_1 \neq [\,]$.
- If $o = \mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \alpha'.u\rangle \to \mathtt{CC}[\![[\alpha']tu]\!]\langle\alpha /\!\!/ \alpha'.u\rangle = o'$, with $|\mathtt{CC}[\![[\alpha]t]\!]_\alpha > 1$. Then $\Phi$ has the following form
$$\frac{\Phi_c \triangleright \Gamma_c \vdash \mathtt{CC}[\![[\alpha]t]\!] : \# \mid \Delta_c; \alpha : \mathcal{V}' \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}_u \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash \mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta_c \vee \Delta_u \vee \alpha' : \vee_{\ell \in L} \mathcal{V}_\ell}\,(\mathtt{r})$$
where $c = \mathtt{CC}[\![[\alpha]t]\!]$, $\mathcal{V}' = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}$, $\mathcal{I}_u = (\wedge_{\ell \in L} \mathcal{I}^*_\ell)^*$, $\mathcal{A} = \#$, $\Gamma = \Gamma_c \wedge \Gamma_u$ and $\Delta = \Delta_c \vee \Delta_u \vee \alpha' : \vee_{\ell \in L} \mathcal{V}_\ell$. Since $|\mathtt{CC}[\![[\alpha]t]\!]|_\alpha > 1$ implies $L \neq \emptyset$ by Lemma 7.3, we have that $\mathcal{I}_u = \wedge_{\ell \in L} \mathcal{I}^*_\ell$. By Lemma 10.2 there are $L_1$, $L_2$, $\Gamma^1_u$, $\Gamma^2_u$, $\Delta^1_u$, $\Delta^2_u$, $\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]}$ s.t.
  - $L = L_1 \uplus L_2$, where $L_1 \neq \emptyset$.
  - $\Gamma_u = \Gamma^1_u \wedge \Gamma^2_u$ and $\Delta_u = \Delta^1_u \vee \Delta^2_u$,
  - $\Theta^1_u \triangleright \Gamma^1_u \Vdash u : \wedge_{\ell \in L_1} \mathcal{I}^*_\ell \mid \Delta^1_u$,
  - $\Theta^2_u \triangleright \Gamma^2_u \Vdash u : \wedge_{\ell \in L_2} \mathcal{I}^*_\ell \mid \Delta^2_u$,
  - $\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]} \triangleright \Gamma_c \wedge \Gamma^1_u \vdash \mathtt{CC}[\![[\alpha']tu]\!] : \mathcal{A} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L_2}; \alpha' : \vee_{\ell \in L_1} \mathcal{V}_\ell \vee \Delta_c \vee \Delta^1_u$, and
  - $\mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta^1_u\right)$.

  Moreover, $|\mathtt{CC}[\![[\alpha]t]\!]|_\alpha > 1$ implies $|\mathtt{CC}[\![[\alpha']tu]\!]|_\alpha > 0$ so that $L_2 \neq \emptyset$ holds by Lemma 7.3 and thus $\wedge_{\ell \in L_2} \mathcal{I}^*_\ell = (\wedge_{\ell \in L_2} \mathcal{I}^*_\ell)^*$. Then we can build the following derivation $\Phi'$:
$$\frac{\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]} \qquad \Theta^2_u}{\Gamma' \vdash \mathtt{CC}[\![[\alpha']tu]\!]\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta'}\,(\mathtt{r})$$
where $\Gamma' = (\Gamma_c \wedge \Gamma^1_u) \wedge \Gamma^2_u = \Gamma$, $\Delta' = (\alpha' : \vee_{\ell \in L_1} \mathcal{V}_\ell \vee \Delta_c \vee \Delta^1_u) \vee \Delta^2_u \vee (\alpha' : \vee_{\ell \in L_2} \mathcal{V}_\ell) = \Delta$.

  We conclude since
$$\begin{array}{l} \mathtt{sz}\left(\Phi'\right) = \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]}\right) + \mathtt{sz}\left(\Theta^2_u\right) + |L_2| - \frac{1}{2} \\ =_{Lemma\ 10.2} \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta^1_u\right) + \mathtt{sz}\left(\Theta^2_u\right) + |L_2| - \frac{1}{2} \\ = \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u\right) + |L_2| - \frac{1}{2} \\ < \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u\right) + |L| - \frac{1}{2} = \mathtt{sz}\left(\Phi\right) \end{array}$$
The step $<$ is justified because $L_1 \neq \emptyset$ and thus $|L_2| < |L|$.
- If $o = \mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \alpha'.u\rangle \to \mathtt{CC}[\![[\alpha']tu]\!] = o'$, with $|\mathtt{CC}[\![[\alpha]t]\!]|_\alpha = 1$. The derivation $\Phi$ has the following form

$$\frac{\Phi_c \triangleright \Gamma_c \vdash \mathtt{CC}[\![[\alpha]t]\!] : \# \mid \Delta_c; \alpha : \mathcal{V}' \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}_u \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash \mathtt{CC}[\![[\alpha]t]\!]\langle \alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta_c \vee \Delta_u \vee \alpha' : \vee_{\ell\in L}\mathcal{V}_\ell}\,(\mathtt{r})$$

where $c = \mathtt{CC}[\![[\alpha]t]\!]$, $\mathcal{V}' = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle_{\ell\in L}$, $\mathcal{I}_u = (\wedge_{\ell\in L}\mathcal{I}_\ell^*)^*$, $\mathcal{A} = \#$, $\Gamma = \Gamma_c \wedge \Gamma_u$ and $\Delta = \Delta_c \vee \Delta_u \vee \alpha' : \vee_{\ell\in L}\mathcal{V}_\ell$. Since $|\mathtt{CC}[\![[\alpha]t]\!]|_\alpha = 1$ implies $L \neq \emptyset$ by Lemma 7.3, we have that $\mathcal{I}_u = \wedge_{\ell\in L}\mathcal{I}_\ell^*$. By Lemma 10.2 there are $L_1$, $L_2$, $\Gamma_u^1$, $\Gamma_u^2$, $\Delta_u^1$, $\Delta_u^2$, $\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]}$ s.t.

- $L = L_1 \uplus L_2$, where $L_1 \neq \emptyset$.
- $\Gamma_u = \Gamma_u^1 \wedge \Gamma_u^2$ and $\Delta_u = \Delta_u^1 \vee \Delta_u^2$,
- $\Theta_u^1 \triangleright \Gamma_u^1 \Vdash u : \wedge_{\ell\in L_1}\mathcal{I}_\ell^* \mid \Delta_u^1$,
- $\Theta_u^2 \triangleright \Gamma_u^2 \Vdash u : \wedge_{\ell\in L_2}\mathcal{I}_\ell^* \mid \Delta_u^2$,
- $\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]} \triangleright \Gamma_c \wedge \Gamma_u^1 \vdash \mathtt{CC}[\![[\alpha']tu]\!] : \mathcal{A} \mid \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle_{\ell\in L_2}; \alpha' : \vee_{\ell\in L_1}\mathcal{V}_\ell \vee \Delta_c \vee \Delta_u^1$, and
- $\mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha']tu]\!]}\right) = \mathtt{sz}\left(\Phi_{\mathtt{CC}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u^1\right)$.

Moreover, $|\mathtt{C}[\![[\alpha]t]\!]|_\alpha = 1$ implies $|\mathtt{C}[\![[\alpha']tu]\!]|_\alpha = 0$ so that $L_2 = \emptyset$ and $L = L_1$ holds by Lemma 7.3. Thus, $\Theta_u^1 = \Theta_u$ and so on. We then set $\Phi' = \Phi_{\mathtt{C}[\![[\alpha']tu]\!]}$ and conclude since

$$\begin{aligned}&\mathtt{sz}\left(\Phi'\right) = \mathtt{sz}\left(\Phi_{\mathtt{C}[\![[\alpha']tu]\!]}\right)\\ &=_{\mathit{Lemma}\ 10.2}\ \mathtt{sz}\left(\Phi_{\mathtt{C}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u\right)\\ &< \mathtt{sz}\left(\Phi_{\mathtt{C}[\![[\alpha]t]\!]}\right) + \mathtt{sz}\left(\Theta_u\right) + |L| - \tfrac{1}{2} = \mathtt{sz}\left(\Phi\right)\end{aligned}$$

The step $<$ is justified because $L \neq \emptyset$, so that $|L| \geqslant 1$ implies $|L| - \frac{1}{2} > 0$.

□

**Lemma 10.3** (Reverse Partial Substitution)**.** *Let $\Phi \triangleright \Gamma \vdash \mathtt{OT}[\![u]\!] : \mathcal{A} \mid \Delta$, where $x \notin \mathtt{fv}(u)$. Then, there exist $\Gamma_0, \Delta_0, \mathcal{I}_0 \neq [\,], \Gamma_u, \Delta_u$ such that*

- $\Gamma = \Gamma_0 \wedge \Delta_u$,
- $\Delta = \Delta_0 \vee \Delta_u$,
- $\Phi_{\mathtt{OT}[\![x]\!]} \triangleright \Gamma_0 \wedge x : \mathcal{I}_0 \vdash \mathtt{OT}[\![x]\!] : \mathcal{A} \mid \Delta_0$
- $\triangleright\Gamma_u \Vdash u : \mathcal{I}_0 \mid \Delta_u$.

*Proof.* The proof is by induction on the context $\mathtt{OT}$. For this induction to work, we need as usual to adapt the statement for auxiliary derivations. We only show the case $\mathtt{OT} = \Box$ since all the other ones are straightforward and rely on suitable partitions of the contexts in the premises. So assume $\mathtt{OT} = \Box$, then $\mathcal{A} = \mathcal{U}$ for some $\mathcal{U}$. We set $\Gamma_0 = \Delta_0 = \emptyset$, $\mathcal{I}_0 = [\mathcal{U}]$, $\Gamma_u = \Gamma$, $\Delta_u = \Delta$ (so that $\triangleright\Gamma_u \Vdash u : \mathcal{I}_0 \mid \Delta_u$ holds by using the $(\wedge)$ rule), and

$$\Phi_x = \frac{}{x : [\mathcal{U}] \vdash x : \mathcal{U} \mid \emptyset}\,(\mathtt{ax})$$

The claimed set and context equalities trivially hold. □

**Lemma 10.4** (Reverse Partial Replacement)**.** *Let $\Gamma \vdash \mathtt{OC}[\![[\alpha']tu]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}; \Delta$, where $\alpha, \alpha' \notin \mathtt{fn}(u)$. Then there exist $\Gamma_0, \Delta_0, \mathcal{V}_0, K \neq \emptyset, (\mathcal{I}_k)_{k\in K}, (\mathcal{V}_k)_{k\in K}, \Gamma_u, \Delta_u$ such that*

- $\Gamma = \Gamma_0 \wedge \Gamma_u$,
- $\Delta = \Delta_0 \vee \Delta_u$,
- $\mathcal{V} = \mathcal{V}_0 \vee_{k\in K} \mathcal{V}_k$,
- $\triangleright\Gamma_0 \vdash \mathtt{OC}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K} \vee \Delta_0$, *and*
- $\triangleright\Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u$

*Proof.* The proof is by induction on the context $\mathtt{OC}$. For this induction to work, we need as usual to adapt the statement for auxilary derivations. Notice that $\mathcal{V} \neq \langle\,\rangle$ by Lemma 7.3, since $\alpha' \in \mathtt{fn}(\mathtt{OC}[\![[\alpha']tu]\!])$. We only show the case $\mathtt{OC} = \boxdot$ since all the other ones are

straightforward. So assume $\mathtt{OC} = \boxdot$. Then the derivation of $[\alpha']tu$ has the following form, where $K \neq \emptyset$:

$$\dfrac{\dfrac{\Phi_t \triangleright \Gamma_0 \vdash t : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K} \mid \alpha' : \mathcal{V}_0; \Delta_0 \qquad \triangleright \Gamma_u \Vdash u : \wedge_{k\in K}\mathcal{I}_k^* \mid \Delta_u}{\Gamma_0 \wedge \Gamma_u \vdash tu : \vee_{k\in K}\mathcal{V}_k \mid \alpha' : \mathcal{V}_0; \Delta_0 \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}*})}{\Gamma_0 \wedge \Gamma_u \vdash [\alpha']tu : \# \mid \alpha' : \mathcal{V}_0 \vee_{k\in K} \mathcal{V}_k; \Delta_0 \vee \Delta_u}\,(\#_{\mathtt{i}})$$

where $\Gamma = \Gamma_0 \wedge \Gamma_u$, and $\Delta = \Delta_0 \vee \Delta_u$ and $\mathcal{V} = \mathcal{V}_0 \vee_{k\in K} \mathcal{V}_k$.

We then construct the following derivation :

$$\dfrac{\Phi_t \triangleright \Gamma_0 \vdash t : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K} \mid \alpha' : \mathcal{V}_0; \Delta_0}{\Gamma \vdash [\alpha]t : \# \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k\rangle_{k\in K} \vee \Delta_0}\,(\#_{\mathtt{i}})$$

Thus, we have all the claimed set and context equalities. □

**Property 2 (Subject Expansion for $\lambda\mu_{\mathtt{s}}$).** Let $\Phi' \triangleright \Gamma \vdash o' : \mathcal{A} \mid \Delta$. If $o \to_{\lambda\overline{\mu}\mathtt{s}} o'$ (*i.e.* a non-erasing $\lambda\mu_{\mathtt{s}}$-step), then $\Phi \triangleright \Gamma \vdash o : \mathcal{A} \mid \Delta$.

*Proof.* By induction on the non erasing reduction relation $\to_{\lambda\overline{\mu}\mathtt{s}}$. We only show the main cases of non-erasing reduction at the root, the other ones being straightforward.

- If $o = (\mathtt{L}[\lambda x.t])u \to \mathtt{L}[t[x/u]] = o'$, we proceed by induction on $\mathtt{L}$, by detailing only the case $\mathtt{L} = \Box$ as the other one is straightforward.

  The derivation $\Phi'$ has the following form :

  $$\dfrac{\Phi_t \triangleright \Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma \vdash t[x/u] : \mathcal{U} \mid \Delta}\,(\mathtt{s})$$

  We then construct the following derivation $\Phi$:

  $$\dfrac{\dfrac{\Phi_t \triangleright \Gamma_t; x : \mathcal{I} \vdash t : \mathcal{U} \mid \Delta_t}{\Gamma_t \vdash \lambda x.t : \langle \mathcal{I} \to \mathcal{U}\rangle \mid \Delta_t}\,(\Rightarrow_{\mathtt{i}}) \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma \vdash (\lambda x.t)u : \mathcal{U} \mid \Delta}\,(\Rightarrow_{\mathtt{e}*})$$

- If $o = (\mathtt{L}[\mu\alpha.c])u \to \mathtt{L}[\mu\alpha'.c\langle\alpha /\!\!/ \alpha'.u\rangle] = o'$, where $\alpha'$ is fresh, then we proceed by induction on $\mathtt{L}$, by detailing only the case $\mathtt{L} = \Box$ as the other one is straightforward. Then $\Phi'$ has the following form :

  $$\dfrac{\dfrac{\Phi_c \triangleright \Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_\alpha; \Delta_c \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}_u \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash c\langle\alpha /\!\!/ \alpha'.u\rangle : \# \mid \Delta_c \vee \Delta_u; \alpha' : \mathcal{V}_{\alpha'}}\,(\mathtt{r})}{\Gamma_c \wedge \Gamma_u \vdash \mu\alpha'.c\langle\alpha /\!\!/ \alpha'.u\rangle : (\mathcal{V}_{\alpha'})^* \mid \Delta_c \vee \Delta_u}\,(\#_{\mathtt{e}})$$

  where $\mathcal{V}_\alpha = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle_{\ell\in L}$, $\mathcal{I}_u = (\wedge_{\ell\in L}\mathcal{I}_\ell^*)^*$, $\mathcal{V}_{\alpha'} = \vee_{\ell\in L}\mathcal{V}_\ell$, $\mathcal{A} = (\mathcal{V}_{\alpha'})^* = (\vee_{\ell\in L}\mathcal{V}_\ell)^*$, $\Gamma = \Gamma_c \wedge \Gamma_u$ and $\Delta = \Delta_c \vee \Delta_u$. Notice that the name assignment of the judgment typing $c\langle\alpha /\!\!/ \alpha'.u\rangle$ has the form $\Delta_c \vee \Delta_u; \alpha' : \mathcal{V}_{\alpha'}$ since $\alpha'$ is a fresh name by hypothesis, so that $\alpha' \notin \mathtt{dom}(\Delta_c \vee \Delta_u)$ holds by Lemma 7.3. We now consider two cases:
  - If $L \neq \emptyset$, then $\langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle^*_{\ell\in L} = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle_{\ell\in L}$, $(\wedge_{\ell\in L}\mathcal{I}_\ell^*)^* = \wedge_{\ell\in L}\mathcal{I}_\ell^*$, $\mathcal{A} = (\vee_{\ell\in L}\mathcal{V}_\ell)^* = \vee_{\ell\in L}\mathcal{V}_\ell$, so that we construct the following derivation $\Phi$:

    $$\dfrac{\dfrac{\Phi_c \triangleright \Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_\alpha; \Delta_c}{\Gamma_c \vdash \mu\alpha.c : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell\rangle_{\ell\in L} \mid \Delta_c}\,(\#_{\mathtt{e}}) \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \wedge_{\ell\in L}\mathcal{I}_\ell^* \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash (\mu\alpha.c)u : \vee_{\ell\in L}\mathcal{V}_\ell \mid \Delta_c \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}*})$$

– If $L = \emptyset$, then, in the derivation above, $(\mathcal{V}_{\alpha'})^* = (\vee_{\ell\in L}\mathcal{V}_\ell)^* = \langle\xi\rangle$ for some blind type $\xi$. Then we choose $\langle\mathcal{I}_\ell \to \mathcal{V}_\ell\rangle^*_{\ell\in L}$ to be $\langle[\,] \to \langle\xi\rangle\rangle$, which is a blind type. We then construct the following derivation $\Phi$:

$$\dfrac{\dfrac{\Phi_c \triangleright \Gamma_c \vdash c : \# \mid \alpha : \mathcal{V}_\alpha; \Delta_c}{\Gamma_c \vdash \mu\alpha.c : \langle[\,] \to \langle\xi\rangle\rangle \mid \Delta_c}\,(\#_{\mathtt{e}}) \qquad \Theta_u \triangleright \Gamma_u \Vdash u : [\,]^* \mid \Delta_u}{\Gamma_c \wedge \Gamma_u \vdash (\mu\alpha.c)u : \langle\xi\rangle \mid \Delta_c \vee \Delta_u}\,(\Rightarrow_{\mathtt{e}*})$$

We conclude since $\mathcal{A} = \langle\xi\rangle$.

- If $o = \mathtt{TT}[\![x]\!][x/u] \to \mathtt{TT}[\![u]\!][x/u] = o'$, with $|\mathtt{TT}[\![x]\!]|_x > 1$. The derivation $\Phi'$ has the following form:

$$\dfrac{\Phi_{\mathtt{TT}[\![u]\!]} \triangleright \Gamma_{\mathtt{TT}}; x : \mathcal{I} \vdash \mathtt{TT}[\![u]\!] : \mathcal{A} \mid \Delta_{\mathtt{TT}} \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}^* \mid \Delta_u}{\Gamma_{\mathtt{TT}} \wedge \Gamma_u \vdash \mathtt{TT}[\![u]\!][x/u] : \mathcal{A} \mid \Delta_{\mathtt{TT}} \vee \Delta_u}\,(\mathtt{s})$$

where $x \in \mathtt{fv}(\mathtt{TT}[\![u]\!])$ implies $\mathcal{I} \neq [\,]$ by Lemma 7.3, so that $\mathcal{I}^* = \mathcal{I}$.

By Lemma 10.3 applied to $\Phi_{\mathtt{TT}[\![u]\!]}$, we have $\Gamma'_0$, $\Delta_0$, $\mathcal{I}_0 \neq [\,]$, $\Gamma'_u$, $\Delta'_u$ such that
– $\Gamma_{\mathtt{TT}}; x : \mathcal{I} = \Gamma'_0 \wedge \Gamma'_u$,
– $\Delta_{\mathtt{TT}} = \Delta_0 \vee \Delta'_u$,
– $\Phi_{\mathtt{TT}[\![x]\!]} \triangleright \Gamma'_0 \wedge x : \mathcal{I}_0 \vdash \mathtt{TT}[\![x]\!] : \mathcal{A} \mid \Delta_0$
– $\triangleright\Gamma'_u \Vdash u : \mathcal{I}_0 \mid \Delta'_u$.
We set $\mathcal{I}'' = \mathcal{I} \wedge \mathcal{I}_0$, $\Gamma''_u = \Gamma_u \wedge \Gamma'_u$, $\Delta''_u = \Delta_u \vee \Delta'_u$. Thus, in particular, $(\mathcal{I}'')^* = \mathcal{I}''$. By Lemma 7.3, $x \notin \mathtt{dom}(\Gamma'_u)$, so that $\Gamma'_0 = \Gamma_0; x : \mathcal{I}$ for some $\Gamma_0$ and thus $\Gamma'_0 \wedge x : \mathcal{I}_0 = \Gamma_0; x : \mathcal{I}''$. By Lemma 5.1, there is a derivation $\Theta''_u \triangleright \Gamma''_u \Vdash u : \mathcal{I}'' \mid \Delta''_u$. We then construct the following derivation $\Phi$ :

$$\dfrac{\Phi_{\mathtt{TT}[\![x]\!]} \qquad \Theta''_u \triangleright \Gamma''_u \vdash u : \mathcal{I}'' \mid \Delta''_u}{\Gamma_0 \wedge \Gamma''_u \vdash \mathtt{TT}[\![x]\!][x/u] : \mathcal{A} \mid \Delta_0 \vee \Delta''_u}\,(\mathtt{s})$$

We conclude since $\Gamma_0 \wedge \Gamma''_u = \Gamma_0 \wedge \Gamma'_u \wedge \Gamma_u = \Gamma_{\mathtt{TT}} \wedge \Gamma_u = \Gamma$ and $\Delta_0 \vee \Delta''_u = \Delta_0 \vee \Delta'_u \vee \Delta_u = \Delta_{\mathtt{TT}} \vee \Delta_u = \Delta$.

- If $o = \mathtt{TT}[\![x]\!][x/u] \to \mathtt{TT}[\![u]\!] = o'$, with $|\mathtt{TT}[\![x]\!]|_x = 1$. The derivation $\Phi'$ ends with $\Gamma \vdash \mathtt{TT}[\![u]\!] : \mathcal{A} \mid \Delta$ where $x \notin \mathtt{dom}(\Gamma)$ by Lemma 7.3. By Lemma 10.3 applied to $\Phi'$, we have $\Gamma_0$, $\Delta_0$, $\mathcal{I}_0 \neq [\,]$, $\Gamma_u$, $\Delta_u$ such that
– $\Gamma = \Gamma_0 \wedge \Gamma_u$,
– $\Delta = \Delta_0 \vee \Delta_u$,
– $\Phi_{\mathtt{TT}[\![x]\!]} \triangleright \Gamma_0 \wedge x : \mathcal{I}_0 \vdash \mathtt{TT}[\![x]\!] : \mathcal{A} \mid \Delta_0$
– $\triangleright\Gamma_u \Vdash u : \mathcal{I}_0 \mid \Delta_u$.
Thus in particular $\mathcal{I}_0^* = \mathcal{I}_0$. Since $x \notin \mathtt{dom}(\Gamma)$, $x \notin \mathtt{dom}(\Gamma_0)$, so that $\Gamma_0 \wedge x : \mathcal{I}_0 = \Gamma_0; x : \mathcal{I}_0$. We then construct the following derivation $\Phi$ :

$$\dfrac{\Phi_{\mathtt{TT}[\![x]\!]} \triangleright \Gamma_u \Vdash u : \mathcal{I}_0 \mid \Delta_u}{\Gamma_0 \wedge \Gamma_u \vdash \mathtt{TT}[\![x]\!][x/u] : \mathcal{U} \mid \Delta_0 \vee \Delta_u}\,(\mathtt{s})$$

We conclude since $\Gamma = \Gamma_0 \wedge \Gamma_u$ and $\Delta = \Delta_0 \vee \Delta_u$.

- If $o = \mathtt{CC}[\![[\alpha]t]\!]\langle\alpha /\!\!/ \alpha'.u\rangle \to \mathtt{CC}[\![[\alpha']tu]\!]\langle\alpha /\!\!/ \alpha'.u\rangle = o'$, with $|\mathtt{CC}[\![[\alpha]t]\!]_\alpha > 1$. Then $\Phi'$ has the following form :

$$\frac{\Phi'_0 \triangleright \Gamma_{\mathtt{cc}} \vdash \mathtt{cc}[\![[\alpha']tu]\!] : \mathcal{A} \mid \Delta_{\mathtt{cc}}; \alpha' : \mathcal{V}_{\alpha'}; \alpha : \mathcal{V}_\alpha \qquad \Theta_u \triangleright \Gamma_u \Vdash u : \mathcal{I}_u \mid \Delta_u}{\Gamma_{\mathtt{cc}} \wedge \Gamma_u \vdash \mathtt{cc}[\![[\alpha']tu]\!]\langle \alpha /\!\!/ \alpha'.u\rangle : \mathcal{A} \mid (\Delta_{\mathtt{cc}}; \alpha' : \mathcal{V}_{\alpha'}) \vee \Delta_u \vee \alpha' : \mathcal{V}} \, (\mathtt{r})$$

where $\mathcal{V}_\alpha = \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L}$, $\mathcal{I}_u = (\wedge_{\ell \in L} \mathcal{I}_\ell^*)^*$, $\mathcal{V} = \vee_{\ell \in L} \mathcal{V}_\ell$, $\Gamma = \Gamma_{\mathtt{cc}} \wedge \Gamma_u$ and $\Delta = (\Delta_{\mathtt{cc}}; \alpha' : \mathcal{V}_{\alpha'}) \vee \Delta_u \vee \alpha' : \mathcal{V} = (\Delta_{\mathtt{cc}} \vee \Delta_u; \alpha' : \mathcal{V}_{\alpha'} \vee \mathcal{V})$ since $\alpha' \notin \mathtt{fn}(u)$ implies $\alpha' \notin \mathtt{dom}(\Delta_u)$. Since $\alpha \in \mathtt{fn}(\mathtt{cc}[\![[\alpha']tu]\!])$, then $L \neq \emptyset$ by Lemma 7.3, so that $\mathcal{I}_u = \wedge_{\ell \in L} \mathcal{I}_\ell^*$.

By Lemma 10.4 applied to $\Phi'_0$, we have $\Gamma_0$, $\Delta'_0, \Phi_0$, $\mathcal{V}_0$, $K \neq \emptyset$, $(\mathcal{I}_k)_{k \in K}$, $(\mathcal{V}_k)_{k \in K}$, $\Gamma'_u, \Delta'_u$, and $\Theta'_u$ such that
- $\Gamma_{\mathtt{cc}} = \Gamma_0 \wedge \Gamma'_u$,
- $\Delta_{\mathtt{cc}}; \alpha : \mathcal{V}_\alpha = \Delta'_0 \vee \Delta'_u$,
- $\mathcal{V}_{\alpha'} = \mathcal{V}_0 \vee_{k \in K} \mathcal{V}_k$,
- $\Phi_0 \triangleright \Gamma_0 \vdash \mathtt{cc}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta'_0$ and
- $\triangleright \Gamma'_u \Vdash u : \wedge_{k \in K} \mathcal{I}_k^* \mid \Delta'_u$

We set $L'' = L \uplus K$, $\Gamma''_u = \Gamma_u \wedge \Gamma'_u$, $\Delta''_u = \Delta_u \vee \Delta'_u$ and $(\mathcal{I}''_u)^* = \mathcal{I}''_u = \wedge_{\ell \in L''} \mathcal{I}_\ell^*$ (indeed, $L'' \supseteq K \neq \emptyset$). By Lemma 7.3, $\alpha \notin \mathtt{dom}(\Delta'_u)$, so that $\Delta'_0 = \Delta_0; \alpha : \mathcal{V}_\alpha$ for some $\Delta_0$ and $\alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta'_0 = \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_\ell \to \mathcal{V}_\ell \rangle_{\ell \in L''}; \Delta_0$ since $\alpha' \notin \mathtt{dom}(\Delta'_0)$. By Lemma 5.1, there is $\Theta''_u \triangleright \Gamma''_u \Vdash u : \mathcal{I}''_u \mid \Delta''_u$. We then construct the following derivation $\Phi$:

$$\frac{\Phi_0 \triangleright \Gamma_0 \vdash \mathtt{cc}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta'_0 \quad \Theta''_u \triangleright \Gamma''_u \Vdash u : \mathcal{I}''_u \mid \Delta''_u}{\Gamma_0 \wedge \Gamma''_u \vdash \mathtt{cc}[\![[\alpha]t]\!]\langle \alpha /\!\!/ \alpha'.u\rangle : \mathcal{A} \mid \alpha' : \mathcal{V}_0 \vee_{\ell \in L''} \mathcal{V}_\ell; \Delta_0 \vee \Delta''_u} \, (\mathtt{r})$$

We conclude since $\Gamma_0 \wedge \Gamma''_u = \Gamma_0 \wedge \Gamma'_u \wedge \Gamma_u = \Gamma_{\mathtt{cc}} \wedge \Gamma_u = \Gamma$, $\Delta_0 \vee \Delta''_u = \Delta_0 \vee \Delta'_u \vee \Delta_u = \Delta_{\mathtt{cc}} \vee \Delta_u$ and $\mathcal{V}_0 \vee_{\ell \in L''} \mathcal{V}_\ell = \mathcal{V}_0 \vee_{k \in K} \mathcal{V}_k \vee_{\ell \in L} \mathcal{V}_\ell = \mathcal{V}_{\alpha'} \vee_{\ell \in L} \mathcal{V}_\ell = \mathcal{V}_{\alpha'} \vee \mathcal{V}$.

- If $o = \mathtt{cc}[\![[\alpha]t]\!]\langle \alpha /\!\!/ \alpha'.u\rangle \to \mathtt{cc}[\![[\alpha']tu]\!] = o'$, with $|\mathtt{cc}[\![[\alpha]t]\!]|_\alpha = 1$, then the derivation $\Phi'$ necessarily ends with the judgment $\Gamma \vdash \mathtt{cc}[\![[\alpha']tu]\!] : \mathcal{A} \mid \Delta_{\mathtt{cc}}; \alpha' : \mathcal{V}$, where $\Delta = \Delta_{\mathtt{cc}}; \alpha' : \mathcal{V}$.

  By Lemma 10.4 applied to $\Phi'$, we have $\Gamma_0$, $\Delta_0$, $\mathcal{V}_0$, $K \neq \emptyset$, $(\mathcal{I}_k)_{k \in K}$, $(\mathcal{V}_k)_{k \in K}$, $\Gamma_u$, $\Delta_u$, and $\Theta_u$ such that
  - $\Gamma = \Gamma_0 \wedge \Gamma_u$,
  - $\Delta_{\mathtt{cc}} = \Delta_0 \vee \Delta_u$,
  - $\mathcal{V} = \mathcal{V}_0 \vee_{k \in K} \mathcal{V}_k$,
  - $\Phi_0 \triangleright \Gamma_0 \vdash \mathtt{cc}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta_0$ and
  - $\Theta_u \triangleright \Gamma_u \Vdash u : \wedge_{k \in K} \mathcal{I}_k^* \mid \Delta_u$

  Notice that $K \neq \emptyset$ implies $(\wedge_{k \in K} \mathcal{I}_k^*)^* = \wedge_{k \in K} \mathcal{I}_k^*$. Moreover, by Lemma 7.3, since $\alpha \notin \mathtt{fn}(\mathtt{cc}[\![[\alpha']tu]\!])$, then $\alpha \notin \mathtt{dom}(\Delta)$, thus $\alpha \notin \mathtt{dom}(\Delta_0)$ and $\alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta_0 = \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K}; \Delta_0$. We then construct $\Phi$ :

$$\frac{\Phi_0 \triangleright \Gamma_0 \vdash \mathtt{cc}[\![[\alpha]t]\!] : \mathcal{A} \mid \alpha' : \mathcal{V}_0; \alpha : \langle \mathcal{I}_k \to \mathcal{V}_k \rangle_{k \in K} \vee \Delta_0 \quad \Theta_u \triangleright \Gamma_u \Vdash u : \wedge_{k \in K} \mathcal{I}_k^* \mid \Delta_u}{\Gamma \vdash \mathtt{cc}[\![[\alpha]t]\!]\langle \alpha /\!\!/ \alpha'.u\rangle : \mathcal{A} \mid (\Delta_0; \alpha' : \mathcal{V}_0) \vee \Delta_u \vee \alpha' : \vee_{k \in K} \mathcal{V}_k} \, (\mathtt{r})$$

  We conclude since $\alpha' \notin \mathtt{fn}(u)$ implies $\alpha' \notin \mathtt{dom}(\Delta_u)$ by Lemma 7.3 so that $(\Delta_0; \alpha' : \mathcal{V}_0) \vee \Delta_u \vee \alpha' : \vee_{k \in K} \mathcal{V}_k = \Delta_0 \vee \Delta_u; \alpha' : \mathcal{V}_0 \vee_{k \in K} \mathcal{V}_k = \Delta_{\mathtt{cc}}; \alpha' : \mathcal{V} = \Delta$ as desired.

□