Math. Struct. in Comp. Science: page 1 of 34.
doi:10.1017/S0960129517000111

A resource aware semantics for a focused intuitionistic calculus

DELIA KESNER[†] and DANIEL VENTURA[‡]

[†]*IRIF, CNRS, Univ. Paris-Diderot, Paris, France*
Email: kesner@pps.univ-paris-diderot.fr
[‡]*INF, Univ. Federal de Goiás, Goiânia, Brazil*
Email: daniel@inf.ufg.br

Received 28 April 2016; revised 23 March 2017

We investigate a new computational interpretation for an intuitionistic focused sequent calculus which is compatible with a resource aware semantics. For that, we associate to Herbelin's syntax a type system based on non-idempotent intersection types, together with a set of reduction rules – inspired from the *substitution at a distance* paradigm – that preserves (and decreases the size of) typing derivations. The non-idempotent approach allows us to use very simple combinatorial arguments, only based on this measure decreasingness, to characterize *linear-head* and *strongly* normalizing terms by means of typability. For the sake of completeness, we also study typability (and the corresponding strong normalization characterization) in the calculus obtained from the former one by projecting the explicit cuts.

1. Introduction

Intuitionistic logic can be expressed in different formal systems such as natural deduction and sequent calculi. Equivalence between these two formal styles has been widely studied (Espírito Santo 2009; Gentzen 1969; Pottinger 1977; Prawitz 1965; Zucker 1974), i.e. every derivation in one system can be encoded into a derivation in the other one. However, this correspondence is not one-to-one, in particular, several *cut-free* proofs in an intuitionistic sequent calculus correspond to the same *normal* natural deduction derivation. This gives rise to restrictions of sequent calculi, the so-called *focused* sequent calculi (Andreoli 1992), which establish a better relationship with natural deduction. Indeed, there is a one-to-one correspondence between cut-free proofs in focused sequent calculi and normal derivations in natural deduction.

In 1994, Herbelin (1995) introduced the $\overline{\lambda}$-calculus, obtained by a computational interpretation of the focused sequent calculus for the minimal intuitionistic logic LJT. In contrast to the usual λ-calculus notation for natural deduction, $\overline{\lambda}$ notation brings head variables to the surface, treats sequences of arguments as lists, and encodes cut typing rules with *explicit cuts*. Its operational semantics is specified by means of a complete set of cut-elimination rules. The calculus is permutation-free and can be used to describe proof-search in pure Prolog and some of its extensions (Miller et al. 1991). The reduction system of the $\overline{\lambda}$-calculus was then extended (Dyckhoff and Urban 2003) with permutation rules, thus showing how to model beta-reduction by giving a natural basis for implementations of functional languages.

Unfortunately, Herbelin's calculus is not compatible with a resource aware semantics, mainly because propagation of explicit cuts w.r.t. the *structure* of $\overline{\lambda}$-terms induces useless duplications of empty resources (c.f. technical discussion in Section 4). This is substantiated when trying to interpret the λ-calculus by means of proof-nets (Girard 1996) or non-idempotent intersection types (pioneered by Gardner (1994), Boudol et al. (1999), Kfoury (1996) and Kfoury and Wells (2004)).

This paper proposes a new computational interpretation for the focused intuitionistic sequent calculus LJT, that we call E-*calculus*, which is compatible with a resource aware semantics. The new calculus keeps Herbelin's syntax but changes the operational semantics of $\overline{\lambda}$ to a resource-controlled interpretation, inspired from the structural lambda-calculus (Accattoli and Kesner 2010), and the linear substitution calculus (Accattoli et al. 2014; Milner 2007). The terms of the E-calculus can be seen as λ-terms with explicit cuts of the form $t[x\backslash u]$, where $[x\backslash u]$ is propagated according to the number of free occurrence of x in t (and not w.r.t. the structure of terms). For the sake of completeness, we also study in the second part of the paper the I-*calculus*, a formalism using full – in contrast to partial – substitution, in which normal forms are exactly the same as those of the E-calculus. An I-reduction step is obtained by projecting E-reduction steps into terms without explicit cuts. In other words, E-reduction implements the meta-level operators of the I-calculus by using a resource aware semantics specified by means of explicit reduction rules. Therefore, the paper gives a self-contained study of computational interpretations based on intuitionistic sequent calculi, completely independent from their isomorphic natural deduction counterparts.

A second contribution of this paper is to provide type systems based on non-idempotent intersection types for both E and I calculi. Intersection types were introduced to give characterizations of strongly β-normalizing terms in the λ-calculus (Coppo and Dezani-Ciancaglini 1978; Krivine 1993; Pottinger 1980); since then, they have been used to characterize termination properties in a broader sense (Coppo et al. 1981), as well as to construct models of the λ-calculus itself (Barendregt et al. 1983). Commonly, intersection types are *idempotent*, i.e. $\sigma \wedge \sigma = \sigma$, but we use here *non-idempotent* types (Boudol et al. 1999; Kfoury 1996; Kfoury and Wells 2004), suitable to obtain *quantitative* information about reduction sequences.

In this paper, we define non-idempotent type systems for both E and I calculi.

The first system, called $\mathcal{H}_{\mathrm{E}}$, characterizes linear-head normalizing terms, i.e. a term t is typable in $\mathcal{H}_{\mathrm{E}}$ if and only if t is linear-head E-normalizing. Linear-head reduction is for terms with explicit cuts what *head reduction* is for λ-terms. Moreover, linear-head reduction cannot be simply expressed as a strategy of β-reduction, the reason being that β-reduction implements *full* substitution, while linear-head reduction only uses a *partial*/*linear* notion of substitution.

The second system, called $\mathcal{S}_{\mathrm{E}}$, characterizes strongly normalizing terms, i.e. a term t is typable in $\mathcal{S}_{\mathrm{E}}$ if and only if t is strongly E-normalizing. Because of the non-idempotent approach, the characterization proofs mentioned above use simple combinatorial arguments, specified by a *Weighted Subject Reduction property*, and make no use of reducibility techniques, as required in the idempotent case.

The third system, obtained by restricting $\mathcal{S}_{\mathtt{E}}$ to pure terms (i.e. terms without explicit cuts) also characterizes strong I-normalization, as expected. The implication 'Typability implies Normalization' is obtained from the corresponding result for E-terms mentioned above, by using an appropriate projection lemma. The converse implication 'Normalization implies Typability' is developed independently, i.e. without using the corresponding result for the E-calculus (which would need to prove that the E-calculus preserves I-normalization).

Some related work: In the last years, there has been a growing interest in non-idempotent intersection types. The relation between the size of a non-idempotent intersection typing derivation and the head/weak-normalization execution time of λ-terms by means of abstract machines was established by de Carvalho (2007). Non-idempotence is used to reason about the longest reduction sequence of strongly normalizing terms in both the lambda-calculus (Bernadet and Lengrand 2011; De Benedetti and Ronchi Della Rocca 2013) and in different lambda-calculi with explicit substitutions (Bernadet and Lengrand 2013; Kesner and Ventura 2014b). Non-idempotent types also appear in the study of needed reduction (Gardner 1994), linearization of the lambda-calculus (Kfoury 1996), type inference (Kfoury and Wells 2004; Neergaard and Mairson 2004), different characterizations of solvability (Pagani and Ronchi Della Rocca 2011) and verification of higher order programs (Ong and Ramsay 2011). While the inhabitation problem for intersection types is known to be undecidable in the idempotent case (Urzyczyn 1999), decidability was recently proved (Bucciarelli et al. 2014) through a sound and complete algorithm in the non-idempotent case. Concerning the use of *idempotent* intersection types for focused intuitionistic sequent calculi, two different papers (Espírito Santo et al. 2012; Ghilezan et al. 2011) provide characterizations of strongly normalizing terms by means of typability, but none of them use quantitative information about reduction, as presented in this paper. Moreover, in contrast to Ghilezan et al. (2011), which is based on explicit control operators for weakening and contraction, we keep the simple, original syntax of Herbelin.

The work presented in this paper originates from a first computational interpretation of LJT appearing in an unpublished technical report (Kesner and Ventura 2014a). The approach in Kesner and Ventura (2014a) formulates the typing rules by introducing witness derivations *everywhere*, so that it is too costly and resource demanding (see the discussion at the end of Section 11). The type systems in this paper *only* require *witness* derivations for potentially erasable arguments of functions and cuts. As a consequence, the upper bound for the longest reduction sequence of a strongly normalizing term obtained in this paper, represented just by the size of a typing derivation, is tighter than the one in Kesner and Ventura (2014a).

Structure of the paper: After giving some general notions of rewriting used all along the paper (Section 2), the explicit E-calculus is introduced in Section 3. Two different typing systems for the E-calculus are introduced in Section 4 followed by its main properties (Section 5). Characterizations of linear-head and strong E-normalization are studied in Sections 6 and 7, respectively. Section 8 presents the syntax and the operational semantics of the I-calculus, obtained by projecting E into a pure grammar without explicit cuts. Section 9 introduces non-idempotent types for the I-calculus and Section 10 studies

a characterization of I-normalization by means of typability. Finally, we conclude in Section 11.

This paper is an extended version of Kesner and Ventura (2015).

2. General notions of rewriting

This section introduces some general notions of rewriting used all along the paper for different reduction relations.

A reduction relation $\to_{\mathcal{R}}$ on a set $\mathcal{O}$ is a subset of $\mathcal{O} \times \mathcal{O}$. We denote by $\to^*_{\mathcal{R}}$ (resp. $\to^+_{\mathcal{R}}$) the **reflexive–transitive** (resp. **transitive**) closure of a given reduction relation $\to_{\mathcal{R}}$. An object $o \in \mathcal{O}$ is in **$\mathcal{R}$-normal form**, or **$\mathcal{R}$-nf**, written $o \in \mathtt{NF}(\mathcal{R})$, if there is no o' such that $o \to_{\mathcal{R}} o'$. Similarly, o **has an $\mathcal{R}$-normal form**, if there exists $o' \in \mathtt{NF}(\mathcal{R})$ such that $o \to^*_{\mathcal{R}} o'$. The reduction relation $\mathcal{R}$ is **confluent** if and only if for all objects o_1, o_2, o_3 such that $o_1 \to^*_{\mathcal{R}} o_2$ and $o_1 \to^*_{\mathcal{R}} o_3$, there is o_4 being able to close the diagram, i.e. $o_2 \to^*_{\mathcal{R}} o_4$ and $o_3 \to^*_{\mathcal{R}} o_4$. An object o is **strongly $\mathcal{R}$-normalizing**, written $o \in \mathcal{SN}(\mathcal{R})$, if there is no infinite $\mathcal{R}$-reduction sequence starting at o, and o is **$\mathcal{R}$-finitely branching** if the set $\{o' \mid o \to_{\mathcal{R}} o'\}$ is finite. If an object o is $\mathcal{R}$-strongly normalizing and $\mathcal{R}$-finitely branching, then the **depth of** o, written $\eta_{\mathcal{R}}(o)$, is the maximal length of an $\mathcal{R}$-reduction sequence starting at o.

3. The E-calculus

This section introduces the syntax and the operational semantics of the E-calculus. The term language follows from Herbelin (1995), while the reduction rules aim to give a resource aware semantics based on the *substitution at a distance* paradigm (Accattoli et al. 2014; Milner 2007). Given a countable infinite set of symbols $x, y, z, \ldots$, we define the following three syntactic categories:

$$\begin{array}{lll} \textbf{(E-objects)} & o, p & ::= t \mid l \\ \textbf{(E-terms)} & t, u & ::= x l \mid t l \mid \lambda x.t \mid t[x\backslash t] \\ \textbf{(E-lists)} & l, m & ::= \epsilon \mid t ; l \end{array}$$

The **empty list** is denoted by ϵ and the construction $[x\backslash u]$ is said to be an **explicit cut**. Remark that the symbol x alone is not an object of the syntax (term variables in natural deduction style are encoded by $x\epsilon$), and cuts do not apply to lists, but only to terms, i.e. $l[x\backslash u]$ is not in the grammar. We may write $t l_1 \ldots l_n$ for $(\ldots(t l_1)\ldots l_n)$ and x_ϵ for $x\epsilon$.

The **size** of an object o, written $|o|$, is defined by induction as follows:

$$\begin{array}{llllll} |xl| & := |l| + 1 & & |\epsilon| & := 1 \\ |tl| & := |t| + |l| & & |t;l| & := |t| + |l| + 1 \\ |\lambda x.t| & := |t| + 1 & & |t[x\backslash u]| & := |t| + |u| + 1 \end{array}$$

The notions of **free** and **bound** variables are defined as usual, in particular,

$$\begin{array}{llllll} \mathtt{fv}(xl) & := \{x\} \cup \mathtt{fv}(l) & & \mathtt{fv}(\epsilon) & := \varnothing \\ \mathtt{fv}(tl) & := \mathtt{fv}(t) \cup \mathtt{fv}(l) & & \mathtt{fv}(t;l) & := \mathtt{fv}(t) \cup \mathtt{fv}(l) \\ \mathtt{fv}(\lambda x.t) & := \mathtt{fv}(t) \setminus \{x\} & & \mathtt{fv}(t[x\backslash u]) & := (\mathtt{fv}(t) \setminus \{x\}) \cup \mathtt{fv}(u) \end{array}$$

The set of **positions** of o, written $\mathtt{pos}(o)$, is a finite language over $\{0,1\}$ inductively defined as follows: $\varepsilon \in \mathtt{pos}(o)$ for every o (the root position); $0p \in \mathtt{pos}(\lambda x.t)$ if $p \in \mathtt{pos}(t)$; $0p \in \mathtt{pos}(xl)$ if $p \in \mathtt{pos}(l)$; $0p \in \mathtt{pos}(tl)$ (resp. $\mathtt{pos}(t;l)$ and $\mathtt{pos}(t[x\backslash u])$) if $p \in \mathtt{pos}(t)$; $1p \in \mathtt{pos}(tl)$ (resp. $\mathtt{pos}(t;l)$) if $p \in \mathtt{pos}(l)$; $1p \in \mathtt{pos}(t[x\backslash u])$ if $p \in \mathtt{pos}(u)$. The **subterm of** t **at position** p is written $t|_p$ and defined as expected.

The **number of free occurrences of** x **in** o is written $|o|_x$. We work with the standard notions of **α-conversion** (i.e. renaming of bound variables for abstractions and cuts), and Barendregt's convention (Barendregt 1984).

We also consider different categories of $\mathtt{E}$-contexts:

$$\begin{array}{lll}
\textbf{(E-head cut contexts)} & \mathtt{L} & ::= \Box \mid \mathtt{L}[x\backslash t] \\
\textbf{(E-object contexts)} & \mathtt{O},\mathtt{P} & ::= \mathtt{C} \mid \mathtt{V} \\
\textbf{(E-term contexts)} & \mathtt{C},\mathtt{D} & ::= \Box \mid x\mathtt{V} \mid \mathtt{C}l \mid \lambda y.\mathtt{C} \mid \mathtt{C}[y\backslash u] \mid t[y\backslash \mathtt{C}] \mid t\mathtt{V} \\
\textbf{(E-list contexts)} & \mathtt{V},\mathtt{U} & ::= \mathtt{C};l \mid t;\mathtt{V}
\end{array}$$

When the replacement of the hole of $\mathtt{O}$ by the object o is well defined (i.e. gives an object), then we denote it by $\mathtt{O}[o]$. Similarly, $\mathtt{L}[t]$ denotes the term obtained by replacing the hole of $\mathtt{L}$ by the term t. We write $\mathtt{C}^x$ for a context $\mathtt{C}$ which does not capture the free variable x, i.e. there are no abstractions or explicit cuts in the context that binds the variable x. For instance, $C = \lambda y.\Box$ can be specified as $\mathtt{C}^x$ while $C = \lambda x.\Box$ cannot. In order to emphasize this particular property, we write $\mathtt{C}^x[\![t]\!]$ instead of $\mathtt{C}^x[t]$, and we may omit x whenever it is clear from the context.

The **reduction relation** $\to_{\mathtt{E}}$ is defined as the closure by contexts $\mathtt{O}$ of the following rewriting rules:

$$\begin{array}{llllll}
\mathtt{L}[\lambda x.t]\epsilon & \mapsto_{\mathtt{dB}_{\mathtt{nil}}} & \mathtt{L}[\lambda x.t] & \mathtt{L}[xl]m & \mapsto_{@_{\mathtt{var}}} & \mathtt{L}[x(l@m)] \\
\mathtt{L}[\lambda x.t](u;l) & \mapsto_{\mathtt{dB}_{\mathtt{cons}}} & \mathtt{L}[t[x\backslash u]l] & \mathtt{L}[tl]m & \mapsto_{@_{\mathtt{app}}} & \mathtt{L}[t(l@m)]
\end{array}$$

$$\begin{array}{llll}
\mathtt{C}^x[\![x\,l]\!][x\backslash u] & \mapsto_{\mathtt{c}} & \mathtt{C}^x[\![u\,l]\!][x\backslash u] & \text{if } |\mathtt{C}^x[\![x\,l]\!]|_x > 1 \\
\mathtt{C}^x[\![x\,l]\!][x\backslash u] & \mapsto_{\mathtt{d}} & \mathtt{C}^x[\![u\,l]\!] & \text{if } |\mathtt{C}^x[\![x\,l]\!]|_x = 1 \\
t[x\backslash u] & \mapsto_{\mathtt{w}} & t & \text{if } |t|_x = 0
\end{array}$$

where the operation $\_@\_$ is defined by the following two equations:

$$\epsilon@l := l \qquad\qquad (u;l)@m := u;(l@m)$$

An example of reduction sequence is

$$(\lambda x.x(x_\epsilon;\epsilon))(u;\epsilon) \to_{\mathtt{dB}_{\mathtt{cons}}} x(x_\epsilon;\epsilon)[x\backslash u]\epsilon \to_{@_{\mathtt{var}}} x(x_\epsilon;\epsilon)[x\backslash u] \to_{\mathtt{c}} x(u\epsilon;\epsilon)[x\backslash u] \to_{\mathtt{d}} u(u\epsilon;\epsilon)$$

The reduction relation $\to_{\mathtt{E}}$ can also be refined. For every rewrite rule $\mapsto_X$, we write $\to_X$ for the closure of $\mapsto_X$ by all contexts $\mathtt{O}$. We define $\mathtt{B@} := \{\mathtt{dB}_{\mathtt{nil}}, \mathtt{dB}_{\mathtt{cons}}, @_{\mathtt{var}}, @_{\mathtt{app}}\}$ and $\to_{\mathtt{B@}} := \bigcup_{X\in\mathtt{B@}} \to_X$. The **non-erasing** reduction relation $\to_{\mathtt{E\backslash w}}$ is given by $\to_{\mathtt{B@}\cup\{\mathtt{c},\mathtt{d}\}}$, i.e. $\to_{\mathtt{E\backslash w}} = \to_{\mathtt{E}} \setminus \to_{\mathtt{w}}$; this relation plays a key role in the characterization of strongly $\mathtt{E}$-normalizing terms (c.f. Section 7).

There are many differences between our reduction rules and those in Herbelin (1995). First of all, the use of the meta-operation @ for concatenating lists in the rules $\mapsto_{@_{\mathtt{var}}}$ and $\mapsto_{@_{\mathtt{app}}}$ replaces the explicit concatenation rules in Herbelin (1995). This is particularly convenient since we only reduce terms, even if these terms occur inside lists, so that

the proofs are simpler/shorter because there are less rules and only of one kind. A major difference with Herbelin (1995) is the use of rules *at a distance*, specified by means of (term and list) contexts, where the propagation of cuts is not performed by structural induction on terms, since they are consumed according to the multiplicity of their corresponding variables. As a consequence, the behaviour of the explicit cut operator is specified by a resource aware semantics, thus preventing the useless duplication of empty resources, which happens in Herbelin (1995) when using reduction steps of the form $(tl)[x\backslash u] \to t[x\backslash u]l[x\backslash u]$, where $|tl|_x = 0$. This is particularly unsuitable when considering non-idempotent types (c.f. the discussion at the end of Section 4). In contrast to other calculi at a distance which only contains $\mathtt{w}$ and $\mathtt{c}$-rules, for example the linear substitution calculus (Accattoli et al. 2014; Milner 2007), we also consider here a dereliction rule $\mathtt{d}$. This is appropriate to obtain a *weighted* subject reduction property relative to our typing system (c.f. Section 4), which would fail for the alternative rewriting rule $\mathtt{C}[\![x\,l]\!][x\backslash u] \mapsto_{\mathtt{c}} \mathtt{C}[\![u\,l]\!][x\backslash u]$ when $|\mathtt{C}[\![x\,l]\!]|_x = 1$.

4. Typing systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$ for E-terms

This section introduces the typing systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$ for the E-calculus. Given a countable infinite set of *base types* $\alpha, \beta, \gamma, \ldots$, we consider **types** and **multiset types** defined as follows:

(types)	τ, σ, ρ	$::=$	$\alpha \mid \mathcal{M} \supset \tau$
(multiset types)	$\mathcal{M}$	$::=$	$\{\!\{\tau_i\}\!\}_{i\in I}$ where I is a finite set.

Our types are *strict* (Coppo and Dezani-Ciancaglini 1980; van Bakel 1992), i.e. the type on the right-hand side of a functional type is never a multiset. We also make use of usual notations for multisets, as in de Carvalho (2007), so that $\{\!\{\ \}\!\}$ denotes the empty multiset, and $\{\!\{\sigma, \sigma, \tau\}\!\}$ must be understood as $\sigma \wedge \sigma \wedge \tau$, where the symbol $\wedge$ enjoys commutativity and associativity but not idempotence, i.e. $\sigma \wedge \sigma$ is not equal to σ.

Type assignments, written Γ, Δ, are functions from variables to multiset types (i.e. sets of pairs), assigning the empty multiset to all but a finite set of variables. The domain of Γ is given by $\mathtt{dom}(\Gamma) := \{x \mid \Gamma(x) \neq \{\!\{\ \}\!\}\}$. The **intersection of type assignments**, written $\Gamma + \Delta$, is defined by $(\Gamma + \Delta)(x) := \Gamma(x) + \Delta(x)$, where the symbol $+$ denotes multiset union. Hence, $\mathtt{dom}(\Gamma + \Delta) = \mathtt{dom}(\Gamma) \cup \mathtt{dom}(\Delta)$. Thus, for example, $\{x : \{\!\{\sigma\}\!\}, y : \{\!\{\{\!\{\ \}\!\} \supset \tau\}\!\}\} + \{z : \{\!\{\tau\}\!\}, x : \{\!\{\{\!\{\sigma\}\!\} \supset \sigma\}\!\}\} = \{x : \{\!\{\sigma, \{\!\{\sigma\}\!\} \supset \sigma\}\!\}, y : \{\!\{\{\!\{\ \}\!\} \supset \tau\}\!\}, z : \{\!\{\tau\}\!\}\}$.

We write $\Gamma \setminus\!\setminus x$ for the assignment $(\Gamma \setminus\!\setminus x)(x) = \{\!\{\ \}\!\}$ and $(\Gamma \setminus\!\setminus x)(y) = \Gamma(y)$ if $y \neq x$. When $\mathtt{dom}(\Gamma)$ and $\mathtt{dom}(\Delta)$ are disjoint, we use $\Gamma; \Delta$ instead of $\Gamma + \Delta$, and we write $x : \{\!\{\sigma_i\}\!\}_{i\in I}; \Gamma$, even when $I = \varnothing$, for the assignment $(x : \{\!\{\sigma_i\}\!\}_{i\in I}; \Gamma)(x) = \{\!\{\sigma_i\}\!\}_{i\in I}$ and $(x : \{\!\{\sigma_i\}\!\}_{i\in I}; \Gamma)(y) = \Gamma(y)$ if $y \neq x$.

The symbol $\_$ is called the **empty stoup**. A **stoup** Σ is either a type σ or the empty stoup. **Type environments** are pairs of the form $\Gamma \mid \Sigma$, where Γ is a type assignment and Σ is a stoup. **Type judgments** are triples of the form $\Gamma \mid \Sigma \vdash o : \tau$, where $\Gamma \mid \Sigma$ is a type environment, o is an object, and τ a type. The $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$ type systems for the E-calculus are given in Figures 1 and 2, respectively; both deriving type judgements of the form $\Gamma \mid \_ \vdash t : \tau$ and $\Gamma \mid \sigma \vdash l : \tau$, where t is a term and l is a list. We write $\Gamma \mid \Sigma \vdash_X o : \tau$ or $\Phi \rhd_X \Gamma \mid \Sigma \vdash o : \tau$ to denote **derivations** in the system $X \in \{\mathcal{H}_{\mathrm{E}}, \mathcal{S}_{\mathrm{E}}\}$.

$$\frac{}{\emptyset \mid \tau \vdash \epsilon : \tau}\ (\mathtt{ax}) \qquad \frac{\Gamma \mid \_ \vdash t : \tau}{\Gamma \setminus\setminus x \mid \_ \vdash \lambda x.t : \Gamma(x) \supset \tau}\ (\supset \mathtt{r})$$

$$\frac{\Gamma \mid \sigma \vdash l : \tau}{\Gamma + \{x{:}\{\!\{\sigma\}\!\}\} \mid \_ \vdash xl : \tau}\ (\mathtt{hlist}) \qquad \frac{\Gamma \mid \_ \vdash t : \sigma \qquad \Delta \mid \sigma \vdash l : \tau}{\Gamma + \Delta \mid \_ \vdash tl : \tau}\ (\mathtt{app})$$

$$\frac{(\Delta_i \mid \_ \vdash t : \sigma_i)_{i \in I} \qquad \Gamma \mid \sigma \vdash l : \tau}{\Gamma +_{i \in I} \Delta_i \mid \{\!\{\sigma_i\}\!\}_{i \in I} \supset \sigma \vdash t; l : \tau}\ (\supset \mathtt{l})$$

$$\frac{(\Delta_i \mid \_ \vdash u : \sigma_i)_{i \in I} \qquad x{:}\{\!\{\sigma_i\}\!\}_{i \in I}; \Gamma \mid \_ \vdash t : \tau}{\Gamma +_{i \in I} \Delta_i \mid \_ \vdash t[x \backslash u] : \tau}\ (\mathtt{cut})$$

Fig. 1. The type system $\mathcal{H}_{\mathrm{E}}$ for the E-calculus.

Example 4.1. The following is a derivation for the term $y(z\epsilon;\epsilon)$ in system $\mathcal{H}_{\mathrm{E}}$:

$$\dfrac{\dfrac{\dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}\ (\mathtt{ax})}{\varnothing \mid \{\!\{\ \}\!\} \supset \sigma \vdash z\epsilon; \epsilon : \sigma}\ (\supset \mathtt{l})}{y : \{\!\{\{\!\{\ \}\!\} \supset \sigma\}\!\} \mid \_ \vdash y(z\epsilon;\epsilon) : \sigma}\ (\mathtt{hlist})$$

Let us call it $\Phi_{y(z\epsilon;\epsilon)}$ for further references.

Example 4.2. The following is a derivation, that we call $\Phi_{x(x\epsilon;\epsilon)}$, in system $\mathcal{S}_{\mathrm{E}}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}\ (\mathtt{ax})}{x : \{\!\{\sigma\}\!\} \mid \_ \vdash x\epsilon : \sigma}(\mathtt{hlist}) \qquad \dfrac{}{\varnothing \mid \tau \vdash \epsilon : \tau}\ (\mathtt{ax})}{x : \{\!\{\sigma\}\!\} \mid \{\!\{\sigma\}\!\} \supset \tau \vdash x\epsilon; \epsilon : \tau}(\supset \mathtt{l}_{\in})}{x : \{\!\{\sigma, \{\!\{\sigma\}\!\} \supset \tau\}\!\} \mid \_ \vdash x(x\epsilon;\epsilon) : \tau}\ (\mathtt{hlist})$$

The **hlist-size** of the type derivation Φ is a positive natural number written $\mathtt{sz2}(\Phi)$ which denotes the size of Φ where every node `hlist` counts 2. Intuitively, the node `hlist` counts 2 because it corresponds, in the standard sequent calculus, to an axiom rule followed by a contraction. For instance, the derivation $\Phi_{y(z\epsilon;\epsilon)}$ in Example 4.1 has `hlist`-size 4 while $\Phi_{x(x\epsilon;\epsilon)}$ in Example 4.2 has `hlist`-size 7.

Notice that both $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$ are *goal-directed*, i.e. for each type judgement of the form $\Gamma \mid \Sigma \vdash o : \tau$ there is a *unique* typing rule whose conclusion matches the type judgement. For instance, in system $\mathcal{S}_{\mathrm{E}}$, there are two different rules to type a list $t;l$, but the type in the stoup Σ discriminates between them. There is a similar distinction in system $\mathcal{S}_{\mathrm{E}}$ for terms of the form $t[x\backslash u]$ depending on whether x is a free variable of t or not. A consequence of this property is that statements usually known as generation lemmas (c.f. Barendregt et al. 1983; Lengrand et al. 2004) are straightforward in both systems.

The (app) typing rule is the *head-cut* rule in the underlying logical system; similarly, (cut), ($\mathtt{cut}_{\notin}$) and ($\mathtt{cut}_{\in}$) give an interpretation of the so-called *mid-cut* (Herbelin 1995). The type derivation for t (resp. for u) in rule ($\supset \mathtt{l}_{\notin}$) (resp. ($\mathtt{cut}_{\notin}$)) is called a **witness** derivation, and turns out to be essential to guarantee strong normalization of the whole typed term $t;l$ (resp. $t[x\backslash u]$).

Typing Rules $\{(\mathtt{ax}), (\mathtt{hlist}), (\mathtt{app}), (\supset \mathtt{r})\}$ plus

$$\frac{\Gamma \mid \_ \vdash t : \rho \qquad \Delta \mid \tau \vdash l : \sigma}{\Delta + \Gamma \mid \{\!\!\{\ \}\!\!\} \supset \tau \vdash t; l : \sigma}\ (\supset \mathtt{l}_{\notin})$$

$$\frac{(\Gamma_j \mid \_ \vdash t : \tau_j)_{j\in J} \quad J \neq \emptyset \qquad \Delta \mid \tau \vdash l : \sigma}{\Delta +_{j\in J} \Gamma_j \mid \{\!\!\{\tau_j\}\!\!\}_{j\in J} \supset \tau \vdash t; l : \sigma}\ (\supset \mathtt{l}_{\in})$$

$$\frac{\Delta \mid \_ \vdash u : \sigma \qquad \Gamma \mid \_ \vdash t : \tau \qquad x \notin \mathtt{dom}(\Gamma)}{\Gamma + \Delta \mid \_ \vdash t[x\backslash u] : \tau}\ (\mathtt{cut}_{\notin})$$

$$\frac{(\Delta_j \mid \_ \vdash u : \sigma_j)_{j\in J} \qquad J \neq \emptyset \qquad x : \{\!\!\{\sigma_j\}\!\!\}_{j\in J}; \Gamma \mid \_ \vdash t : \tau}{\Gamma +_{j\in J} \Delta_j \mid \_ \vdash t[x\backslash u] : \tau}\ (\mathtt{cut}_{\in})$$

Fig. 2. The type system $\mathcal{S}_{\mathrm{E}}$ for the E-calculus.

Observe that the case $I = \emptyset$ in rules $(\supset \mathtt{l})$ and $(\mathtt{cut})$ of system $\mathcal{H}_{\mathrm{E}}$ gives

$$\frac{\Gamma \mid \sigma \vdash l : \tau}{\Gamma \mid \{\!\!\{\ \}\!\!\} \supset \sigma \vdash u; l : \tau}\ (\supset \mathtt{l}) \qquad \frac{\Gamma \mid \_ \vdash t : \tau}{\Gamma \mid \_ \vdash t[x\backslash u] : \tau}\ (\mathtt{cut})$$

where u is *any* term. Indeed, non-terminating terms like $t(\Omega; l)$ or $t[x\backslash\Omega]$ are typable in system $\mathcal{H}_{\mathrm{E}}$, for $\Omega = (\lambda x.xx_{\epsilon})\lambda x.xx_{\epsilon}$. For a further discussion on the need of witness in system $\mathcal{S}_{\mathrm{E}}$, see Example 5.5 in Section 5.

The rules $(\supset \mathtt{l}_{\notin})$ and $(\supset \mathtt{l}_{\in})$ (resp. $(\mathtt{cut}_{\notin})$ and $(\mathtt{cut}_{\in})$) in system $\mathcal{S}_{\mathrm{E}}$ can be specified by means of a unique typing rule $(\supset \mathtt{l}_{\mathcal{S}})$ (resp. $(\mathtt{cut}_{\mathcal{S}})$), usually used in the proofs in order to save some space. They have the following form:

$$\frac{(\Gamma_j \mid \_ \vdash t : \tau_j)_{j\in J} \quad \Delta \mid \tau \vdash l : \sigma}{\Delta +_{j\in J} \Gamma_j \mid \{\!\!\{\tau_i\}\!\!\}_{i\in I} \supset \tau \vdash t; l : \sigma}\ (\supset \mathtt{l}_{\mathcal{S}}) \qquad \frac{(\Delta_j \mid \_ \vdash u : \sigma_j)_{j\in J} \quad x : \{\!\!\{\sigma_i\}\!\!\}_{i\in I}; \Gamma \mid \_ \vdash t : \tau}{\Gamma +_{j\in J} \Delta_j \mid \_ \vdash t[x\backslash u] : \tau}\ (\mathtt{cut}_{\mathcal{S}})$$

where $(I = \emptyset \Rightarrow |J| = 1)$ and $(I \neq \emptyset \Rightarrow I = J)$.

Both type systems are *relevant* (Damiani and Giannini 1994), i.e. typing environments only contain the consumed premises.

Lemma 4.1 (Relevance).

1 . If $\Gamma \mid \Sigma \vdash_{\mathcal{H}_{\mathrm{E}}} o : \tau$, then $\mathtt{dom}(\Gamma) \subseteq \mathtt{fv}(o)$.
2 . If $\Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathrm{E}}} o : \tau$, then $\mathtt{dom}(\Gamma) = \mathtt{fv}(o)$.

Proof. By induction on typing derivations. □

Moreover, in contrast to Barendregt et al. (1983) and Bernadet and Lengrand (2013), no subtyping relation is needed for abstractions and/or applications in system $\mathcal{S}_{\mathrm{E}}$.

5. Properties of systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$

This section proves some fundamental properties of the systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$, namely, that any typing is preserved by reduction and anti-reduction, i.e. by transforming/rewriting an object *forward* (Section 5.2) and *backward* (Section 5.1). The forward properties are key

to obtain 'typability implies normalization,' while the backward ones are used to prove 'normalization implies typability.'

Since the developments of the forward (resp. backward) proofs for systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$ are quite similar, we present them in a factorized form. In particular, we start by introducing the notion of typed occurrence of a redex in a general framework (for any type system X), the concept being especially important to state the forward properties of system $\mathcal{H}_{\mathrm{E}}$.

In order to understand which are the redex occurrences actually constrained by a type system, i.e. the reducible subterms that verify some particular type specification, let us consider a derivation $\Phi \triangleright \Gamma \mid \Sigma \vdash_X o : \tau$, for $X \in \{\mathcal{H}_{\mathrm{E}}, \mathcal{S}_{\mathrm{E}}\}$. A position $p \in \mathtt{pos}(o)$ is a **typed occurrence** or **T-occurrence** of the object o in the derivation Φ (of system X) if either $p = \varepsilon$, or $p = ip'$ $(i = 0, 1)$ and $p' \in \mathtt{pos}(o|_i)$ is a typed occurrence of $o|_i$ in *some* of their corresponding subderivations of Φ. Intuitively, a typed occurrence indicates a subterm of o which has some typing information in Φ.

Example 5.1. Let consider the following $\mathcal{H}_{\mathrm{E}}$-derivation, where the notations x_ϵ, z_ϵ are respectively used to abbreviate the terms $x\epsilon$ and $z\epsilon$:

$$\Phi_{x_\epsilon} := \dfrac{\dfrac{}{\varnothing \mid \{\!\{\sigma,\sigma\}\!\} \supset \tau \vdash \epsilon : \{\!\{\sigma,\sigma\}\!\} \supset \tau}\,(\mathtt{ax})}{x : \{\!\{\{\!\{\sigma,\sigma\}\!\} \supset \tau\}\!\} \mid \_ \vdash x_\epsilon : \{\!\{\sigma,\sigma\}\!\} \supset \tau}\,(\mathtt{hlist})$$

Similarly, consider a derivation $\Phi_{z_\epsilon} \triangleright z : \{\!\{\varphi\}\!\} \mid \_ \vdash z_\epsilon : \varphi$. Let $t = y(z_\epsilon ; \epsilon)$ be a term and Φ_t, $\Phi_{t'}$ two possible derivations:

$$\Phi_t := \dfrac{\dfrac{\Phi_{z_\epsilon} \qquad \dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}\,(\mathtt{ax})}{z : \{\!\{\varphi\}\!\} \mid \{\!\{\varphi\}\!\} \supset \sigma \vdash z_\epsilon ; \epsilon : \sigma}\,(\supset \mathtt{l})}{z : \{\!\{\varphi\}\!\}, y : \{\!\{\{\!\{\varphi\}\!\} \supset \sigma\}\!\} \mid \_ \vdash y(z_\epsilon ; \epsilon) : \sigma}\,(\mathtt{hlist}) \qquad \Phi'_t := \dfrac{\dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}\,(\mathtt{ax})}{\dfrac{\varnothing \mid \{\!\{\,\}\!\} \supset \sigma \vdash z_\epsilon ; \epsilon : \sigma}{y : \{\!\{\{\!\{\,\}\!\} \supset \sigma\}\!\} \mid \_ \vdash y(z_\epsilon ; \epsilon) : \sigma}\,(\mathtt{hlist})}\,(\supset \mathtt{l})$$

The positions ε, 0 and 01 of $y(z_\epsilon ; \epsilon)$ are T-occurrences in both Φ_t and Φ'_t while the position 00 is a T-occurrence only in Φ_t. Thus, given a derivation Φ of the form:

$$\dfrac{\Phi_{x_\epsilon} \qquad \dfrac{\Phi_t \qquad \Phi'_t \qquad \dfrac{}{\varnothing \mid \tau \vdash \epsilon : \tau}\,(\mathtt{ax})}{z : \{\!\{\varphi\}\!\}, y : \{\!\{\{\!\{\varphi\}\!\} \supset \sigma, \{\!\{\,\}\!\} \supset \sigma\}\!\} \mid \{\!\{\sigma,\sigma\}\!\} \supset \tau \vdash t ; \epsilon : \tau}\,(\supset \mathtt{l})}{x : \{\!\{\{\!\{\sigma,\sigma\}\!\} \supset \tau\}\!\}, z : \{\!\{\varphi\}\!\}, y : \{\!\{\{\!\{\varphi\}\!\} \supset \sigma, \{\!\{\,\}\!\} \supset \sigma\}\!\} \mid \_ \vdash x_\epsilon(t ; \epsilon) : \tau}\,(\mathtt{app})$$

and a derivation Φ' of the form:

$$\dfrac{\Phi_{x_\epsilon} \qquad \dfrac{\Phi'_t \qquad \Phi'_t \qquad \dfrac{}{\varnothing \mid \tau \vdash \epsilon : \tau}\,(\mathtt{ax})}{y : \{\!\{\{\!\{\,\}\!\} \supset \sigma, \{\!\{\,\}\!\} \supset \sigma\}\!\} \mid \{\!\{\sigma,\sigma\}\!\} \supset \tau \vdash t ; \epsilon : \tau}\,(\supset \mathtt{l})}{x : \{\!\{\{\!\{\sigma,\sigma\}\!\} \supset \tau\}\!\}, y : \{\!\{\{\!\{\,\}\!\} \supset \sigma, \{\!\{\,\}\!\} \supset \sigma\}\!\} \mid \_ \vdash x_\epsilon(t ; \epsilon) : \tau}\,(\mathtt{app})$$

we have that ε, 0, 1, 00, 10, 11 and 101 are T-occurrences of $x_\epsilon(t ; \epsilon)$ in Φ and Φ', while 100 is a T-occurrence in Φ but not in Φ'.

The notion of *redex occurrence* in calculi with explicit substitutions at a distance (Accattoli et al. 2014) is more subtle than the one in standard rewriting, simply because one unique term may give rise to different reduction steps at the root, e.g. given $t = (x(x\epsilon ; \epsilon))[x \backslash u]$ we have $t \to_{\mathtt{c}} (u(x\epsilon ; \epsilon))[x \backslash u]$ as well as $t \to_{\mathtt{c}} (x(u\epsilon ; \epsilon))[x \backslash u]$. Thus,

a position $p \in \mathtt{pos}(o)$ is said to be a **X-redex occurrence of** o, for $X \in \mathtt{B@} \cup \{\mathtt{w}\}$, if $t|_p$ has the form of the left-hand side of the rewriting rule $\mapsto_X$; and $p \in \mathtt{pos}(o)$ is a X-redex occurrence of o, for $X \in \{\mathtt{c}, \mathtt{d}\}$, if $p = p_1 p_2$, and $t|_{p_1} = \mathtt{C}^x[\![x\, l]\!][x \backslash u]$ and $t|_p = xl$. For example 00 and 000 are both c-redex occurrences of the term $\lambda z.(x(x\epsilon;\epsilon))[x \backslash u]$, where, in both cases, $p_1 = 0$. Note that, while p indicates which occurence of x is the target of the cut, the position p_1 is uniquely determined by the position of the cut itself.

A redex occurrence of o which is also a typed occurrence of o in Φ is said to be a **redex typed occurrence** or **redex T-occurrence** of o in Φ. Remark that, given any object o, and any derivation Φ for o in system $\mathcal{S}_{\mathtt{E}}$, every occurrence of o is a T-occurrence of o in Φ. However, in system $\mathcal{H}_{\mathtt{E}}$ this is not the case as illustrated in Example 5.1 above.

5.1. *The 'Forward' properties*

This section shows preservation of any typing by reduction, i.e. by transforming/rewriting an object *forward*. In contrast to classical *subject reduction* properties for non-quantitative typing systems, our preservation property can be *weighted* (Lemma 5.3), i.e. if Φ is a type derivation for t and t reduces to t', then t' is not only typable, but it is typable by a derivation smaller than Φ. This is crucial to obtain 'typablity implies normalization' properties which do not need any reducibility argument to be proved.

The weighted subject reduction lemma makes use of several properties, including a quantitative linear substitution lemma (Lemma 5.2), stating that from a type derivation of an object $\mathtt{O}[\![xl]\!]$ and type derivations for u, we can get a type derivation for the object $\mathtt{O}[\![ul]\!]$, obtained by linear substitution, i.e. by linearly substituting the occurrence x in the hole of the context $\mathtt{O}$ by u. The lemma also gives the size of the resulting derivation as a function on the sizes of the original ones.

We start by showing that given type derivations for two lists l and m, it is possible to construct a type derivation for the concatenation $l@m$.

Lemma 5.1. Let $X = \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$. If $\Phi_l \triangleright \Gamma \mid \delta \vdash_X l : \sigma$ and $\Phi_m \triangleright \Delta \mid \sigma \vdash_X m : \tau$, then there exists $\Phi_{l@m} \triangleright \Gamma + \Delta \mid \delta \vdash_X l@m : \tau$ such that $\mathtt{sz2}(\Phi_{l@m}) = \mathtt{sz2}(\Phi_l) + \mathtt{sz2}(\Phi_m) - 1$.

Proof. By induction on the type derivation $\Phi_l \triangleright \Gamma \mid \delta \vdash_X l : \sigma$. □

Intuitively, when combining Φ_l with Φ_m, one (ax) application becomes unnecessary. For instance, let $l = z_\epsilon;\epsilon$, and Φ_{z_ϵ} as in Example 5.1 and $\Phi_m \triangleright \Delta \mid \sigma \vdash_{\mathcal{H}_{\mathtt{E}}} m : \tau$, we have both

$$\Phi_l := \dfrac{\Phi_{z_\epsilon} \qquad \dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}\,(\mathtt{ax})}{z : \{\!\{\varphi\}\!\} \mid \{\!\{\varphi\}\!\} \supset \sigma \vdash z_\epsilon;\epsilon : \sigma} \qquad \Phi_{l@m} := \dfrac{\Phi_{z_\epsilon} \qquad \Phi_m}{\Delta + z : \{\!\{\varphi\}\!\} \mid \{\!\{\varphi\}\!\} \supset \sigma \vdash z_\epsilon;m : \tau}$$

Lemma 5.2 (Linear substitution). Let $X \in \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$. If $\Phi_{\mathtt{O}[\![xl]\!]} \triangleright x : \{\!\{\rho_i\}\!\}_{i\in I}; \Gamma \mid \Sigma \vdash_X \mathtt{O}[\![xl]\!] : \tau$ and $(\Phi^i_u \triangleright \Delta_i \mid \_ \vdash_X u : \rho_i)_{i\in I}$, then $\Phi_{\mathtt{O}[\![ul]\!]} \triangleright x : \{\!\{\rho_i\}\!\}_{i\in I\setminus K}; \Gamma + _{i\in K}\, \Delta_i \mid \Sigma \vdash_X \mathtt{O}[\![ul]\!] : \tau$, for some $K \subseteq I$ where $\mathtt{sz2}(\Phi_{\mathtt{O}[\![ul]\!]}) = \mathtt{sz2}(\Phi_{\mathtt{O}[\![xl]\!]}) +_{i\in K} \mathtt{sz2}(\Phi^i_u) - |K|$. Moreover, if $\mathtt{O}|_p = \Box$ and $p \in \mathtt{pos}(\mathtt{O}[\![xl]\!])$ is a T-occurrence of $\mathtt{O}[\![xl]\!]$ in $\Phi_{\mathtt{O}[\![xl]\!]}$, then $K \neq \varnothing$.

Proof. By induction on the typing derivation $\Phi_{\mathtt{O}[\![xl]\!]} \triangleright x : \{\!\{\rho_i\}\!\}_{i\in I}; \Gamma \mid \Sigma \vdash_X \mathtt{O}[\![xl]\!] : \tau$. We only show here the cases $\mathtt{O} = \Box$ and $\mathtt{O} = \mathtt{C}; m$, as the other ones are similar to the second case.

— If $\mathtt{O} = \Box$, then $\Sigma = \_$ and Φ_{xl} is of the form:

$$\frac{\Phi_l \triangleright \Gamma' \mid \sigma \vdash_X l : \tau}{\Gamma' + \{x : \{\!\{\sigma\}\!\}\} \mid \_ \vdash_X xl : \tau}\ (\mathtt{ax})$$

Moreover, $\mathtt{sz2}(\Phi_{xl}) = \mathtt{sz2}(\Phi_l) + 2$. Therefore, $\sigma = \rho_k$ for some $k \in I$ and $\Gamma' = x : \{\!\{\rho_i\}\!\}_{i \in I \setminus \{k\}}; \Gamma$. Hence,

$$\Phi_{ul} := \frac{\Phi_u^k \triangleright \Delta_k \mid \_ \vdash_X u : \rho_k \qquad \Phi_l \triangleright x : \{\!\{\rho_i\}\!\}_{i \in I \setminus \{k\}}; \Gamma \mid \rho_k \vdash_X l : \tau}{x : \{\!\{\rho_i\}\!\}_{i \in I \setminus \{k\}}; \Gamma + \Delta_k \mid \_ \vdash_X ul : \tau}$$

Moreover, $\mathtt{sz2}(\Phi_{ul}) = \mathtt{sz2}(\Phi_l) + \mathtt{sz2}(\Phi_u^k) + 1 = \mathtt{sz2}(\Phi_{xl}) + \mathtt{sz2}(\Phi_u^k) - 1$. The result then holds for $K := \{k\}$.

— If $\mathtt{O} = \mathtt{C}; m$, then $\mathtt{O}[\![xl]\!] = \mathtt{C}[\![xl]\!]; m$. Moreover, the stoup Σ is equal to $\{\!\{\sigma_l\}\!\}_{l \in L} \supset \varphi$ and $\Phi_{\mathtt{C}[\![xl]\!];m}$ necessarily ends as follows:

$$\frac{(\Phi^j_{\mathtt{C}[\![xl]\!]} \triangleright \Gamma_j \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \sigma_j)_{j \in J} \qquad \Phi_m \triangleright \Pi \mid \varphi \vdash_X m : \tau}{\Pi +_{j \in J} \Gamma_j \mid \{\!\{\sigma_l\}\!\}_{l \in L} \supset \varphi \vdash_X \mathtt{C}[\![xl]\!]; m : \tau},$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathrm{E}}$ and $L = \varnothing$), and $J = L$ otherwise. In addition, the resulting type assignment $\Pi +_{j \in J} \Gamma_j$ has the form of the statement of the lemma, i.e. $\Pi +_{j \in J} \Gamma_j = x : \{\!\{\rho_i\}\!\}_{i \in I}; \Gamma$. Thus, Π can be written as $x : \{\!\{\rho_i\}\!\}_{i \in I_m}; \Pi'$ and each Γ_j ($j \in J$) can be written as $x : \{\!\{\rho_i\}\!\}_{i \in I_j}; \Gamma'_j$, for some I_m and some I_j ($j \in J$) such that $I = I_m \cup_{j \in J} I_j$, and some Π' and some Γ'_j ($j \in J$) such that $\Gamma = \Pi' +_{j \in J} \Gamma'_j$. Moreover, $\mathtt{sz2}(\Phi_{\mathtt{C}[\![xl]\!];m}) = \mathtt{sz2}(\Phi_m) +_{j \in J} \mathtt{sz2}(\Phi^j_{\mathtt{C}[\![xl]\!]}) + 1$ by definition of $\mathtt{sz2}()$. By *i.h.* for each $j \in J$, we have $\Phi^j_{\mathtt{C}[\![ul]\!]} \triangleright x : \{\!\{\rho_i\}\!\}_{i \in I_j \setminus K_j}; \Gamma'_j +_{i \in K_j} \Delta_i \mid \_ \vdash \mathtt{C}[\![ul]\!] : \sigma_j$ for some $K_j \subseteq I_j$ where $\mathtt{sz2}(\Phi^j_{\mathtt{C}[\![ul]\!]}) = \mathtt{sz2}(\Phi^j_{\mathtt{C}[\![xl]\!]}) +_{i \in K_j} \mathtt{sz2}(\Phi^i_u) - |K_j|$. Let $K := \cup_{j \in J} K_j$. Therefore, $\Phi_{\mathtt{C}[\![ul]\!];m}$ is of the form:

$$\frac{(\Phi^j_{\mathtt{C}[\![ul]\!]} \triangleright x : \{\!\{\rho_i\}\!\}_{i \in I_j \setminus K_j}; \Gamma'_j +_{i \in K_j} \Delta_i \mid \_ \vdash \mathtt{C}[\![ul]\!] : \sigma_j)_{j \in J} \qquad \Phi_m \triangleright x : \{\!\{\rho_i\}\!\}_{i \in I_m}; \Pi' \mid \varphi \vdash m : \tau}{x : \{\!\{\rho_i\}\!\}_{i \in I \setminus K}; \Pi' +_{j \in J} \Gamma'_j +_{i \in K} \Delta_i \mid \{\!\{\sigma_l\}\!\}_{l \in L} \supset \varphi \vdash \mathtt{C}[\![ul]\!]; m : \tau},$$

where $\mathtt{sz2}(\Phi_{\mathtt{C}[\![ul]\!];m}) = \mathtt{sz2}(\Phi_m) + _{j \in J} \mathtt{sz2}(\Phi^j_{\mathtt{C}[\![ul]\!]}) + 1 =_{i.h.} \mathtt{sz2}(\Phi_m) + _{j \in J} (\mathtt{sz2}(\Phi^j_{\mathtt{C}[\![xl]\!]}) + _{i \in K_j} \mathtt{sz2}(\Phi^i_u) - |K_j|) + 1 = \mathtt{sz2}(\Phi_m) + _{j \in J} \mathtt{sz2}(\Phi^j_{\mathtt{C}[\![xl]\!]}) + _{i \in K} \mathtt{sz2}(\Phi^i_u) - |K| + 1 = \mathtt{sz2}(\Phi_{\mathtt{C}[\![xl]\!];m}) + _{i \in K} \mathtt{sz2}(\Phi^i_u) - |K|$. This concludes the first part of the proof of the statement.

Now, suppose $p \in \mathtt{pos}(\mathtt{C}[\![xl]\!]; m)$ is a T-occurrence of $\mathtt{C}[\![xl]\!]; m$ such that $(\mathtt{C}; m)|_p = \Box$. Then, $p = 0p'$ where $p' \in \mathtt{pos}(\mathtt{C}[\![xl]\!])$ is a T-occurrence in $\Phi^j_{\mathtt{C}[\![xl]\!]}$ for some $j \in J$. In this case, $K_j \neq \varnothing$ holds by the *i.h.* so that $K \neq \varnothing$. □

Example 5.2. Let t be the term $x(x_\epsilon; \epsilon)$, and consider the following typing derivation Φ_t (in either system $\mathcal{S}_{\mathrm{E}}$ or $\mathcal{H}_{\mathrm{E}}$) such that $\mathtt{sz2}(\Phi_t) = 7$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}}{x : \{\!\{\sigma\}\!\} \mid \_ \vdash x_\epsilon : \sigma}\ (\mathtt{hlist}) \qquad \dfrac{}{\varnothing \mid \tau \vdash \epsilon : \tau}}{x : \{\!\{\sigma\}\!\} \mid \{\!\{\sigma\}\!\} \supset \tau \vdash x_\epsilon; \epsilon : \tau}}{x : \{\!\{\sigma, \{\!\{\sigma\}\!\} \supset \tau\}\!\} \mid \_ \vdash x(x_\epsilon; \epsilon) : \tau}\ (\mathtt{hlist})$$

Let u be a term such that $\Phi_u \triangleright \Delta \mid \_ \vdash u : \sigma$. Then, for $t' = x(u\,\epsilon; \epsilon)$, $\Phi_{t'}$ is of the form:

$$\dfrac{\dfrac{\dfrac{\Phi_u \triangleright \Delta \mid \_ \vdash u : \sigma \qquad \dfrac{}{\varnothing \mid \sigma \vdash \epsilon : \sigma}}{\Delta \mid \_ \vdash u\epsilon : \sigma} \qquad \dfrac{}{\varnothing \mid \tau \vdash \epsilon : \tau}}{\Delta \mid \{\!\{\sigma\}\!\} \supset \tau \vdash u\epsilon;\epsilon : \tau}}{x : \{\!\{\{\!\{\sigma\}\!\} \supset \tau\}\!\} + \Delta \mid \_ \vdash t' : \tau}\ (\texttt{hlist})$$

where $\texttt{sz2}(\Phi_{t'}) = \texttt{sz2}(\Phi_u) + 6 = \texttt{sz2}(\Phi_t) + \texttt{sz2}(\Phi_u) - 1$.

We can now state one of our main *forward* properties for systems $\mathcal{H}_{\mathrm{E}}$ and $\mathcal{S}_{\mathrm{E}}$, which describes all the reduction steps that not only preserve types and environments but also decrease the measure $\texttt{sz2}(\_)$. The forthcoming Examples 5.3 and 5.4 illustrate a case when that is not the case.

Lemma 5.3 (Weighted subject reduction for the E-calculus). Let $X \in \{\mathcal{H}_{\mathrm{E}}, \mathcal{S}_{\mathrm{E}}\}$. Let $\Phi \triangleright \Gamma \mid \Sigma \vdash_X o : \tau$ and $o \to_{\mathrm{E}} o'$ reduces a $(\mathtt{B@}, \mathtt{c}, \mathtt{d}, \mathtt{w})$-redex T-occurrence of o in Φ.

1. If $X = \mathcal{H}_{\mathrm{E}}$, then $\Phi' \triangleright \Gamma \vdash_{\mathcal{H}_{\mathrm{E}}} o' : \tau$ and $\texttt{sz2}(\Phi) > \texttt{sz2}(\Phi')$.
2. If $X = \mathcal{S}_{\mathrm{E}}$ and the E-step is not $\mathtt{w}$, then $\Phi' \triangleright \Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathrm{E}}} o' : \tau$ and $\texttt{sz2}(\Phi) > \texttt{sz2}(\Phi')$.

Proof. By induction on the reduction relation $\to_{\mathrm{E}}$, using Lemmas 5.1 and 5.2. We only show the most interesting cases; remark that $\to_{\mathtt{w}}$ is out of scope when $X = \mathcal{S}_{\mathrm{E}}$.

— If $o = \mathtt{L}[\![\lambda x.v]\!](u;l) \to_{\mathtt{dB}_{\mathtt{cons}}} \mathtt{L}[\![v[x\backslash u]l]\!] = o'$, then we show $\texttt{sz2}(\Phi) > \texttt{sz2}(\Phi')$ by induction on $\mathtt{L}$. Let $\mathtt{L} = \Box$. By construction, we have that $\Sigma = \_$ and Φ is of the form:

$$\dfrac{\dfrac{\Phi_v \triangleright x : \{\!\{\rho_l\}\!\}_{l\in L}; \Pi \mid \_ \vdash_X v : \sigma}{\Pi \mid \_ \vdash_X \lambda x.v : \{\!\{\rho_l\}\!\}_{l\in L} \supset \sigma} \qquad \Phi_{u;l} := \dfrac{(\Phi_u^j \triangleright \Gamma_j \mid \_ \vdash_X u : \rho_j)_{j\in J} \qquad \Phi_l \triangleright \Delta \mid \sigma \vdash_X l : \tau}{\Delta +_{j\in J} \Gamma_j \mid \{\!\{\rho_l\}\!\}_{l\in L} \supset \sigma \vdash_X u;l : \tau}}{\Pi + \Delta +_{j\in J} \Gamma_j \mid \_ \vdash_X (\lambda x.v)(u;l) : \tau}$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathrm{E}}$ and $L = \varnothing$), and $J = L$ otherwise. Moreover, $\texttt{sz2}(\Phi) = \texttt{sz2}(\Phi_v) + _{j\in J}\, \texttt{sz2}(\Phi_u^j) + \texttt{sz2}(\Phi_l) + 3$. Hence, we have Φ' of the form:

$$\Phi_{v[x\backslash u]} := \dfrac{\dfrac{\Phi_v \triangleright x : \{\!\{\rho_l\}\!\}_{l\in L}; \Pi \mid \_ \vdash_X v : \sigma \quad (\Phi_u^j \triangleright \Gamma_j \mid \_ \vdash_X u : \rho_j)_{j\in J}}{\Pi +_{j\in J} \Gamma_j \mid \_ \vdash_X v[x\backslash u] : \sigma} \qquad \Phi_l \triangleright \Delta \mid \sigma \vdash_X l : \tau}{\Pi + \Delta +_{j\in J} \Gamma_j \mid \_ \vdash_X v[x\backslash u]l : \tau}$$

Moreover, $\texttt{sz2}(\Phi') = \texttt{sz2}(\Phi_{v[x\backslash u]}) + \texttt{sz2}(\Phi_l) + 1 = \texttt{sz2}(\Phi_v) + _{j\in J}\, \texttt{sz2}(\Phi_u^j) + \texttt{sz2}(\Phi_l) + 2 < \texttt{sz2}(\Phi)$.

Let $\mathtt{L} = \mathtt{L}'[y\backslash u']$, so that $\mathtt{L}[\![t]\!] = \mathtt{L}'[\![t]\!][y\backslash u']$ for any t. By construction, Φ is of the form:

$$\Phi_{\mathtt{L}[\![t]\!]} \qquad \Phi_{u;l} := \dfrac{(\Phi_u^j \triangleright \Gamma_j \mid \_ \vdash_X u : \sigma_j)_{j\in J} \qquad \Phi_l \triangleright \Delta \mid \sigma \vdash_X l : \tau}{\Delta +_{j\in J} \Gamma_j \mid \{\!\{\sigma_l\}\!\}_{l\in L} \supset \sigma \vdash_X u;l : \tau}$$

$$\Gamma_0 +_{j\in J'} \Pi_j + \Delta +_{j\in J} \Gamma_j \vdash_X \mathtt{L}'[\![\lambda x.v]\!][y\backslash u'](u;l) : \tau$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathrm{E}}$ and $L = \varnothing$) and $J = L$ otherwise. Moreover,

$$\Phi_{\mathtt{L}[\![t]\!]} := \dfrac{\Phi_{\mathtt{L}'[\![t]\!]} \triangleright \Gamma_0; y : \{\!\{\rho_l\}\!\}_{l\in L'} \vdash_X \mathtt{L}'[\![\lambda x.v]\!] : \{\!\{\sigma_l\}\!\}_{l\in L} \supset \sigma \qquad (\Phi_{u'}^j \triangleright \Pi_j \mid \_ \vdash_X u' : \rho_j)_{j\in J'}}{\Gamma_0 +_{j\in J'} \Pi_j \vdash_X \mathtt{L}'[\![\lambda x.v]\!][y\backslash u'] : \{\!\{\sigma_l\}\!\}_{l\in L} \supset \sigma}$$

where $|J'| = 1$ if ($X = \mathcal{S}_{\mathrm{E}}$ and $L' = \varnothing$) and $J' = L'$ otherwise. We can then construct the following derivation:

$$\Phi_{\mathtt{L}'[\![\lambda x.v]\!](u;l)} := \dfrac{\Phi_{\mathtt{L}'[\![l]\!]} \rhd \Gamma_0; y : \{\!\{\rho_l\}\!\}_{l\in L'} \vdash_X \mathtt{L}'[\![\lambda x.v]\!] : \{\!\{\sigma_l\}\!\}_{l\in L} \supset \sigma \qquad \Phi_{u;l}}{\Gamma_0; y : \{\!\{\rho_l\}\!\}_{l\in L'} + \Delta + _{j\in J}\, \Gamma_j \vdash_X \mathtt{L}'[\![\lambda x.v]\!](u;l) : \tau}$$

By the *i.h.* there is a derivation $\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!]} \rhd \Gamma_0; y : \{\!\{\rho_l\}\!\}_{l\in L'} + \Delta + _{j\in J}\, \Gamma_j \vdash_X \mathtt{L}'[\![v[x\backslash u]l]\!] : \tau$ such that $\mathtt{sz2}(\Phi_{\mathtt{L}'[\![\lambda x.v]\!](u;l)}) > \mathtt{sz2}(\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!]})$.
We thus conclude with a derivation $\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!][y\backslash u']}$ of the form:

$$\dfrac{\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!]} \rhd \Gamma_0; y : \{\!\{\rho_l\}\!\}_{l\in L'} + \Delta + _{j\in J}\, \Gamma_j \vdash_X \mathtt{L}'[\![v[x\backslash u]l]\!] : \tau \qquad (\Phi^j_{u'} \rhd \Pi_j \mid \_ \vdash_X u' : \rho_j)_{j\in J'}}{\Gamma_0 + _{j\in J'}\, \Pi_j + \Delta + _{j\in J}\, \Gamma_j \vdash_X \mathtt{L}'[\![v[x\backslash u]l]\!][y\backslash u'] : \tau}$$

Besides, $\mathtt{sz2}(\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!][y\backslash u']}) = \mathtt{sz2}(\Phi_{\mathtt{L}'[\![v[x\backslash u]l]\!]}) + _{j\in J'}\, \mathtt{sz2}(\Phi^j_{u'}) + 1 <_{i.h.} \mathtt{sz2}(\Phi_{\mathtt{L}'[\![\lambda x.v]\!](u;l)}) + _{j\in J'}\, \mathtt{sz2}(\Phi^j_{u'}) + 1 = \mathtt{sz2}(\Phi_{\mathtt{L}'[\![\lambda x.v]\!]}) + _{j\in J'}\, \mathtt{sz2}(\Phi^j_{u'}) + \mathtt{sz2}(\Phi_{u;l}) + 2 = \mathtt{sz2}(\Phi_{\mathtt{L}'[\![\lambda x.v]\!][y\backslash u'](u;l)})$.

— If $o = \mathtt{C}[\![xl]\!][x\backslash u] \to_{\mathtt{c}} \mathtt{C}[\![ul]\!][x\backslash u] = o'$, then, by construction, $\Sigma = \_$ and

$$\Phi := \dfrac{(\Phi^i_u \rhd \Delta_i \mid \_ \vdash_X u : \rho_i)_{i\in I} \qquad \Phi_{\mathtt{C}[\![xl]\!]} \rhd x : \{\!\{\rho_i\}\!\}_{i\in I}; \Pi \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \tau}{\Pi + _{i\in I}\, \Delta_i \mid \_ \vdash_X \mathtt{C}[\![xl]\!][x\backslash u] : \tau}$$

where $\mathtt{sz2}(\Phi) = \mathtt{sz2}(\Phi_{\mathtt{C}[\![xl]\!]}) + _{i\in I}\, \mathtt{sz2}(\Phi^i_u) + 1$ (since $|\mathtt{C}[\![xl]\!]|_x > 1$ then $I \neq \varnothing$). By Lemma 5.2, we have derivations $\Phi_{\mathtt{C}[\![ul]\!]} \rhd x : \{\!\{\rho_i\}\!\}_{i\in I\setminus K}; \Pi + _{i\in K}\, \Delta_i \mid \_ \vdash \mathtt{C}[\![ul]\!] : \tau$ for some $K \subseteq I$ such that $\mathtt{sz2}(\Phi_{\mathtt{C}[\![ul]\!]}) = \mathtt{sz2}(\Phi_{\mathtt{C}[\![xl]\!]}) + _{i\in K}\, \mathtt{sz2}(\Phi^i_u) - |K|$. Hence,

$$\Phi' := \dfrac{\Phi_{\mathtt{C}[\![ul]\!]} \rhd x : \{\!\{\rho_i\}\!\}_{i\in I\setminus K}; \Pi + _{i\in K}\, \Delta_i \mid \_ \vdash_X \mathtt{C}[\![ul]\!] : \tau \qquad (\Phi^i_u \rhd \Delta_i \mid \_ \vdash_X u : \rho_i)_{i\in I\setminus K}}{\Pi + _{i\in I}\, \Delta_i \mid \_ \vdash_X \mathtt{C}[\![ul]\!][x\backslash u] : \tau}$$

where $\mathtt{sz2}(\Phi') = \mathtt{sz2}(\Phi_{\mathtt{C}[\![ul]\!]}) + _{i\in I\setminus K}\, \mathtt{sz2}(\Phi^i_u) + 1 =_{L.5.2} \mathtt{sz2}(\Phi_{\mathtt{C}[\![xl]\!]}) + _{i\in I}\, \mathtt{sz2}(\Phi^i_u) - |K| + 1 \leqslant \mathtt{sz2}(\Phi)$. By hypothesis, the hole of $\mathtt{C}$ is a T-occurrence in Φ, so that Lemma 5.2 guarantees $K \neq \varnothing$ and thus $\mathtt{sz2}(\Phi') < \mathtt{sz2}(\Phi)$. □

Although typability is stable by all the rules in system $\mathcal{H}_{\mathtt{E}}$, the weighted subject reduction property as it stands does not hold when reduction occurs in untyped occurrences of redexes. This is illustrated in the following example.

Example 5.3. Let $t = (\lambda y.x_\epsilon)(u;\epsilon)$ and consider the following typing derivation for t:

$$\Phi_t := \dfrac{\dfrac{\dfrac{\overline{\varnothing \mid \sigma \vdash \epsilon : \sigma}}{x : \{\!\{\sigma\}\!\} \mid \_ \vdash x_\epsilon : \sigma}}{x : \{\!\{\sigma\}\!\} \mid \_ \vdash \lambda y.x_\epsilon : \{\!\{\ \}\!\} \supset \sigma} \qquad \dfrac{\overline{\varnothing \mid \sigma \vdash \epsilon : \sigma}}{\varnothing \mid \{\!\{\ \}\!\} \supset \sigma \vdash u; \epsilon : \sigma}\,(\supset \mathtt{l})}{x : \{\!\{\sigma\}\!\} \mid \_ \vdash t : \sigma}$$

where $\mathtt{sz2}(\Phi_t) = 7$. Since there is no typing information for u, the size of Φ_t does not depend by any means on u. Observe that any redex occurrence in u would not be typed in Φ_t and, for any reduction sequence $u \to^+_{\mathtt{E}} u'$, the typing derivation for $t' = (\lambda y.x_\epsilon)(u';\epsilon)$ is obtained from Φ_t by replacing u by u' in rule $(\supset \mathtt{l})$ above. In other words, $\Phi_{t'} \rhd x : \{\!\{\sigma\}\!\} \mid \_ \vdash t' : \sigma$ where $\mathtt{sz2}(\Phi_{t'}) = \mathtt{sz2}(\Phi_t)$.

Notice also that typability is stable by the rule $\to_{\mathtt{w}}$, but the weighted subject reduction property as it stands neither does hold in system $\mathcal{S}_{\mathtt{E}}$ due to the relevance property and the use of witnesses. This is illustrated in the following example.

Example 5.4. Let $t = \lambda y.(\lambda z.x_\epsilon)(y;\epsilon) \to_{\mathtt{dB}_{\mathtt{cons}}} \lambda y.x_\epsilon[z\backslash y]\epsilon \to_{\mathtt{w}} \lambda y.x_\epsilon\epsilon \to_{@_{\mathtt{var}}} \lambda y.x_\epsilon = t'$. We have that t is typable, e.g. we can derive $x : \{\!\{\sigma\}\!\} \mid \_ \vdash t : \{\!\{\tau\}\!\} \supset \sigma$. Despite the fact that t' is also typable, the judgement $x : \{\!\{\sigma\}\!\} \mid \_ \vdash t' : \{\!\{\tau\}\!\} \supset \sigma$ is not derivable. Indeed, $\triangleright x : \{\!\{\sigma\}\!\} \mid \_ \vdash t' : \{\!\{\ \}\!\} \supset \sigma$, so the type of t is not preserved.

This phenomenon is due to the loss of information in the typing environment of the derivation corresponding to the $\mathtt{w}$-redex. For instance, the judgement $x : \{\!\{\sigma\}\!\}, y : \{\!\{\tau\}\!\} \vdash x_\epsilon[z\backslash y] : \sigma$ is derivable and $u = x_\epsilon[z\backslash y] \to_{\mathtt{w}} x_\epsilon$, but $x : \{\!\{\sigma\}\!\}, y : \{\!\{\tau\}\!\} \vdash x_\epsilon : \sigma$ is not derivable. Indeed, $\triangleright x : \{\!\{\sigma\}\!\} \vdash x_\epsilon : \sigma$, so the type environment of x_ϵ is shrinked. Then, if v (resp. v') is the first (resp. second) term in the reduction sequence above, we observe that when we replace the former (sub)derivation by the shrinked one in the derivation of $x : \{\!\{\sigma\}\!\} \mid \_ \vdash v : \{\!\{\tau\}\!\} \supset \sigma$, we obtain a derivation of $x : \{\!\{\sigma\}\!\} \mid \_ \vdash v' : \{\!\{\ \}\!\} \supset \sigma$. Thus, the non-preservation of type environments affects the preservation of types.

On the other hand, witnesses are necessary to guarantee that arguments are normalizable, so that their type environments must be included in the whole typing derivation. The following example illustrates the need of such environments.

Example 5.5. Let $I = \lambda x.x_\epsilon$ (identity), $A = \lambda x.x(x_\epsilon;\epsilon)$ (self-application), $\Omega = A(A;\epsilon)$ (self-reproductor), and consider an alternative typing rule where the environment of a term in the head of a list is not memorized in the final environment of the object:

$$\frac{\Gamma \mid \_ \vdash t : \rho \qquad \Delta \mid \tau \vdash l : \sigma}{\Delta \mid \{\!\{\ \}\!\} \supset \tau \vdash t;l : \sigma}\ (\supset \mathtt{l}_{\notin}{}')$$

We have $\Phi_I \triangleright \varnothing \mid \_ \vdash I : \{\!\{\tau\}\!\} \supset \tau$, $\Phi_A \triangleright \varnothing \mid \_ \vdash A : \{\!\{\sigma, \{\!\{\sigma\}\!\} \supset \varphi\}\!\} \supset \varphi$ and the following derivation Φ_m for $m = z(z_\epsilon;\epsilon);\epsilon$:

$$\frac{\Phi_{z(z_\epsilon;\epsilon)} \triangleright z : \{\!\{\rho', \{\!\{\rho'\}\!\} \supset \rho\}\!\} \vdash z(z_\epsilon;\epsilon) : \rho \qquad \dfrac{}{\varnothing \mid \{\!\{\tau\}\!\} \supset \tau \vdash \epsilon : \{\!\{\tau\}\!\} \supset \tau}}{\varnothing \mid \{\!\{\ \}\!\} \supset \{\!\{\tau\}\!\} \supset \tau \vdash m : \{\!\{\tau\}\!\} \supset \tau}\ (\supset \mathtt{l}_{\notin}{}')$$

The term $t = (\lambda z.(\lambda y.I)m)(A;\epsilon)$ is then typable:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Phi_I}{\varnothing \mid \_ \vdash \lambda y.I : \{\!\{\ \}\!\} \supset \{\!\{\tau\}\!\} \supset \tau} \qquad \Phi_m}{\varnothing \mid \_ \vdash (\lambda y.I)m : \{\!\{\tau\}\!\} \supset \tau}}{\varnothing \mid \_ \vdash \lambda z.(\lambda y.I)m : \{\!\{\ \}\!\} \supset \{\!\{\tau\}\!\} \supset \tau} \qquad \dfrac{\Phi_A \qquad \dfrac{}{\varnothing \mid \{\!\{\tau\}\!\} \supset \tau \vdash \epsilon : \{\!\{\tau\}\!\} \supset \tau}}{\varnothing \mid \{\!\{\ \}\!\} \supset \{\!\{\tau\}\!\} \supset \tau \vdash A;\epsilon : \{\!\{\tau\}\!\} \supset \tau}}{\varnothing \mid \_ \vdash t : \{\!\{\tau\}\!\} \supset \tau}$$

but $t \to_{\mathtt{E}}^{+} (\lambda y.I)(\Omega;\epsilon)$, which is a non-terminating term.

It is also worth noticing that the $\mathtt{sz2}$ function plays a central role in obtaining a strictly decreasing measure in the lemma above. More precisely, if we consider the standard measure on typing derivations, written $\mathtt{sz}$, which counts 1 for every node of the derivation tree, then the weighted subject reduction property does not hold. Here is an example.

Example 5.6. Let $t, u, t', \Phi_t, \Phi_u^1$ and $\Phi_{t'}$ as in the Example 5.2 and let $\Phi_u^2 \triangleright \Delta_2 \mid \_ \vdash u : \{\!\{\sigma\}\!\} \supset \tau$. Remark that $\mathtt{sz}(\Phi_t) = 5$ and that $\mathtt{sz}(\Phi_{t'}) = 5 + \mathtt{sz}(\Phi_u^1)$. Therefore,

$$\Phi := \frac{\Phi_u^1 \rhd \Delta_1 \mid \_ \vdash u : \sigma \qquad \Phi_u^2 \rhd \Delta_2 \mid \_ \vdash u : \{\!\{\sigma\}\!\} \supset \tau \qquad \Phi_t \rhd x : \{\!\{\sigma, \{\!\{\sigma\}\!\} \supset \tau\}\!\} \mid \_ \vdash t : \tau}{\Delta_1 + \Delta_2 \mid \_ \vdash t[x\backslash u] : \tau}$$

where $\mathtt{sz}(\Phi) = 6 + _{i=1,2}\ \mathtt{sz}(\Phi_u^i)$.

Given the reduction step $t[x\backslash u] \to_{\mathtt{c}} x(u\,\epsilon;\epsilon)[x\backslash u] = t'[x\backslash u]$, there is a derivation Φ' typing $t'[x\backslash u]$ such that $\mathtt{sz}(\Phi') = \mathtt{sz}(\Phi)$. Indeed,

$$\Phi' := \frac{\Phi_u^2 \rhd \Delta_2 \mid \_ \vdash u : \{\!\{\sigma\}\!\} \supset \tau \qquad \Phi_{t'} \rhd \Delta_1 \mid \{\!\{\sigma\}\!\} \supset \tau \vdash t' : \tau}{\Delta_1 + \Delta_2 \mid \_ \vdash t'[x\backslash u] : \tau}$$

As we mentioned in the introduction, the $\bar{\lambda}$-calculus (Herbelin 1995) is not compatible with a resource aware semantics, as illustrated by the following example. Consider a $\bar{\lambda}$-reduction of the form $o = (tl)[x\backslash u] \to t[x\backslash u]l[x\backslash u] = o'$, and suppose $|tl|_x = 0$. Let Φ be a typing derivation for the object o, thus having the following form:

$$\Phi := \frac{\dfrac{\Phi_t \rhd \Gamma_t \mid \_ \vdash t : \sigma_t \qquad \Phi_l \rhd \Gamma_l \mid \sigma_t \vdash l : \tau}{\Gamma_t + \Gamma_l \mid \_ \vdash tl : \tau} \qquad \Phi_u \rhd \Delta \mid \_ \vdash u : \sigma}{(\Gamma_t + \Gamma_l) + \Delta \mid \_ \vdash (tl)[x\backslash u] : \tau}$$

The typing derivation for the object o', let say Φ', must use *twice* the typing tree Φ_u, and thus $\mathtt{sz2}(\Phi) > \mathtt{sz2}(\Phi')$ cannot hold. In other words, propagation of substitution w.r.t. the structure of terms induces useless duplications of empty resources, turning out to be inappropriate in the framework of resource aware semantics.

We end this section with two main corollaries of Lemma 5.3 which are key to establish termination of reduction for typable terms in the forthcoming sections.

Corollary 5.1. If o is $\mathcal{H}_{\mathtt{E}}$-typable, then any $\mathtt{E}$-reduction sequence contracting only $\mathtt{E}$-redex T-occurrences is finite.

As noticed before, given a type derivation Φ for an object o in system $\mathcal{S}_{\mathtt{E}}$, any redex of o is a redex T-occurrence of o in Φ. As a consequence, if $\Phi \rhd \Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathtt{E}}} o : \tau$ and $o \to_{\mathtt{E}\backslash\mathtt{w}} o'$, then $\Phi' \rhd \Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathtt{E}}} o' : \tau$ and $\mathtt{sz2}(\Phi) > \mathtt{sz2}(\Phi')$. Thus,

Corollary 5.2. If o is $\mathcal{S}_{\mathtt{E}}$-typable, then $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

5.2. *The 'Backward' properties*

This section shows preservation of any typing by anti-reduction, i.e. by transforming/rewriting a term *backward*. This property is crucial to obtain 'normalization implies typability.' We start by proving a couple of technical results.

The following property gives the converse implication of Lemma 5.1.

Lemma 5.4. Let $X = \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$. If $\Phi_{l@m} \rhd \Gamma \mid \delta \vdash_X l@m : \tau$, then there exists $\Gamma_1, \Gamma_2, \sigma$ such that $\Gamma = \Gamma_1 + \Gamma_2$, $\Phi_l \rhd \Gamma_1 \mid \delta \vdash_X l : \sigma$ and $\Phi_m \rhd \Gamma_2 \mid \sigma \vdash_X m : \tau$.

Proof. The proof is by induction on $\Phi_{l@m} \rhd \Gamma \mid \delta \vdash_X l@m : \tau$. □

While the following property gives the converse implication of Lemma 5.2.

Lemma 5.5 (Reverse linear substitution). Let $X \in \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$ and $\mathtt{O}[\![xl]\!]$ be an $\mathtt{E}$-object and u be an $\mathtt{E}$-term such that $|u|_x = 0$ and $\Phi \rhd \Gamma \mid \Sigma \vdash_X \mathtt{O}[\![ul]\!] : \tau$. Then there exist $\Gamma_0, I, (\Gamma_i)_{i\in I}, (\sigma_i)_{i\in I}$ such that $\Gamma = \Gamma_0 +_{i\in I} \Gamma_i$, $(\Phi^i_u \rhd \Gamma_i \mid \_ \vdash_X u : \sigma_i)_{i\in I}$ and $\Phi' \rhd \Gamma_0 + \{x : \{\!\{\sigma_i\}\!\}_{i\in I}\} \mid \Sigma \vdash_X \mathtt{O}[\![xl]\!] : \tau$. In particular, if $X = \mathcal{S}_{\mathtt{E}}$, then $I \neq \varnothing$.

Proof. Let $X \in \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$. We reason by induction on the typing derivation $\Phi_{\mathtt{O}[\![ul]\!]} \rhd \Gamma \mid \Sigma \vdash_X \mathtt{O}[\![ul]\!] : \tau$. We only show here the most interesting case.

If $\mathtt{O} = \mathtt{C}; m$, then by construction $\Phi_{\mathtt{C}[\![ul]\!];m}$ is of the form:

$$\frac{(\Phi^j_{\mathtt{C}[\![ul]\!]} \rhd \Delta_j \vdash_X \mathtt{C}[\![ul]\!] : \rho_j)_{j\in J} \qquad \Phi_m \rhd \Gamma_m \mid \varphi \vdash_X m : \tau}{\Gamma_m +_{j\in J} \Delta_j \mid \{\!\{\rho_l\}\!\}_{l\in L} \supset \varphi \vdash_X \mathtt{C}[\![ul]\!]; m : \tau}$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathtt{E}}$ and $L = \varnothing$), and $J = L$ otherwise. By the *i.h.* for each $j \in J$, $\Delta_j = \Delta^j_0 +_{i\in I_j} \Gamma_i$ and $\Phi'_j \rhd \Delta^j_0 + \{x : \{\!\{\sigma_i\}\!\}_{i\in I_j}\} \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \rho_j$ and $(\Phi^i_u \rhd \Gamma_i \mid \_ \vdash_X u : \sigma_i)_{i\in I_j}$. Let $I := \cup_{j\in J} I_j$. Hence,

$$\Phi' := \frac{(\Phi'_j \rhd \Delta^j_0 + \{x : \{\!\{\sigma_i\}\!\}_{i\in I_j}\} \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \rho_j)_{j\in J} \qquad \Phi_m \rhd \Gamma_m \mid \varphi \vdash_X m : \tau}{\Gamma_m +_{j\in J} \Delta^0_j + \{x : \{\!\{\sigma_i\}\!\}_{i\in I}\} \mid \{\!\{\rho_l\}\!\}_{l\in L} \supset \varphi \vdash_X \mathtt{C}[\![xl]\!]; m : \tau}$$

We then conclude with $\Gamma_0 := \Gamma_m +_{j\in J} \Delta^0_j$ since $\Gamma_0 +_{i\in I} \Gamma_i = \Gamma_m +_{j\in J} \Delta^0_j +_{j\in J} (+_{i\in I_j} \Gamma_i) = \Gamma_m +_{j\in J} (\Delta^0_j +_{i\in I_j} \Gamma_i) = \Gamma_m +_{j\in J} \Delta_j = \Gamma$. □

Finally, we present the last key property, relating typing with expansion.

Lemma 5.6 (Subject expansion for the $\mathtt{E}$-calculus). Let $X \in \{\mathcal{H}_{\mathtt{E}}, \mathcal{S}_{\mathtt{E}}\}$, $o \to_{\mathtt{E}} o'$ and $\Phi' \rhd \Gamma \mid \Sigma \vdash_X o' : \tau$.

1. If $X = \mathcal{H}_{\mathtt{E}}$, then $\Phi \rhd \Gamma \mid \Sigma \vdash_{\mathcal{H}_{\mathtt{E}}} o : \tau$.
2. If $X = \mathcal{S}_{\mathtt{E}}$ and the reduction is not a $\mathtt{w}$-step, then $\Phi \rhd \Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathtt{E}}} o : \tau$.

Proof. The proof is by induction on $o \to_{\mathtt{E}} o'$, using Lemma 5.4 and 5.5. We only show here the following two most interesting cases:

— If $o = \mathtt{L}[\![\lambda x.v]\!](u; l) \to_{\mathtt{dB}_{\mathtt{cons}}} \mathtt{L}[\![v[x\backslash u]l]\!] = o'$, then we proceed by induction on $\mathtt{L}$. Let $\mathtt{L} = \Box$. By construction, we have $\Sigma = \_$ and $\Gamma = \Gamma_0 +_{j\in J} \Gamma_j + \Delta$ and Φ' is of the form:

$$\frac{\dfrac{\Phi_v \rhd \Gamma_0; x : \{\!\{\rho_i\}\!\}_{i\in I} \mid \_ \vdash_X v : \sigma \qquad (\Phi^j_u \rhd \Gamma_j \mid \_ \vdash_X u : \rho_j)_{j\in J}}{\Gamma_0 +_{j\in J} \Gamma_j \mid \_ \vdash_X v[x\backslash u] : \sigma} \qquad \Phi_l \rhd \Delta \mid \sigma \vdash_X l : \tau}{\Gamma_0 +_{j\in J} \Gamma_j + \Delta \mid \_ \vdash_X v[x\backslash u]l : \tau}$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathtt{E}}$ and $I = \varnothing$), and $J = I$ otherwise. Therefore, we can construct the derivation Φ below:

$$\frac{\dfrac{\Phi_v \rhd \Gamma_0; x : \{\!\{\rho_i\}\!\}_{i\in I} \mid \_ \vdash_X v : \sigma}{\Gamma_0 \mid \_ \vdash_X \lambda x.v : \{\!\{\rho_i\}\!\}_{i\in I} \supset \sigma} \qquad \dfrac{(\Phi^j_u \rhd \Gamma_j \mid \_ \vdash_X u : \rho_j)_{j\in J} \qquad \Phi_l \rhd \Delta \mid \sigma \vdash_X l : \tau}{\Delta +_{j\in J} \Gamma_j \mid \{\!\{\rho_i\}\!\}_{i\in I} \supset \sigma \vdash_X u; l : \tau}}{\Gamma_0 +_{j\in J} \Gamma_j + \Delta \mid \_ \vdash_X (\lambda x.v)(u; l) : \tau}$$

If $\mathtt{L} = \mathtt{L}'[y\backslash s]$, then $\mathtt{L}[\![v[x\backslash u]l]\!] = \mathtt{L}'[\![v[x\backslash u]l]\!][y\backslash s]$. By construction, we have a derivation of the following form:

$$\Phi' := \frac{\Phi_t \rhd \Gamma_0; y : \{\!\{\rho_i\}\!\}_{i\in I} \mid \_ \vdash_X \mathtt{L}'[\![v[x\backslash u]l]\!] : \tau \qquad (\Phi^j_s \rhd \Gamma_j \mid \_ \vdash_X s : \rho_j)_{j\in J}}{\Gamma_0 +_{j\in J} \Gamma_j \mid \_ \vdash_X \mathtt{L}'[\![v[x\backslash u]l]\!][y\backslash s] : \tau}$$

where $|J| = 1$ if ($X = \mathcal{S}_{\mathtt{E}}$ and $I = \varnothing$), and $J = I$ otherwise. By *i.h.* on Φ_t, one has $\Phi'_t \triangleright \Gamma_0; y : \{\!|\rho_i|\!\}_{i\in I} \mid \_ \vdash \mathtt{L}'[\![\lambda x.v]\!](u;l) : \tau$ and, by construction, $\Gamma_0; y : \{\!|\rho_i|\!\}_{i\in I} = \Delta_1 + \Delta_2$ such that $\Phi_{t_1} \triangleright \Delta_1 \mid \_ \vdash \mathtt{L}'[\![\lambda x.v]\!] : \sigma$ and $\Phi_{t_2} \triangleright \Delta_2 \mid \sigma \vdash (u;l) : \tau$. We can assume by α-conversion that $|u;l|_y = 0$ thus, by Lemma 4.1, we necessarily have that $\Delta_1 = \Delta'; y : \{\!|\rho_i|\!\}_{i\in I}$ and $\Gamma_0 = \Delta' + \Delta_2$. Therefore, we can construct Φ below:

$$\dfrac{\dfrac{\Phi_{t_1} \triangleright \Delta'; y : \{\!|\rho_i|\!\}_{i\in I} \mid \_ \vdash_X \mathtt{L}'[\![\lambda x.v]\!] : \sigma \qquad (\Phi_s^j \triangleright \Gamma_j \mid \_ \vdash_X s : \rho_j)_{j\in J}}{\Delta' +_{j\in J} \Gamma_j \mid \_ \vdash_X \mathtt{L}'[\![\lambda x.v]\!][y\backslash s] : \sigma} \qquad \Phi_{t_2} \triangleright \Delta_2 \mid \sigma \vdash_X (u;l) : \tau}{\Gamma_0 +_{j\in J} \Gamma_j \mid \_ \vdash_X \mathtt{L}'[\![\lambda x.v]\!][y\backslash s](u;l) : \tau}$$

— If $o = \mathtt{C}[\![xl]\!][x\backslash u] \to_{\mathtt{c}} \mathtt{C}[\![ul]\!][x\backslash u] = o'$, then by construction $\Sigma = \_$ and

$$\Phi' := \dfrac{\Phi_{\mathtt{C}[\![ul]\!]} \triangleright x : \{\!|\rho_j|\!\}_{j\in J}; \Pi \mid \_ \vdash_X \mathtt{C}[\![ul]\!] : \tau \qquad (\Phi_u^j \triangleright \Gamma_j \mid \_ \vdash_X u : \rho_j)_{j\in J}}{\Pi +_{j\in J} \Gamma_j \mid \_ \vdash_X \mathtt{C}[\![ul]\!][x\backslash u] : \tau}$$

Note that $|\mathtt{C}[\![ul]\!]|_x \geqslant 1$ thus $J \neq \varnothing$ if $X = \mathcal{S}_{\mathtt{E}}$. By Lemma 5.5, $\exists \Gamma_0, \exists I$, $\exists (\Gamma_i)_{i\in I}$, $\exists (\rho_i)_{i\in I}$ such that $x : \{\!|\rho_j|\!\}_{j\in J}; \Pi = \Gamma_0 +_{i\in I} \Gamma_i$, $\Phi_{\mathtt{C}[\![xl]\!]} \triangleright \Gamma_0 + \{x : \{\!|\rho_i|\!\}_{i\in I}\} \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \tau$ and $(\Phi_u^i \triangleright \Gamma_i \mid \_ \vdash_X u : \rho_i)_{i\in I}$. In particular, $I \neq \varnothing$ if $X = \mathcal{S}_{\mathtt{E}}$. By Lemma 4.1 and α-conversion, we necessarily have that $\Gamma_0 = x : \{\!|\rho_j|\!\}_{j\in J}; \Pi'$ such that $\Pi = \Pi' +_{i\in I} \Gamma_i$ thus $\Gamma_0 + \{x : \{\!|\rho_i|\!\}_{i\in I}\} = x : \{\!|\rho_l|\!\}_{k\in I\cup J}; \Pi'$. Let $K := I \cup J$. Hence,

$$\Phi := \dfrac{\Phi_{\mathtt{C}[\![xl]\!]} \triangleright x : \{\!|\rho_k|\!\}_{k\in K}; \Pi' \mid \_ \vdash_X \mathtt{C}[\![xl]\!] : \tau \qquad (\Phi_u^k \triangleright \Gamma_k \mid \_ \vdash_X u : \rho_k)_{k\in K}}{\Pi' +_{k\in K} \Gamma_k \mid \_ \vdash_X \mathtt{C}[\![xl]\!][x\backslash u] : \tau}$$

Observe that $\Pi' +_{k\in K} \Gamma_k = \Pi' +_{i\in I} \Gamma_i +_{j\in J} \Gamma_j = \Pi +_{j\in J} \Gamma_j$.

□

6. Characterization of linear-head E-normalization

In this section, we define **linear-head reduction** for the E-calculus by translating the corresponding notion in calculi with explicit substitutions specified in natural deduction style (Accattoli 2012). While linear-head reduction cannot be stated as a strategy of the λ-calculus, it can be simply stated as a particular restriction of the E-calculus; related to abstract machines (Danos and Regnier 2003) and linear logic (Girard 1987).

The main result of this section is a characterization of linear-head normalization by means of the typing system $\mathcal{H}_{\mathtt{E}}$.

First, we consider the set of **linear-head contexts** defined by the following grammar:

$$\mathtt{H} ::= \Box \mid \lambda x.\mathtt{H} \mid \mathtt{H}l \mid \mathtt{H}[x\backslash t]$$

Linear-head reduction, written $\to_{\mathtt{lh}}$, is the closure under *linear-head contexts* of the following rules:

$$\begin{array}{llllll} \mathtt{L}[\lambda x.t]\epsilon & \mapsto_{\mathtt{dB}_{\mathtt{nil}}} & \mathtt{L}[\lambda x.t] & \mathtt{L}[xl]m & \mapsto_{@_{\mathtt{var}}} & \mathtt{L}[x(l@m)] \\ \mathtt{L}[\lambda x.t](u;l) & \mapsto_{\mathtt{dB}_{\mathtt{cons}}} & \mathtt{L}[t[x\backslash u]l] & \mathtt{L}[tl]m & \mapsto_{@_{\mathtt{app}}} & \mathtt{L}[t(l@m)] \end{array}$$

$$\begin{array}{llll} \mathtt{H}^x[\![xl]\!][x\backslash u] & \mapsto_{\mathtt{lhc}} & \mathtt{H}^x[\![ul]\!][x\backslash u] & \text{if } |\mathtt{H}^x[\![xl]\!]|_x > 1 \\ \mathtt{H}^x[\![xl]\!][x\backslash u] & \mapsto_{\mathtt{lhd}} & \mathtt{H}^x[\![ul]\!] & \text{if } |\mathtt{H}^x[\![xl]\!]|_x = 1 \end{array}$$

Notice that there is no erasing rule, and that $\mathtt{dB}_{\mathtt{nil}}$, $\mathtt{dB}_{\mathtt{cons}}$, $@_{\mathtt{var}}$ and $@_{\mathtt{app}}$ are the same rules used to define E-reduction in Section 3. Rules `lhc` (resp. `lhd`) is the restriction of rule `c` (resp. `d`) to linear-head contexts. Therefore, the *leftmost* (i.e. *head*) occurrence of

a variable x in a term is substituted by u. This partial (i.e. *linear*) substitution is only performed on head occurrences. For instance, the following is a linear-head reduction sequence, where $s = \lambda z.z_\epsilon$:

$$(\lambda yx.x(x_\epsilon;\epsilon))(w_\epsilon;s;\epsilon) \to_{\mathtt{dB}_{\mathtt{cons}}} (\lambda x.x(x_\epsilon;\epsilon))[y\backslash w_\epsilon](s;\epsilon) \to_{\mathtt{dB}_{\mathtt{cons}}} (x(x_\epsilon;\epsilon))[x\backslash s][y\backslash w_\epsilon]\epsilon \to_{\mathtt{lhc}}$$
$$((\lambda z.z_\epsilon)(x_\epsilon;\epsilon))[x\backslash s][y\backslash w_\epsilon]\epsilon \to_{\mathtt{dB}_{\mathtt{cons}}} (z_\epsilon[z\backslash x_\epsilon]\epsilon)[x\backslash s][y\backslash w_\epsilon]\epsilon \to_{\mathtt{lhd}} ((x_\epsilon\epsilon)\epsilon)[x\backslash s][y\backslash w_\epsilon]\epsilon \to_{\mathtt{lhd}}$$
$$((((\lambda z.z_\epsilon)\epsilon)\epsilon)\epsilon)[y\backslash w_\epsilon]\epsilon \to^4_{\mathtt{dB}_{\mathtt{nil}}} (\lambda z.z_\epsilon)[y\backslash w_\epsilon]$$

Remark that the last four ($\mathtt{dB}_{\mathtt{nil}}$)-steps are not essential to reveal a head redex, i.e. the sequence of ϵ's above would not block the cut $[y\backslash w_\epsilon]$, if a head occurrence of y exists. But the normal-form $(\lambda z.z_\epsilon)[y\backslash w_\epsilon]$ is certainly more readable than $((((\lambda z.z_\epsilon)\epsilon)\epsilon)\epsilon)[y\backslash w_\epsilon]\epsilon$. A similar situation happens with rules $@_{\mathtt{var}}$ and $@_{\mathtt{app}}$, which cannot change the (head) status of redexes. So, even if rules $\mathtt{dB}_{\mathtt{nil}}$, $@_{\mathtt{var}}$ and $@_{\mathtt{app}}$ are not essential to define linear-head reductions in this framework, we prefer to keep them in order to obtain more readable normal forms in a sequent calculus notation.

The notion of lh-redex occurrence is defined as expected, according to the corresponding notion of redex occurrence given in Section 5. Therefore, since $\to_{\mathtt{lh}} \subseteq \to_{\mathtt{E}}$, both weighted subject reduction (c.f. Lemma 5.3) and subject expansion (Lemma 5.6) hold also for the reduction lh. Remark, however, that lh-normal forms are not necessarily E-normal forms. For instance, the term $t = (x(y\epsilon;\epsilon))[y\backslash \mathtt{I}]$ is in lh-nf but not in E-nf.

We now consider an inductive definition aiming to capture the set of E-terms containing linear-head redexes. This inductive set is denoted by $\neg\mathtt{lh}$ to emphasize the fact it contains terms which are *not* in lh-normal form. The auxiliary sets B@-red and cd-red denote, respectively, terms reducible by $\{\mathtt{dB}_{\mathtt{nil}}, \mathtt{dB}_{\mathtt{cons}}, @_{\mathtt{var}}, @_{\mathtt{app}}\}$ and $\{\mathtt{lhc}, \mathtt{lhd}\}$.

$$\frac{t \in \mathtt{B@\text{-}red}}{t \in \neg\mathtt{lh}} \qquad \frac{t \in \mathtt{cd\text{-}red}}{t \in \neg\mathtt{lh}}$$

$$\frac{}{\mathtt{L}[\lambda x.u]\epsilon \in \mathtt{B@\text{-}red}} \qquad \frac{}{\mathtt{L}[\lambda x.u](v;l) \in \mathtt{B@\text{-}red}} \qquad \frac{}{\mathtt{L}[xl]m \in \mathtt{B@\text{-}red}} \qquad \frac{}{\mathtt{L}[tl]m \in \mathtt{B@\text{-}red}}$$

$$\frac{u \in \mathtt{B@\text{-}red}}{ul \in \mathtt{B@\text{-}red}} \qquad \frac{u \in \mathtt{B@\text{-}red}}{u[x\backslash v] \in \mathtt{B@\text{-}red}} \qquad \frac{u \in \mathtt{B@\text{-}red}}{\lambda x.u \in \mathtt{B@\text{-}red}}$$

$$\frac{}{yl \in A_{yl}} \qquad \frac{u \in A_{yl}}{ul \in A_{yl}} \qquad \frac{u \in A_{yl}\ \&\ x \neq y}{\lambda x.u \in A_{yl}} \qquad \frac{u \in A_{yl}\ \&\ x \neq y}{u[x\backslash v] \in A_{yl}}$$

$$\frac{u \in A_{yl}}{u[y\backslash v] \in \mathtt{cd\text{-}red}} \qquad \frac{u \in \mathtt{cd\text{-}red}}{ul \in \mathtt{cd\text{-}red}} \qquad \frac{u \in \mathtt{cd\text{-}red}}{\lambda x.u \in \mathtt{cd\text{-}red}} \qquad \frac{u \in \mathtt{cd\text{-}red}}{u[x\backslash v] \in \mathtt{cd\text{-}red}}$$

Thus, in particular, $t \in \neg\mathtt{lh}$ implies $\mathtt{H}[t] \in \neg\mathtt{lh}$. Indeed, we can formally show the following equivalence:

Lemma 6.1. Let t be an E-term. Then, $t \notin$ lh-nf if and only if $t \in \neg\mathtt{lh}$.

Proof. We first prove the following more general statements:

1. $t = \mathtt{H}^y[\![yl]\!]$ if and only if $t \in A_{yl}$.
2. t is $\{\mathtt{lhc}, \mathtt{lhd}\}$-reducible if and only if $t \in \mathtt{cd\text{-}red}$.
3. Let $\mathtt{B@} = \{\mathtt{dB}_{\mathtt{nil}}, \mathtt{dB}_{\mathtt{cons}}, @_{\mathtt{var}}, @_{\mathtt{app}}\}$. Then, t is B@-reducible if and only if $t \in \mathtt{B@\text{-}red}$.

1. The left-to-right implication is by induction on $\mathtt{H}$ and the right-to-left implication is by induction on the context A_{yl}.
2. Let t be $\{\mathtt{lhc}, \mathtt{lhd}\}$-reducible. Then, $t = \mathtt{H}_0[\mathtt{H}_1^y[\![yl]\!][y\backslash u]]$ and it is then sufficient to show $\mathtt{H}_1^y[\![yl]\!][y\backslash u] \in \mathtt{cd}\text{-red}$. By point (1), we have $\mathtt{H}_1^y[\![yl]\!] \in A_{yl}$ so that $\mathtt{H}_1^y[\![yl]\!][y\backslash u] \in \mathtt{cd}\text{-red}$ holds by definition.
   Let $t \in \mathtt{cd}\text{-red}$. We reason by induction on $\mathtt{cd}$-red in order to specify t as $\mathtt{H}_0[t']$, with t' $\{\mathtt{lhc}, \mathtt{lhd}\}$-reducible, so that t itself is $\{\mathtt{lhc}, \mathtt{lhd}\}$-reducible. For the base case, $t = u[y\backslash v] \in \mathtt{cd}\text{-red}$ comes from $u \in A_{yl}$. By point (1) $u = \mathtt{H}_1^y[\![yl]\!]$, so that $\mathtt{H}_1^y[\![yl]\!][y\backslash v]$ is $\{\mathtt{lhc}, \mathtt{lhd}\}$-reducible (and $\mathtt{H}_0 = \Box$ in this case). All the inductive cases are straightforward.
3. Let t be $\mathtt{B@}$-reducible. Then, $t = \mathtt{H}_0[t']$, where t' is the left-hand side of some rule in $\mathtt{B@}$. It is then sufficient to show $t' \in \mathtt{B@}\text{-red}$, which is straightforward.
   Let $t \in \mathtt{B@}\text{-red}$. We reason by induction on $\mathtt{B@}$-red in order to specify t as $\mathtt{H}_0[t']$, with t' $\mathtt{B@}$-reducible, so that t itself is $\mathtt{B@}$-reducible. The base cases as well the inductive cases are straightforward.

Now, take $t \notin \mathtt{lh}\text{-nf}$. Then, t is $\mathtt{lh}$-reducible and by points (2) and (3) either $t \in \mathtt{cd}\text{-red}$ or $t \in \mathtt{B@}\text{-red}$, which implies $t \in \neg\mathtt{lh}$.

Conversely, if $t \in \neg\mathtt{lh}$, then either $t \in \mathtt{B@}\text{-red}$ or $t \in \mathtt{cd}\text{-red}$ and we conclude $t \notin \mathtt{lh}\text{-nf}$ by points (1) and (2). □

Similarly, an inductive definition of $\mathtt{E}$-terms in $\mathtt{lh}$-nf can be given as follows, where $|l|_; = n$ denotes the number of ";" in l:

$$\frac{u \in \mathcal{A}_y^n}{\lambda x.u \in \mathcal{A}_y^n} \qquad \frac{u \in \mathcal{A}_y^n \;\&\; x \neq y}{u[x\backslash v] \in \mathcal{A}_y^n} \qquad \frac{u \in \mathcal{B}_y^n}{u \in \mathcal{A}_y^n} \qquad \frac{u \in \mathcal{B}_y^n \;\&\; x \neq y}{u[x\backslash v] \in \mathcal{B}_y^n} \qquad \frac{|l|_; = n}{yl \in \mathcal{B}_y^n}$$

Indeed, we can show the following equivalence:

Lemma 6.2. Let t be an $\mathtt{E}$-term. Then, $t \in \mathtt{lh}\text{-nf}$ if and only if $t \in \mathcal{A}_y^n$ for some y and n.

Proof. The left-to-right implication is by induction on t. If $t = yl$, then $t \in \mathcal{B}_y^n \subseteq \mathcal{A}_y^n$ for some n. If $t = \lambda x.u$, then $u \in \mathtt{lh}\text{-nf}$ so that $u \in \mathcal{A}_y^n$ for some n by the *i.h.* This gives $\lambda x.u \in \mathcal{A}_y^n$ by the first rule. If $t = u[x/v]$, the same inductive reasoning applies. If $t = ul$, then u cannot be of the form $\mathtt{L}[\lambda x.u']$, otherwise it would be $\mathtt{dB}$-reducible. Then, $t = yl$, a case which was already treated.

The right-to-left implication holds by a straightforward induction on $\mathcal{A}_y^n$. □

A term t is **linear-head** $\mathtt{E}$**-normalizing** iff $t \in \mathcal{SN}(\mathtt{lh})$ [†].

The characterization of linear-head normalizing terms follows from the lemmas below.

Lemma 6.3. If $\Phi \triangleright \Gamma \mid \_ \vdash_{\mathcal{H}_\mathtt{E}} u : \tau$ and u has no $\mathtt{lh}$-redex T-occurrences in Φ, then $u \in \mathtt{lh}\text{-nf}$.

Proof. Let $\Phi \triangleright \Gamma \mid \_ \vdash_{\mathcal{H}_\mathtt{E}} u : \tau$ and suppose that $u \notin \mathtt{lh}\text{-nf}$. Then, $u \in \neg\mathtt{lh}$ by Lemma 6.1.

First, we prove a property stating that for any $u \in A_{yl}$, yl has a (head) T-occurrence in Φ. If $u = yl \in A_{yl}$, then the property is straightforward. If $u = vm \in A_{yl}$ or $u = \lambda x.v \in A_{yl}$

[†] Note that, since there are no erasing steps, weak and strong normalization coincides for $\mathtt{lh}$-reductions.

or $u = v[x\backslash v'] \in A_{yl}$ for $x \neq y$, where $v \in A_{yl}$, then by the *i.h.* yl has a T-occurrence in the corresponding subderivation of Φ and so it has a T-occurrence in Φ.

Second, we show that $u \in \neg\mathtt{lh}$ implies u has a $\mathtt{lh}$-redex T-occurrence in Φ, thus holding a contradiction. We reason by induction on the definition of $\neg\mathtt{lh}$. If $u = \mathtt{L}[\![\lambda x.v]\!]\epsilon \in \mathtt{B@}$-red, $u = \mathtt{L}[\![\lambda x.u']\!](v;m) \in \mathtt{B@}$-red, $u = \mathtt{L}[\![xl]\!]m \in \mathtt{B@}$-red or $u = \mathtt{L}[\![vl]\!]m \in \mathtt{B@}$-red, then ε is a $\mathtt{lh}$-redex T-occurrence in Φ. If $u = vm \in \mathtt{B@}$-red or $u = v[x\backslash v'] \in \mathtt{B@}$-red or $u = \lambda x.v \in \mathtt{B@}$-red, where $v \in \mathtt{B@}$-red, then by the *i.h.* the subterm v has a $\mathtt{lh}$-redex T-occurrence in the corresponding subderivation of Φ so that also u has a $\mathtt{lh}$-redex T-occurrence in Φ. Exactly the same reasoning applies for $u = vm$, or $u = v[x\backslash v']$ or $u = \lambda x.v$ belonging to $\mathtt{cd}$-red where $v \in \mathtt{cd}$-red. Finally, if $u = v[y\backslash v']$, where $v \in A_{yl}$, then by the first property shown before we know that yl has a (head) T-occurrence in the corresponding subderivation of Φ so that the redex $v[y\backslash v']$ has a (head) T-occurrence in Φ. This concludes the proof. □

Lemma 6.4. Let u be an $\mathtt{E}$-term. If u is liner-head $\mathtt{E}$-normalizing, then u is $\mathcal{H}_{\mathtt{E}}$-typable.

Proof. By induction on the length of the linear-head $\mathtt{E}$-normalizing reduction. Let $u \to_{\mathtt{lh}}^{k} u'$, where $u' \in \mathtt{lh}$-nf. If $k = 0$ (i.e. $u = u'$), then $u \in \mathcal{A}_y^n$, for some symbol y, by Lemma 6.2. Let $\tau^n = \mathcal{M}_1 \supset \cdots \supset \mathcal{M}_n \supset \tau$ $(n \geqslant 0)$ such that $\mathcal{M}_i = \{\!\!\{\ \}\!\!\}$ $(1 \leqslant i \leqslant n)$. We first prove by induction on $|l|_{;} = n$ that $\varnothing \mid \tau^n \vdash_{\mathcal{H}_{\mathtt{E}}} l : \tau$. If $n = 0$, then $l = \epsilon$ and $\varnothing \mid \tau \vdash_{\mathcal{H}_{\mathtt{E}}} \epsilon : \tau$ by the typing rule (ax). If $l = v;m$, then $\varnothing \mid \tau^n \vdash_{\mathcal{H}_{\mathtt{E}}} m : \tau$ by the *i.h.* and $\varnothing \mid \{\!\!\{\ \}\!\!\} \supset \tau^n \vdash_{\mathcal{H}_{\mathtt{E}}} v;m : \tau$ by the rule $(\supset \mathtt{l})$.

Second, we prove by induction on $\mathcal{B}_y^n$ that $u \in \mathcal{B}_y^n$ implies $y : \{\!\!\{\tau^n\}\!\!\} \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u : \tau$.

— If $yl \in \mathcal{B}_y^n$, then $\varnothing \mid \tau^n \vdash_{\mathcal{H}_{\mathtt{E}}} l : \tau$ by the previous point and $y : \{\!\!\{\tau^n\}\!\!\} \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} yl : \tau$ by the rule (hlist).

— If $u[x\backslash v] \in \mathcal{B}_y^n$ comes from $u \in \mathcal{B}_y^n$, then $y : \{\!\!\{\tau^n\}\!\!\} \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u : \tau$ holds by the *i.h.* thus $y : \{\!\!\{\tau^n\}\!\!\} \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u[x\backslash v] : \tau$ holds by the typing rule (cut).

Now, we prove by induction on $\mathcal{A}_y^n$ that $u \in \mathcal{A}_y^n$ implies $\Gamma \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u : \sigma$ where the domain of Γ has at most the symbol y.

— If $u \in \mathcal{A}_y^n$, where $u \in \mathcal{B}_y^n$, then the property follows by the previous point.

— If $\lambda x.u \in \mathcal{A}_y^n$, where $u \in \mathcal{A}_y^n$, then $\Gamma \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u : \sigma$ by the *i.h.* so that $\Gamma \setminus\!\setminus x \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} \lambda x.t : \Gamma(x) \supset \sigma$ by application of the typing rule $(\supset \mathtt{r})$. If Γ has at most y, then also does $\Gamma \setminus\!\setminus x$.

— If $u[x\backslash v] \in \mathcal{A}_y^n$, where $u \in \mathcal{A}_y^n$ and $x \neq y$, then $\Gamma \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u : \sigma$ by the *i.h.* so that $\Gamma \mid \_ \vdash_{\mathcal{H}_{\mathtt{E}}} u[x\backslash v] : \sigma$ by application of the typing rule (cut).

Otherwise, let $u \to_{\mathtt{lh}} v \to_{\mathtt{lh}}^{k} u'$. By the *i.h.* the term v is $\mathcal{H}_{\mathtt{E}}$-typable and thus by Lemma 5.6 the same holds for u. □

Theorem 6.1. Let u be an $\mathtt{E}$-term. Then, u is linear-head $\mathtt{E}$-normalizing iff u is $\mathcal{H}_{\mathtt{E}}$-typable.

Proof. The left-to-right implication holds by Lemma 6.4. For the right-to-left implication, let u be $\mathcal{H}_{\mathtt{E}}$-typable. By Corollary 5.1, and the fact that $\to_{\mathtt{lh}} \subseteq \to_{\mathtt{E}}$, the strategy consisting in contracting $\mathtt{lh}$-redex T-occurrences terminates in a term u' without such redexes. Moreover, u' is $\mathcal{H}_{\mathtt{E}}$-typable by Lemma 5.3. Then, u' turns out to be in $\mathtt{lh}$-nf by Lemma 6.3. Thus, u is head-linear $\mathtt{E}$-normalizing. □

7. Characterization of strong $\mathtt{E}$-normalization

This section is devoted to the characterization of $\mathtt{E}$-strong normalization. The structure of the proof runs along the following lines. We give an alternative definition of the set $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, that we denote by $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, where $\mathtt{E}\backslash\mathtt{w}$ is the non-erasing reduction relation $\rightarrow_{\mathtt{E}} \setminus \rightarrow_{\mathtt{w}}$. We show this new alternative definition to be equivalent to the original one, i.e. $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ iff $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ (Lemma 7.5). We then use this equivalence to obtain the 'typable iff normalizing' property as follows:

— $\mathcal{S}_{\mathtt{E}}$-typability implies $\mathtt{E}\backslash\mathtt{w}$-normalization (Corollary 5.2), implying $\mathtt{E}$-normalization (Lemma 7.7).
— $\mathtt{E}$-normalization trivially implies $\mathtt{E}\backslash\mathtt{w}$-normalization, and every object in $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ is $\mathcal{S}_{\mathtt{E}}$-typable (Lemma 7.6). The equivalence in Lemma 7.5 allows to conclude.

7.1. *An alternative definition of* $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$

We give an alternative definition of $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$. Indeed, the **inductive set of $\mathtt{E}\backslash\mathtt{w}$-strongly normalizing objects**, written $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, is the smallest subset of objects satisfying the following properties:

(EL) $\epsilon \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(NEL) If $t, l \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $t; l \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$,
(L) If $t \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $\lambda x.t \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(HL) If $l \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $xl \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(W) If $t, s \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ and $|t|_x = 0$, then $t[x\backslash s] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
$(\mathtt{dB}_{\mathtt{nil}})$ If $(\lambda x.t)l_1 \dots l_n$ $(n \geqslant 0) \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $(\lambda x.t)\epsilon l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
$(\mathtt{dB}_{\mathtt{cons}})$ If $t[x\backslash u]ml_1 \dots l_n$ $(n \geqslant 0) \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $(\lambda x.t)(u; m)l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
$(@_{\mathtt{var}})$ If $x(m_1 @ m_2)l_1 \dots l_n$ $(n \geqslant 0) \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $(xm_1)m_2 l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
$(@_{\mathtt{app}})$ If $t(m_1 @ m_2)l_1 \dots l_n$ $(n \geqslant 0) \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, then $(tm_1)m_2 l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(C) If $\mathtt{C}[\![u\, l]\!][x\backslash u] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ and $|\mathtt{C}[\![x\, l]\!]|_x > 1$, then $\mathtt{C}[\![x\, l]\!][x\backslash u] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(D) If $\mathtt{C}[\![u\, l]\!] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ and $|\mathtt{C}[\![x\, l]\!]|_x = 1$, then $\mathtt{C}[\![x\, l]\!][x\backslash u] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.
(E) If $(tl)[x\backslash s] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ and $|l|_x = 0$, then $t[x\backslash s]l \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.

In order to show that $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ and $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ are equivalent sets, we first introduce a technical tool. Indeed, we do not want to distinguish terms having explicit cuts at different *head positions*, mainly because they do have exactly the same maximal reduction lengths. More precisely, the **head graphical equivalence** $\sim$ on $\mathtt{E}$-terms, inspired from the σ-equivalence on λ-terms (Regnier 1994) and the σ-equivalence on λ-terms with explicit substitutions (Accattoli and Kesner 2010), is given by the contextual, transitive, symmetric and reflexive closure of the following axiom:

$$(tl)[x\backslash u] \approx t[x\backslash u]l, \text{ where } |l|_x = 0$$

Notice that $(xl)[x\backslash u]$ cannot be $\sim$-converted into $x[x\backslash u]l$ when $x \notin \mathtt{fv}(l)$, since x alone is not a term of the calculus. The key property concerning this equivalence relation is stated below, where $\eta_{\mathcal{R}}(o)$ denotes the maximal length of an $\mathcal{R}$-reduction sequences starting at o.

Lemma 7.1 (Invariance for $\sim$). Let o, o' be $\mathtt{E}$-objects such that $o \sim o'$. Then,

1. $o \to_{\mathtt{E}\mathtt{w}} o_0$, iff $o' \to_{\mathtt{E}\mathtt{w}} o'_0$, where $o_0 \sim o'_0$. In particular, $\eta_{\mathtt{E}\mathtt{w}}(o) = \eta_{\mathtt{E}\mathtt{w}}(o')$.
2. $\Phi \triangleright \Gamma \vdash_{\mathcal{S}_{\mathtt{E}}} o : \tau$, iff $\Phi' \triangleright \Gamma \vdash_{\mathcal{S}_{\mathtt{E}}} o' : \tau$. Moreover, $\mathtt{sz2}(\Phi) = \mathtt{sz2}(\Phi')$.

Proof.

1. The proof consists in showing that $o \to_{\mathtt{E}\mathtt{w}} o'$ and $o \sim o_0$ implies there is o'_o such that $o_0 \to_{\mathtt{E}\mathtt{w}} o'_0$ and $o' \sim o'_0$. The proof is by induction on $o \to_{\mathtt{E}\mathtt{w}} o'$ and case analysis of $o \sim o_0$.
2. The proof is by induction on $o \sim o'$ and is straightforward. □

We now establish some properties of $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ which are used to show that the set $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ is indeed equal to $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.

Lemma 7.2. Let m, l be two E-lists. Then, $m, l \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ iff $m@l \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

Proof. By induction on m, knowing that $m = t; m' \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ iff $t, m' \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$. □

Lemma 7.3. Let $\mathtt{V}$ be an E-list context, t be an E-term and m, l be E-lists.

1. $\mathtt{V}[\![tl]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ implies $\mathtt{V}[\![t]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.
2. $\mathtt{V}[\![t(m@l)]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ implies $\mathtt{V}[\![tm]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

Proof. The proof of the first point is by induction on $\eta_{\mathtt{E}}(\mathtt{V}[\![tl]\!])$ and the proof of the second one is by induction on $\eta_{\mathtt{E}}(\mathtt{V}[\![t(m@l)]\!])$ and uses the first point. □

Let o be an E-object such that $|o|_x = n$. If $|o|_y = 0$, i.e. if y is fresh in o, then we write $o_{[x\backslash y]}$ to denote an arbitrary nondeterministic replacement of i $(0 \leqslant i \leqslant n)$ occurrences of x by the fresh symbol y. Thus, for example, if $o = x\epsilon[z\backslash x\epsilon]$, then $o_{[x\backslash y]}$ may denote one of the terms $x\epsilon[z\backslash x\epsilon]$, $y\epsilon[z\backslash x\epsilon]$, $x\epsilon[z\backslash y\epsilon]$, or $y\epsilon[z\backslash y\epsilon]$.

Lemma 7.4. Let o be a E-object and v be a E-term. If $o \to_{\mathtt{E}\mathtt{w}} o'$, then $o_{[x\backslash y]} \to_{\mathtt{E}\mathtt{w}} o'_{[x\backslash y]}$ and $o_{[x\backslash y]}\{y/v\} \to_{\mathtt{E}\mathtt{w}} o'_{[x\backslash y]}\{y/v\}$.

A consequence of the previous lemma is that $o \to_{\mathtt{E}\mathtt{w}} o'$ implies $o\{x/v\} \to_{\mathtt{E}\mathtt{w}} o'\{x/v\}$.

Corollary 7.1. Let o be a E-object and v be a E-term. If $o_{[x\backslash y]}\{y/v\} \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

A consequence of the previous corollary is that $C[\![vl]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ implies $C[\![xl]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, a property which will be used in the forthcoming key Lemma 7.5.

Although the inclusion $\mathcal{ISN}(\mathtt{E}\backslash\mathtt{w}) \subseteq \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ can be obtained through typability in $\mathcal{S}_{\mathtt{E}}$, we prefer to present here a direct (and self-contained) proof of this inclusion.

Lemma 7.5. $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w}) = \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.

Proof. If $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then one shows $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by induction on $\langle \eta_{\mathtt{E}\mathtt{w}}(o), |o| \rangle$. For the converse one reasons by induction on the definition of $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$. Full details can be found in the Appendix. □

7.2. *The main proof*

This section uses Lemma 7.5 to show the characterization of E-normalization as explained before. More precisely,

Lemma 7.6. Let o be an E-object. If $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then o is $\mathcal{S}_{\mathtt{E}}$-typable.

Proof. By induction on the structure of $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w}) =_{\text{Lemma 7.5}} \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

— If $o = \epsilon$, $o = t;l$, $o = \lambda x.t$, $o = xl$, or $o = u[x\backslash v]$ with $|u|_x = 0$, then the proof is straightforward by using the *i.h.*
— If $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ comes from one of the rules $(\mathtt{dB}_{\mathtt{nil}})$, $(\mathtt{dB}_{\mathtt{cons}})$, (C), (D), $(@_{\mathtt{var}})$ or $(@_{\mathtt{app}})$, then the property holds by the *i.h.* and the subject expansion Lemma 5.6.
— If $t[x\backslash s]l \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ comes from the rule (E), then $(tl)[x\backslash s] \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$, so that $(tl)[x\backslash s]$ is $\mathcal{S}_{\mathtt{E}}$-typable by the *i.h.* and the property holds by Lemma 7.1.2. □

Strong E-normalization can be now obtained from strong E\w-normalization as follows:

Lemma 7.7 (From E\w to E). Let o be an E-object. If $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then $o \in \mathcal{SN}(\mathtt{E})$.

Proof. One first shows a postponement property for w-reduction steps given by: if $o \to_{\mathtt{w}}^{+} \to_{\mathtt{E}\backslash\mathtt{w}} o'$, then $o \to_{\mathtt{E}\backslash\mathtt{w}} \to_{\mathtt{w}}^{+} o'$. Then the property is proved by contradiction using the postponement property and the fact that w-reduction is itself terminating. □

We can now conclude with the main result of this section.

Theorem 7.1. Let o be an E-object. Then o is $\mathcal{S}_{\mathtt{E}}$-typable iff $o \in \mathcal{SN}(\mathtt{E})$.

Proof. Let o be $\mathcal{S}_{\mathtt{E}}$-typable. Then $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ by Corollary 5.2 and $o \in \mathcal{SN}(\mathtt{E})$ by Lemma 7.7. For the converse, $o \in \mathcal{SN}(\mathtt{E}) \subseteq \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ because $\to_{\mathtt{E}\backslash\mathtt{w}} \subseteq \to_{\mathtt{E}}$. We conclude by Lemma 7.6. □

8. The I-calculus

We introduce the syntax and the operational semantics of the I-calculus, slightly differently defined in Espírito Santo (2000). The I-calculus can be obtained from E by an appropriate projection function (c.f. Lemma 8.3).

Given a countable infinite set of symbols $x, y, z, \ldots$, three syntactic categories are defined by the following grammars:

$$\begin{array}{ll} \textbf{(I-objects)} & o ::= t \mid l \\ \textbf{(I-terms)} & t, u, v ::= xl \mid \lambda x.t \mid (\lambda x.t)l \\ \textbf{(I-lists)} & l, m ::= \epsilon \mid t;l \end{array}$$

Remark that general terms of the form tl are not I-terms.

As before, we work with Barendregt's convention and the standard notion of α-conversion. **I-contexts** are defined as E-contexts restricted to I-objects, specified by the following specialized grammar:

$$\begin{array}{lll} \textbf{(I-object contexts)} & \mathtt{O_I} & ::= \mathtt{C_I} \mid \mathtt{V_I} \\ \textbf{(I-term contexts)} & \mathtt{C_I} & ::= \Box \mid x\mathtt{V_I} \mid \lambda x.\mathtt{C_I} \mid (\lambda x.\mathtt{C_I})l \mid (\lambda x.t)\mathtt{V_I} \\ \textbf{(I-list contexts)} & \mathtt{V_I} & ::= \Box \mid \mathtt{C_I}; l \mid t; \mathtt{V_I} \end{array}$$

The **reduction relation** $\rightarrow_{\mathtt{I}}$ is given by the closure of contexts $\mathtt{O_I}$ of the following rules:

$$(\lambda x.t)\epsilon \mapsto_{\beta_\epsilon} \lambda x.t \qquad (\lambda x.t)(u;l) \mapsto_{\beta_{\mathrm{cons}}} t\{x\backslash u\} \circ l$$

where the operations $\_\circ\_$ and $\{\_\backslash\_\}$ are defined as follows:

$$\begin{array}{lll} (xl)\circ m & := & x\ (l@m) \\ ((\lambda y.t)l)\circ m & := & (\lambda y.t)(l@m) \\ (\lambda x.t)\circ m & := & (\lambda x.t)m \end{array} \qquad \begin{array}{lll} \epsilon\{x\backslash v\} & := & \epsilon \\ (u;l)\{x\backslash v\} & := & u\{x\backslash v\}; l\{x\backslash v\} \\ (y\ l)\{x\backslash v\} & := & yl\{x\backslash v\} \\ (xl)\{x\backslash v\} & := & v\ \circ l\{x\backslash v\} \\ ((\lambda y.t)l)\{x\backslash v\} & := & (\lambda y.t\{x\backslash v\})l\{x\backslash v\} \\ (\lambda y.t)\{x\backslash v\} & := & \lambda y.t\{x\backslash v\} \end{array}$$

The substitution operator $\{\_\backslash\_\}$ is defined on α-equivalence classes of terms in order to avoid the capture of free variables. Notice that substitution distributes with respect to @ and $\circ$, i.e. one can show that $(t@l)\{x\backslash u\} = t\{x\backslash u\}@l\{x\backslash u\}$ and $(t\circ l)\{x\backslash u\} = t\{x\backslash u\}\circ l\{x\backslash u\}$. As expected, the I-calculus enjoys confluence:

Theorem 8.1 (Confluence). The reduction relation $\rightarrow_{\mathtt{I}}$ is confluent.

Proof. A self-contained proof can be done by the Tait–Martin Löf technique. The proof is quite standard but we list below the principal steps to follow:

— We define a simultaneous reduction relation $\Rrightarrow$:
  - $o \Rrightarrow o$.
  - $t \Rrightarrow t'$ implies $\lambda x.t \Rrightarrow \lambda x.t'$.
  - $l \Rrightarrow l'$ implies $xl \Rrightarrow xl'$.
  - $t \Rrightarrow t'$ and $l \Rrightarrow l'$ imply $t;l \Rrightarrow t';l'$.
  - $t \Rrightarrow t'$, $u \Rrightarrow u'$ and $l \Rrightarrow l'$ imply $(\lambda x.t)(u;l) \Rrightarrow t'\{x\backslash u'\}\circ l'$.

— We show that $\Rrightarrow$ verifies the diamond property. This is done by induction on the relation $\Rrightarrow$ and a detailed analysis of cases. Thus, we conclude $\Rrightarrow$ is confluent by Baader and Nipkow (1998).

— We easily show that the relations $\Rrightarrow^*$ and $\rightarrow_{\mathtt{I}}^*$ are equal. So that we conclude that $\rightarrow_{\mathtt{I}}$ is confluent. □

An **erasing step** is the closure by contexts of the reduction rule $(\lambda x.t)(u;l) \mapsto t\{x\backslash u\}\circ l$, where $x \notin \mathtt{fv}(t)$, i.e. an erasing step discards the argument u since $x \notin \mathtt{fv}(t)$ implies $t\{x\backslash u\} = t$. Notice that erasing steps cannot be postponed, so that we cannot apply to the I-calculus the same (simple) proof technique used in Section 7 to characterize E-strong normalization. An example of non-erasing step is $(\lambda y.y_\epsilon)(x_\epsilon;x_\epsilon;\epsilon) \rightarrow x(x_\epsilon;\epsilon)$ while $(\lambda y.z_\epsilon)(x_\epsilon;x_\epsilon;\epsilon) \rightarrow z(x_\epsilon;\epsilon)$ is an erasing step. Non-erasing steps play a key role in Lemma 9.4.

The I-calculus can be simulated in the E-calculus in terms of more atomic steps. Indeed,

Lemma 8.1. Let o be an $\mathtt{I}$-term. If $o \to_{\mathtt{I}} o'$, then $o \to_{\mathtt{E}}^{+} o'$.

Proof. One first shows that for all $\mathtt{I}$-objects t, u, l, $t[x\backslash u] \to_{\mathtt{E}}^{*} t\{x\backslash u\}$ and $tl \to_{\mathtt{E}}^{*} t \circ l$. The proof then proceeds by induction on $\mathtt{I}$-reduction. The interesting case is when $o = (\lambda x.t)(u; l) \to_{\mathtt{I}} t\{x\backslash u\} \circ l = o'$, for which we conclude by $o = (\lambda x.t)(u; l) \to_{\mathtt{E}} t[x\backslash u]l \to_{\mathtt{E}}^{*} t\{x\backslash u\}l \to_{\mathtt{E}}^{*} t\{x\backslash u\} \circ l = o'$ using the properties mentioned above. □

A direct consequence is the following property, to be used later in Section 10.1.

Corollary 8.1. Let o be an $\mathtt{I}$-term. Then, $o \in \mathcal{SN}(\mathtt{E})$ implies $o \in \mathcal{SN}(\mathtt{I})$.

Reciprocally, the $\mathtt{E}$-calculus can be projected into the $\mathtt{I}$-calculus. For that, we first remark that the system $\mathtt{sub} = \{\mathtt{w}, \mathtt{d}, \mathtt{c}, @_{\mathtt{var}}, @_{\mathtt{app}}\}$ is locally confluent and terminating. Hence, $\mathtt{sub}$-normal forms of objects are unique; we thus write $\mathtt{sub}(o)$ for the $\mathtt{sub}$-normal form of an object o. Remark that $\mathtt{sub}(o)$ is an $\mathtt{I}$-object for every $\mathtt{E}$-object o.

First, some auxiliary lemmas in order to investigate the projection mentioned above.

Property 8.1. Let $\mathtt{O}_{\mathtt{I}}[\![xl]\!]$, t, u be $\mathtt{I}$-terms. Then,

1. $\mathtt{O}_{\mathtt{I}}[\![xl]\!]\{x\backslash u\} = \mathtt{O}_{\mathtt{I}}[\![u \circ l]\!]\{x\backslash u\}$
2. $t\{x\backslash u\} = t_{[p\backslash y]}\{y\backslash u\}\{x\backslash u\}$, where p is any position of t having a free occurrence of x, and $t_{[p\backslash y]}$ is the term obtained by replacing the free occurrence of x at position p by a fresh variable y.

Proof. The first point can be shown by induction on contexts and the second one by induction on terms. □

Lemma 8.2. Let t be an $\mathtt{I}$-term, l be an $\mathtt{E}$-list and u be an $\mathtt{E}$-term. Then,

1. $\mathtt{sub}(t[x\backslash u]) = t\{x\backslash \mathtt{sub}(u)\}$
2. $\mathtt{sub}(tl) = t \circ \mathtt{sub}(l)$

Proof. The proof is by simultaneous induction on the pair $\langle |t|_x, |t| \rangle$. It uses the fact that $\mathtt{sub}(l@m) = \mathtt{sub}(l)@\mathtt{sub}(m)$, as well as Proposition 8.1. □

Corollary 8.2. Let t, u be $\mathtt{E}$-terms and l be an $\mathtt{E}$-list. Then,

1. $\mathtt{sub}(t[x\backslash u]) = \mathtt{sub}(t)\{x\backslash \mathtt{sub}(u)\}$
2. $\mathtt{sub}(tl) = \mathtt{sub}(t) \circ \mathtt{sub}(l)$

Proof. As noticed before $\mathtt{sub}(t)$ is an $\mathtt{I}$-term. Then, we have

$$\mathtt{sub}(t[x\backslash u]) = \mathtt{sub}(\mathtt{sub}(t)[x\backslash u]) =_{\text{Lemma 8.2}} \mathtt{sub}(\mathtt{sub}(t))\{x\backslash \mathtt{sub}(u)\} = \mathtt{sub}(t)\{x\backslash \mathtt{sub}(u)\}$$
$$\mathtt{sub}(tl) = \mathtt{sub}(\mathtt{sub}(t)l) =_{\text{Lemma 8.2}} \mathtt{sub}(\mathtt{sub}(t)) \circ \mathtt{sub}(l) = \mathtt{sub}(t) \circ \mathtt{sub}(l)$$

□

Lemma 8.3 (Projection). Let o be an $\mathtt{E}$-term. If $o \to_{\mathtt{E}} o'$, then $\mathtt{sub}(o) \to_{\mathtt{I}}^{*} \mathtt{sub}(o')$.

Proof. By induction on $\mathtt{E}$-reductions using Corollary 8.2. □

Using confluence of $\mathtt{I}$, Lemmas 8.1 and 8.3, we obtain the following property.

Corollary 8.3. The reduction relation $\to_{\mathtt{E}}$ is confluent.

$$\frac{}{\emptyset \mid \tau \vdash \epsilon : \tau}\ (\mathtt{ax}) \qquad \frac{\Gamma \mid \_ \vdash t : \tau}{\Gamma \setminus\!\setminus x \mid \_ \vdash \lambda x.t : \Gamma(x) \supset \tau}\ (\supset \mathtt{r})$$

$$\frac{\Gamma \mid \sigma \vdash l : \tau}{\Gamma + \{x{:}\{\!\{\sigma\}\!\}\} \mid \_ \vdash xl : \tau}\ (\mathtt{hlist}) \qquad \frac{\Gamma \mid \_ \vdash t : \rho \qquad \Delta \mid \tau \vdash l : \sigma}{\Delta + \Gamma \mid \{\!\{\ \}\!\} \supset \tau \vdash t; l : \sigma}\ (\supset \mathtt{l}_{\notin})$$

$$\frac{(\Gamma_j \mid \_ \vdash t : \tau_j)_{j \in J} \quad J \neq \emptyset \qquad \Delta \mid \tau \vdash l : \sigma}{\Delta +_{j \in J} \Gamma_j \mid \{\!\{\tau_j\}\!\}_{j \in J} \supset \tau \vdash t; l : \sigma}\ (\supset \mathtt{l}_{\in})$$

$$\frac{\Gamma \mid \_ \vdash \lambda x.t : \sigma \qquad \Delta \mid \sigma \vdash l : \tau}{\Gamma + \Delta \mid \_ \vdash (\lambda x.t)l : \tau}\ (\mathtt{app})$$

Fig. 3. The type system $\mathcal{S}_{\mathtt{I}}$ for the I-calculus.

9. Typing system $\mathcal{S}_{\mathtt{I}}$ for I-terms and its properties

In this section, we restrict to I-objects the system $\mathcal{S}_{\mathtt{E}}$ introduced in Section 4. The resulting system $\mathcal{S}_{\mathtt{I}}$ is given in Figure 3. As before, relevance holds for I-objects.

Lemma 9.1. If $\Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathtt{I}}} o : \tau$, then $\mathtt{dom}(\Gamma) = \mathtt{fv}(o)$.

The proof of 'typable implies normalizing' in Section 10 is obtained through the corresponding property for $\mathcal{S}_{\mathtt{E}}$ and the behaviour of projections established in Lemma 8.3. Therefore, no *forward* properties are needed (nor investigated) in the present section. For the converse, i.e. 'normalizing implies typable,' a self-contained proof is developed, thus the necessity of *backward* properties, such as the forthcoming reverse substitution lemma (c.f. Lemma 9.3).

First, we establish the *backward* property for the $\_ \circ \_$ operation.

Lemma 9.2. If $\Phi_{t \circ l} \rhd \Gamma \mid \_ \vdash t \circ l : \tau$, then there exist Φ_t, Φ_l and σ such that $\Gamma = \Gamma_t + \Gamma_l$, $\Phi_t \rhd \Gamma_t \mid \_ \vdash t : \sigma$ and $\Phi_l \rhd \Gamma_l \mid \sigma \vdash l : \tau$.

Proof. By induction on $\mathtt{sz2}(\Phi_{t \circ l})$ and the structure(s) of t and l, using Lemma 5.4 on pure terms. □

Lemma 9.3 (Reverse substitution). If $x \in \mathtt{fv}(o)$ and $\Phi \rhd_{\mathcal{S}_{\mathtt{I}}} \Gamma \mid \Sigma \vdash o\{x/u\} : \tau$, then there exist $(\Phi^i_u \rhd_{\mathcal{S}_{\mathtt{I}}} \Gamma_i \vdash u : \sigma_i)_{i \in I}$ and $\Phi_o \rhd_{\mathcal{S}_{\mathtt{I}}} x : \{\!\{\sigma_i\}\!\}_{i \in I}; \Gamma_o \mid \Sigma \vdash o : \tau$ for $I \neq \varnothing$ and $\Gamma = \Gamma_o +_{i \in I} \Gamma_i$.

Proof. By induction on the structure of o. We only show the most interesting case.

If $o = yl$, then there are two possibilities.

If $y \neq x$, then $o\{x/u\} = y\, l\{x/u\}$, thus $\Sigma = \_$ and Φ is of the form:

$$\frac{\Phi_{l'} \rhd \Pi_{l'} \mid \sigma \vdash l\{x/u\} : \tau}{\Pi_{l'} + \{y : \{\!\{\sigma\}\!\}\} \vdash y\, l\{x/u\} : \tau}$$

By the *i.h.* $\Pi_{l'} = \Pi_l +_{i \in I} \Pi_i$ and $\Phi_l \rhd x : \{\!\{\sigma_i\}\!\}_{i \in I}; \Pi_l \mid \sigma \vdash l : \tau$ and $(\Phi^i_u \rhd \Pi_i \mid \_ \vdash u : \sigma_i)_{i \in I}$. Then, $\Phi_o \rhd x : \{\!\{\sigma_i\}\!\}_{i \in I}; \Pi_l + \{y : \{\!\{\sigma\}\!\}\} \mid \_ \vdash yl : \tau$ by the rule (hlist) and the result then holds for $\Gamma_o := \Pi_l + \{y : \{\!\{\sigma\}\!\}\}$ and $(\Gamma_i := \Pi_i)_{i \in I}$.

If $y = x$, then $o\{x/u\} = u \circ l\{x/u\}$ and, by Lemma 9.2, $\Sigma = \_$, $\Gamma = \Pi_u + \Pi_{l'}$ such that $\Phi_u \rhd \Pi_u \mid \_ \vdash u : \sigma_u$ and $\Phi_{l'} \rhd \Pi_{l'} \mid \sigma_u \vdash l\{x/u\} : \tau$.

Suppose $x \in \mathtt{fv}(l)$. Then, by the *i.h.* $\Pi_{l'} = \Pi_l +_{j\in J} \Pi_j$ and $\Phi_l \rhd x : \{\!|\sigma_j|\!\}_{j\in J}; \Pi_l \mid \sigma_u \vdash l : \tau$ and $(\Phi^i_u \rhd \Pi_j \mid \_ \vdash u : \sigma_j)_{j\in J}$. Let $I := J \cup \{u\}$, where we suppose w.l.o.g. that $u \notin J$. Then, $\Phi_o \rhd x : \{\!|\sigma_i|\!\}_{i\in I}; \Pi_l \mid \_ \vdash xl : \tau$ by the rule (`hlist`) and the result then holds for $(\Gamma_i := \Pi_i)_{i\in I}$ and $\Gamma_o := \Pi_l$.

Suppose $x \notin \mathtt{fv}(l)$. Then, $J = \varnothing$ and thus $\Phi_o \rhd x : \{\!|\sigma_u|\!\}; \Pi_l \mid \_ \vdash xl : \tau$ holds by rule (`hlist`). We have the derivation Φ_u so the property holds for a singleton I. □

Lemma 9.4 (Subject expansion for non-erasing reductions of the `I`-calculus). If $\Phi' \rhd_{\mathcal{S}_\mathtt{I}} \Gamma \mid \Sigma \vdash o' : \tau$ and $o \to_\mathtt{I} o'$ is a non-erasing step, then there exists Φ such that $\Phi \rhd_{\mathcal{S}_\mathtt{I}} \Gamma \mid \Sigma \vdash o : \tau$.

Proof. By induction on the non-erasing reduction relation $\to_\mathtt{I}$. We only present the proofs for both β_ϵ and $\beta_\mathtt{cons}$ rules. The remaining inductive cases are straightforward.

— If $o = (\lambda x.t)\epsilon \to_{\beta_\epsilon} \lambda x.t = o'$, then $\Sigma = \_$ and $\Phi \rhd \Gamma \mid \_ \vdash (\lambda x.t)\epsilon : \tau$ by the application of the (`ax`) and (`app`) rules.

— If $o = (\lambda x.t)(u;l) \to_{\beta_\mathtt{cons}} t\{x/u\} \circ l = o'$, where $x \in \mathtt{fv}(t)$, then by Lemma 9.2, we have that $\Sigma = \_$, $\Gamma = \Gamma_{t'} + \Gamma_l$ and there is σ such that $\Phi_{t'} \rhd \Gamma_{t'} \mid \_ \vdash t\{x/u\} : \sigma$ and $\Phi_l \rhd \Gamma_l \mid \sigma \vdash l : \tau$. By Lemma 9.3, $\Gamma_{t'} = \Gamma_0 +_{i\in I} \Gamma_i$, where $I \neq \varnothing$, and $\Phi_t \rhd x : \{\!|\rho_i|\!\}_{i\in I}; \Gamma_0 \mid \_ \vdash t : \sigma$ and $(\Phi^i_u \rhd \Gamma_i \mid \_ \vdash u : \rho_i)_{i\in I}$. Then,

$$\Phi := \dfrac{\dfrac{\Phi_t \rhd x : \{\!|\rho_i|\!\}_{i\in I}; \Gamma_0 \mid \_ \vdash t : \sigma}{\Gamma_0 \mid \_ \vdash \lambda x.t : \{\!|\rho_i|\!\}_{i\in I} \supset \sigma} \qquad \dfrac{(\Phi^i_u \rhd \Gamma_i \mid \_ \vdash u : \rho_i)_{i\in I} \qquad \Phi_l \rhd \Gamma_l \mid \sigma \vdash l : \tau}{\Gamma_l +_{i\in I} \Gamma_i \mid \{\!|\rho_i|\!\}_{i\in I} \supset \sigma \vdash u; l : \tau}}{\Gamma_0 + \Gamma_l +_{i\in I} \Gamma_i \mid \_ \vdash (\lambda x.t)(u;l) : \tau}$$

□

10. Characterization of strong `I`-normalization

The characterization of $\mathcal{SN}(\mathtt{I})$ through typability in $\mathcal{S}_\mathtt{I}$ is obtained proving the two implications separately, as in the characterization of $\mathcal{SN}(\mathtt{E})$ developed in Section 7. However, the proofs do not follow the same scheme.

The proof of 'typability implies normalization,' instead of a direct result obtained by means of a weighted subject reduction property, is established by a combination of the respective result in system $\mathcal{S}_\mathtt{E}$ (c.f. Thm. 7.1) and the Projection Lemma 8.1.

On the other hand, in order to obtain the proof of 'normalization implies typability' using the corresponding result for $\mathcal{S}_\mathtt{E}$, one would need the *preservation of strong normalization* (PSN) property, stated as '$o \in \mathcal{SN}(\mathtt{E})$ implies $o \in \mathcal{SN}(\mathtt{I})$.' In general, the PSN property is not easy to prove, so we choose to concentrate on typability properties by developing an alternative self-contained proof argument.

10.1. *Typability implies strong* `I`*-normalization*

In order to characterize the set $\mathcal{SN}(\mathtt{I})$ of strongly `I`-normalizing objects by means of $\mathcal{S}_\mathtt{I}$-typability, we first need to show that every $\mathcal{S}_\mathtt{I}$-typable object is strongly `I`-normalizing. This is obtained as follows:

Corollary 10.1. If $\Phi \rhd_{\mathcal{S}_\mathtt{I}} \Gamma \mid \Sigma \vdash o : \tau$, then $o \in \mathcal{SN}(\mathtt{I})$.

Proof. If o is $\mathcal{S}_{\mathtt{I}}$-typable, then o is also trivially $\mathcal{S}_{\mathtt{E}}$-typable. Theorem 7.1 gives $o \in \mathcal{SN}(\mathtt{E})$ and Corollary 8.1 gives $o \in \mathcal{SN}(\mathtt{I})$. □

10.2. *Strong* I*-normalization implies typability*

This last part of the paper completes the characterization result for the I-calculus, namely, we show that every strongly I-normalizing object is $\mathcal{S}_{\mathtt{I}}$-typable, so that, combined with Corollary 10.1, we obtain a full characterization of strongly I-normalizing objects by means of $\mathcal{S}_{\mathtt{I}}$-typability.

In order to achieve the main result of this section, we define an inductive set of objects $\mathcal{ISN}(\mathtt{I})$ containing the set of strongly I-normalizing objects and contained in the set of $\mathcal{S}_{\mathtt{I}}$-typable ones. The set $\mathcal{ISN}(\mathtt{I})$ is inspired by the idempotent intersection typing system proposed by Valentini (2001), then revisited by Kikuchi (2014).

We start by defining the set $\mathcal{ISN}(\mathtt{I})$ as the smallest subset of I-objects satisfying the following properties:

($\mathtt{ax}$) $\epsilon \in \mathcal{ISN}(\mathtt{I})$.
($\supset \mathtt{r}$) If $t \in \mathcal{ISN}(\mathtt{I})$, then $\lambda x.t \in \mathcal{ISN}(\mathtt{I})$.
($\mathtt{hlist}$) If $l \in \mathcal{ISN}(\mathtt{I})$, then $xl \in \mathcal{ISN}(\mathtt{I})$.
($\supset \mathtt{l}_{\mathcal{S}}$) If $t \in \mathcal{ISN}(\mathtt{I})$ and $l \in \mathcal{ISN}(\mathtt{I})$, then $t;l \in \mathcal{ISN}(\mathtt{I})$.
($\mathtt{app}_{nil}$) If $\lambda x.t \in \mathcal{ISN}(\mathtt{I})$, then $(\lambda x.t)\epsilon \in \mathcal{ISN}(\mathtt{I})$
($\mathtt{app}_{\in}$) If $t\{x\backslash u\} \circ l \in \mathcal{ISN}(\mathtt{I})$ and $x \in \mathtt{fv}(t)$, then $(\lambda x.t)(u;l) \in \mathcal{ISN}(\mathtt{I})$.
($\mathtt{app}_{\notin}$) If $t \circ l \in \mathcal{ISN}(\mathtt{I})$ and $u \in \mathcal{ISN}(\mathtt{I})$ and $x \notin \mathtt{fv}(t)$, then $(\lambda x.t)(u;l) \in \mathcal{ISN}(\mathtt{I})$.

Every strongly I-normalizing object o turns out to be in $\mathcal{ISN}(\mathtt{I})$.

Theorem 10.1. If $o \in \mathcal{SN}(\mathtt{I})$, then $o \in \mathcal{ISN}(\mathtt{I})$.

Proof. By induction on $\langle \eta_{\mathtt{I}}(o), |o| \rangle$.

If $o = \epsilon$, then the statement is trivial. If $o = u;l$, then the *i.h.* gives u and l in $\mathcal{ISN}(\mathtt{I})$ so that $u;l \in \mathcal{ISN}(\mathtt{I})$ using rule ($\supset \mathtt{l}_{\mathcal{S}}$). The same reasoning can be applied if $o = \lambda x.t$ or $o = xl$. If $o = (\lambda x.t)l$, we consider two cases.

— $l = \epsilon$. The *i.h.* gives $\lambda x.t \in \mathcal{ISN}(\mathtt{I})$, then $(\lambda x.t)\epsilon \in \mathcal{ISN}(\mathtt{I})$ using rule ($\mathtt{app}_{nil}$).
— $l = u;l'$. We consider again two cases.
  – $x \in \mathtt{fv}(t)$. Since $\eta_{\mathtt{I}}(t\{x\backslash u\} \circ l) < \eta_{\mathtt{I}}(o)$, then by the *i.h.* we have $t\{x\backslash u\} \circ l \in \mathcal{ISN}(\mathtt{I})$, then we obtain $o \in \mathcal{ISN}(\mathtt{I})$ using rule ($\mathtt{app}_{\in}$).
  – $x \notin \mathtt{fv}(t)$. Since $\eta_{\mathtt{I}}(t\{x\backslash u\} \circ l) = \eta_{\mathtt{I}}(t \circ l) < \eta_{\mathtt{I}}(o)$, then by the *i.h.* we have $t \circ l \in \mathcal{ISN}(\mathtt{I})$. Moreover, $\eta_{\mathtt{I}}(u) < \eta_{\mathtt{I}}(o)$, so that also by the *i.h.* we have $u \in \mathcal{ISN}(\mathtt{I})$. Then, $o \in \mathcal{ISN}(\mathtt{I})$ using rule ($\mathtt{app}_{\notin}$). □

Moreover, every object in $\mathcal{ISN}(\mathtt{I})$ is $\mathcal{S}_{\mathtt{I}}$-typable.

Theorem 10.2. If $o \in \mathcal{ISN}(\mathtt{I})$, then there exists Φ' such that $\Phi' \rhd_{\mathcal{S}_{\mathtt{I}}} \Gamma \mid \Sigma \vdash o : \tau$.

Proof. By induction on the definition of $\mathcal{ISN}(\mathtt{I})$. The cases (ax), ($\supset$ r), (hlist) and ($\supset$ $\mathtt{l}_{\mathcal{S}}$) are straightforward. Let consider $(\lambda x.t)\epsilon \in \mathcal{ISN}(\mathtt{I})$ coming from $\lambda x.t \in \mathcal{ISN}(\mathtt{I})$ by rule ($\mathtt{app}_{nil}$). By the *i.h.* we have $\Phi_{\lambda x.t} \triangleright \Gamma \mid \_ \vdash \lambda x.t : \tau$, thus we conclude by

$$\Phi' := \frac{\Phi_{\lambda x.t} \triangleright \Gamma \mid \_ \vdash \lambda x.t : \tau \qquad \overline{\varnothing \mid \tau \vdash \epsilon : \tau}}{\Gamma \mid \_ \vdash (\lambda x.t)\epsilon : \tau}$$

Consider $(\lambda x.t)(u;l) \in \mathcal{ISN}(\mathtt{I})$ coming from $t\{x\backslash u\} \circ l \in \mathcal{ISN}(\mathtt{I})$ and $x \in \mathtt{fv}(t)$ by rule ($\mathtt{app}_{\in}$). By the *i.h.* $\Phi'_1 \triangleright \Gamma \mid \_ \vdash t\{x\backslash u\} \circ l : \tau$, thus $\Phi' \triangleright \Gamma \mid \_ \vdash (\lambda x.t)(u;l) : \tau$ by Lemma 9.4.

Consider $(\lambda x.t)(u;l) \in \mathcal{ISN}(\mathtt{I})$ coming from $t \circ l \in \mathcal{ISN}(\mathtt{I})$, $u \in \mathcal{ISN}(\mathtt{I})$ and $x \notin \mathtt{fv}(t)$ by rule ($\mathtt{app}_{\notin}$). By the *i.h.* $\Phi'_1 \triangleright \Gamma_1 \mid \_ \vdash t \circ l : \tau$ and $\Phi'_2 \triangleright \Gamma_2 \mid \_ \vdash u : \sigma$. By Lemma 9.2, we get $\Phi_t \triangleright \Gamma_t \mid \_ \vdash t : \tau'$ and $\Phi_l \triangleright \Gamma_l \mid \tau' \vdash l : \tau$, where $\Gamma_1 = \Gamma_t + \Gamma_l$. Thus, we get

$$\Phi' := \frac{\dfrac{\Phi_t \triangleright \Gamma_t \mid \_ \vdash t : \tau'}{\Gamma_t \mid \_ \vdash \lambda x.t : \{\!\{\,\}\!\} \supset \tau'} \qquad \dfrac{\Phi'_2 \triangleright \Gamma_2 \mid \_ \vdash u : \sigma \qquad \Phi_l \triangleright \Gamma_l \mid \tau' \vdash l : \tau}{\Gamma_2 + \Gamma_l \mid \{\!\{\,\}\!\} \supset \tau' \vdash u;l : \tau}}{\Gamma_t + \Gamma_l + \Gamma_2 \mid \_ \vdash (\lambda x.t)(u;l) : \tau}$$

□

We can thus conclude this section with the equivalence between strongly I-normalizing objects and typable objects in system $\mathcal{S}_{\mathtt{I}}$.

Corollary 10.2. $\Gamma \mid \Sigma \vdash_{\mathcal{S}_{\mathtt{I}}} o : \tau$ if and only if $o \in \mathcal{SN}(\mathtt{I})$.

Proof. Typability implies normalization by Corollary 10.1. Normalization implies typability by Theorems 10.1 and 10.2. □

11. Conclusion

This paper proposes a resource aware computational semantics for Herbelin's syntax. The resulting E-calculus can be seen as a refinement of the non-resource aware I-calculus, whose meta-level operations are implemented by more atomic reduction rules in E. In contrast to more complex resource-controlled interpretations realized by means of explicit control operators for weakening and contraction, e.g. Ghilezan et al. (2011), our implementation is achieved by rewriting rules inspired from the *substitution at a distance* paradigm, recently used in successful investigations in computer science.

We define typing systems for both calculi I and E, based on relevant, strict, non-idempotent intersection types. Typing rules of the systems are goal directed. In both cases, typability is used to completely characterize head linear and strongly normalizing terms. Our results are presented in a self-contained form, without resorting to their isomorphic natural deduction counterparts. The proofs only use combinatorial arguments, neither reducibility candidates nor memory operators have been necessary.

Balance between typing and reduction systems in a resource aware framework is sensitive; this is illustrated by the approach in Kesner and Ventura (2014a) and the one taken here. Indeed, adding a dereliction rule to the reduction system, as done in this paper, allows to restrict the witness type derivations to erasable subterms only, while

dereliction can be simply omitted, as done in Kesner and Ventura (2014a), when using a resource-consuming approach based on witnesses everywhere.

On further developments, the technique used in Bernadet and Lengrand (2013) to establish the length of the longest reduction sequence of strongly normalizing terms could be adapted to our framework. Extensions of this ideas and techniques to classical logic can also be considered.

Appendix

Lemma 11.1.

1. If $o' = t[x\backslash u]l_0l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then $o = (\lambda x.t)(u; l_0)l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.
2. If $o' = x(m@l_0)l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then $o = (xm)l_0l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.
3. If $o' = t(m@l_0)l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then $o = (tm)l_0l_1 \dots l_n \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.
4. If $o' = C[\![vl]\!][x\backslash v] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ and $|C[\![xl]\!]|_x > 1$, then $o = C[\![xl]\!][x\backslash v] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.
5. If $o' = C[\![vl]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ and $|C[\![xl]\!]|_x = 1$, then $o = C[\![xl]\!][x\backslash v] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$.

Proof. Let us treat Point 1. To prove $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, it is sufficient to prove that all the one-step reducts of o are in $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$. For that, we first remark that the hypothesis implies that t, u, l_i $(i = 0 \dots n) \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ so that $\eta_{\mathtt{E}\backslash\mathtt{w}}(t)$, $\eta_{\mathtt{E}\backslash\mathtt{w}}(u)$ and $\eta_{\mathtt{E}\backslash\mathtt{w}}(l_i)$ $(i = 0 \dots n)$ are all well-defined. We then reason by induction on $\eta_{\mathtt{E}\backslash\mathtt{w}}(t) + \eta_{\mathtt{E}\backslash\mathtt{w}}(u) + \sum_{i=0}^{n} \eta_{\mathtt{E}\backslash\mathtt{w}}(l_i)$.

— If $o \to o'$, then $o' \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ by hypothesis.
— If $o \to (\lambda x.t')(u; l_0)l_1 \dots l_n$, where $t \to t'$, then $\eta_{\mathtt{E}\backslash\mathtt{w}}(t') < \eta_{\mathtt{E}\backslash\mathtt{w}}(t)$ so that we conclude by the *i.h.*
— If $o \to (\lambda x.t)(u'; l_0)l_1 \dots l_n$, where $u \to u'$, then $\eta_{\mathtt{E}\backslash\mathtt{w}}(u') < \eta_{\mathtt{E}\backslash\mathtt{w}}(u)$ so that we conclude by the *i.h.*
— If $o \to (\lambda x.t)(u; l_0')l_1 \dots l_n$, where $l_0 \to l_0'$ or $o \to (\lambda x.t)(u; l_0)l_1 \dots l_i' \dots l_n$, where $l_i \to l_i'$, then $\eta_{\mathtt{E}\backslash\mathtt{w}}(l_i') < \eta_{\mathtt{E}\backslash\mathtt{w}}(l_i)$, so that we conclude by the *i.h.*
— There is no other possible reduct of o.

Points 2 and 3 use Lemma 7.2 to infer in particular $m, l_0 \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then proceeds similarly. Points 4 and 5 use Corollary 7.1 to infer in particular $C[\![xl]\!] \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then proceeds similarly. □

Lemma 7.5. $\mathcal{SN}(\mathtt{E}\backslash\mathtt{w}) = \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$.

Proof. If $o \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$, then we show $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by induction on $\langle \eta_{\mathtt{E}\backslash\mathtt{w}}(o), |o| \rangle$. We reason by cases. If $o = \epsilon$, $o = t; l$, $o = \lambda x.t$, $o = xl$ or $o = t[x\backslash u]$ with $|t|_x = 0$, then the property is straightforward. Otherwise,

— If $o = t[x\backslash v]$ with $|t|_x > 1$, i.e. $t = \mathtt{C}[\![xl]\!]$, then every o' such that $o \to o'$ verifies $o' \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ and in particular $o' = \mathtt{C}[\![vl]\!][x\backslash v]$. Since $\eta_{\mathtt{E}\backslash\mathtt{w}}(o') < \eta_{\mathtt{E}\backslash\mathtt{w}}(o)$, then $o' \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by the *i.h.* so that we can conclude $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by rule (C).
— If $o = t[x\backslash v]$ with $|t|_x = 1$, i.e. $t = \mathtt{C}[\![xl]\!]$, then every o' such that $o \to o'$ verifies $o' \in \mathcal{SN}(\mathtt{E}\backslash\mathtt{w})$ and in particular $o' = \mathtt{C}[\![vl]\!]$. Since $\eta_{\mathtt{E}\backslash\mathtt{w}}(o') < \eta_{\mathtt{E}\backslash\mathtt{w}}(o)$, then $o' \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by the *i.h.* so that we can conclude $o \in \mathcal{ISN}(\mathtt{E}\backslash\mathtt{w})$ by rule (D).
— If $o = vl$ is an application, then we reason by cases on v.

– If $o = u_0[x\backslash u_1]l_0 \dots l_n$ ($n \geqslant 0$) and $o \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$, then Lemma 7.1.1 gives $\eta_{\mathtt{E}\mathtt{w}}(u_0[x\backslash u_1]l_0 \dots l_n) = \eta_{\mathtt{E}\mathtt{w}}((u_0l_0 \dots l_n)[x\backslash u_1])$. Thus, in particular, $\eta_{\mathtt{E}\mathtt{w}}(u_0l_0 \dots l_n)$, $\eta_{\mathtt{E}\mathtt{w}}(u_1) \leqslant \eta_{\mathtt{E}\mathtt{w}}(o)$. Since $|u_0l_0 \dots l_n|, |u_1| < |o|$, $u_0l_0 \dots l_n, u_1 \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ holds by the *i.h.* We now consider three cases.
If $|u_0l_0 \dots l_n|_x = 0$, then $(u_0l_0 \dots l_n)[x\backslash u_1] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ by rule (W) and thus $o \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ by several applications of rule (E).
If $|u_0l_0 \dots l_n|_x > 1$, then $(u_0l_0 \dots l_n)[x\backslash u_1] = C[\![xl]\!][x\backslash u_1] \to_{\mathtt{c}} C[\![u_1l]\!][x\backslash u_1]$ and thus we have that $\eta_{\mathtt{E}\mathtt{w}}(C[\![u_1l]\!][x\backslash u_1]) < \eta_{\mathtt{E}\mathtt{w}}(C[\![xl]\!][x\backslash u_1])$. The *i.h.* gives $C[\![u_1l]\!][x\backslash u_1] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ so that $(u_0l_0 \dots l_n)[x\backslash u_1] = C[\![xl]\!][x\backslash u_1] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ holds by (C) and $o \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ holds by several applications of rule (E).
If $|u_0l_0 \dots l_n|_x = 1$, then $(u_0l_0 \dots l_n)[x\backslash u_1] = C[\![xl]\!][x\backslash u_1] \to_{\mathtt{d}} C[\![u_1l]\!]$ and thus we have that $\eta_{\mathtt{E}\mathtt{w}}(C[\![u_1l]\!]) < \eta_{\mathtt{E}\mathtt{w}}(C[\![xl]\!][x\backslash u_1])$. The *i.h.* gives $C[\![u_1l]\!] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ so that $(u_0l_0 \dots l_n)[x\backslash u_1] = C[\![xl]\!][x\backslash u_1] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ holds by (D) and $o \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ holds by several applications of rule (E).
In all the three cases, we get $o \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$.

– If $o = xl_0l_1 \dots l_n$, with $n \geqslant 1$, then $o \to_{@_{\mathtt{var}}} x(l_0@l_1) \dots l_n = o'$. Moreover, $\eta_{\mathtt{E}\mathtt{w}}(o') < \eta_{\mathtt{E}\mathtt{w}}(o)$ so that the *i.h.* gives $o' \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$. We conclude by rule $(@_{\mathtt{var}})$.

– If $o = tl_0 \dots l_n$, with $n \geqslant 0$, we reason similarly to the previous case but we conclude with rule $(@_{\mathtt{app}})$.

– If $o = (\lambda x.t)l_0 \dots l_n$, with $n \geqslant 0$, then we reason by cases on l_0. In every case, we can conclude as before but using rules $(\mathtt{dB}_{\mathtt{nil}})$ and $(\mathtt{dB}_{\mathtt{cons}})$.

For the converse, we reason by induction on the definition of $o \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$. If $o = \epsilon$, $o = t;l$, $o = \lambda x.t$, $o = xl$ or $o = t[x\backslash v]$ with $|t|_x = 0$, then the property is straightforward by using the *i.h.* The remaining cases are as follows:

— If $o = (\lambda x.t)(u;l_0)l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $o' = t[x\backslash u]l_0l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then the *i.h.* gives $o' \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ so that in particular $u \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$. We conclude by Lemma 11.1:1.
— If $o = (\lambda x.t)\epsilon l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $o' = (\lambda x.t)l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then the *i.h.* gives $o' \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ and we proceed similarly to the previous case.
— If $o = (xm)l_0l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $o' = x(m@l_0)l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then $o' \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ by the *i.h.* so that we conclude by Lemma 11.1:2.
— If $o = (tm)l_0l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $o' = t(m@l_0)l_1 \dots l_n \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then $o' \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ by the *i.h.* so that we conclude by Lemma 11.1:3.
— If $o = C[\![xl]\!][x\backslash v] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $|C[\![xl]\!]|_x > 1$ and $o' = C[\![vl]\!][x\backslash v] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then $o' \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ by the *i.h.* so that we conclude by Lemma 11.1:4.
— If $o = C[\![xl]\!][x\backslash v] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $|C[\![xl]\!]|_x = 1$ and $o' = C[\![vl]\!] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then $o' \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ by the *i.h.* so that we conclude by Lemma 11.1:5.
— If $v[x\backslash u]l \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$ where $(vl)[x\backslash u] \in \mathcal{ISN}(\mathtt{E}\setminus\mathtt{w})$, then $(vl)[x\backslash u] \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ by the *i.h.* so that $v[x\backslash u]l \in \mathcal{SN}(\mathtt{E}\setminus\mathtt{w})$ since $\eta_{\mathtt{E}\mathtt{w}}(v[x\backslash u]l) =_{\text{Lemma 7.1.1}} \eta_{\mathtt{E}\mathtt{w}}((vl)[x\backslash u])$.

□

References

Accattoli, B. (2012). An abstract factorization theorem for explicit substitutions. In: *LIPIcs*, vol. 15, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 6–21.

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

Accattoli, B. and Kesner, D. (2010). The structural lambda-calculus. In: Dawar, A. and Veith, H. (eds.) *Proceedings of 24th EACSL Conference on Computer Science Logic (CSL)*, Lecture Notes in Computer Science, vol. 6247, Springer-Verlag, 381–395.

Andreoli, J. (1992). Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation* **2** (3) 297–347.

Baader, F. and Nipkow, T. (1998). *Term Rewriting and All That*, Cambridge University Press.

Barendregt, H. (1984). *The Lambda Calculus: Its Syntax and Semantics (revised edition)*, Studies in Logic and the Foundations of Mathematics, vol. 103, Elsevier Science, Amsterdan, The Netherlands.

Barendregt, H., Coppo, M. and Dezani-Ciancaglini, M. (1983). A filter lambda model and the completeness of type assignment. *Bulletin of Symbolic Logic* **48** (4) 931–940.

Bernadet, A. and Lengrand, S. (2011). Complexity of strongly normalising λ-terms via non-idempotent intersection types. In: Hofmann, M. (ed.) *Proceedings of the 14th International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, Lecture Notes in Computer Science, vol. 6604, Springer-Verlag.

Bernadet, A. and Lengrand, S. (2013). Non-idempotent intersection types and strong normalisation. *Logical Methods in Computer Science* **9** (4). doi: 10.2168/LMCS-9(4:3)2013

Boudol, G., Curien, P.-L. and Lavatelli, C. (1999). A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science* **9** (4) 437–482.

Bucciarelli, A., Kesner, D. and Ronchi Della Rocca, S. (2014). The inhabitation problem for non-idempotent intersection types. In: *IFIP Theoretical Computer Science (TCS)*, LNCS, Springer-Verlag, 341–354.

Coppo, M. and Dezani-Ciancaglini, M. (1978). A new type-assignment for lambda terms. *Archiv für Mathematische Logik und Grundlagenforschung* **19** (1) 139–156.

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

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

Damiani, F. and Giannini, P. (1994). A decidable intersection type system based on relevance. In: *International Symposium on Theoretical Computer Science (TACS)*, Lecture Notes in Computer Science, vol. 789, Springer-Verlag, 707–725.

Danos, V. and Regnier, L. (2003). Head Linear Reduction. Available at `http://iml.univ-mrs.fr/~regnier/articles/pam.ps.gz`.

De Benedetti, E. and Ronchi Della Rocca, S. (2013). Bounding normalization time through intersection types. In: Paolini, L. (ed.) *Proceedings of 6th Workshop on Intersection Types and Related Systems (ITRS)*, EPTCS, Cornell University Library, 48–57.

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

Díaz, J., Lanese, I. and Sangiorgi, D. (eds.) (2014). *IFIP Theoretical Computer Science (TCS)*, LNCS, Springer-Verlag.

Dyckhoff, R. and Urban, C. (2003). Strong normalization of herbelin's explicit substitution calculus with substitution propagation. *Journal of Logic and Computation* **13** (5) 689–706.

Espírito Santo, J. (2000). Revisiting the correspondence between cut elimination and normalisation. In: Montanari, U., Rolim, J. D. P. and Welzl, E. (eds.) *Proceedings of the Automata, Languages and Programming, 27th International Colloquium, ICALP 2000*, Lecture Notes in Computer Science, vol. 1853, Springer-Verlag, 600–611.

Espírito Santo, J. (2009). The lambda-calculus and the unity of structural proof theory. *Theory of Computing Systems* **45** (4) 963–994.

Espírito Santo, J., Ivetic, J. and Likavec, S. (2012). Characterising strongly normalising intuitionistic terms. *Fundamenta Informaticae* **121** (1–4) 83–120.

Gardner, P. (1994). Discovering needed reductions using type theory. In: Hagiya, M. and Mitchell, J. C. (eds.) *Proceedings of the Theoretical Aspects of Computer Software, International Conference TACS '94,* Lecture Notes in Computer Science, vol. 789, Springer, 555–574.

Gentzen, G. (1969). The collected papers of Gerhard Gentzen. In: Szabo, M. E. (ed.) *Studies in Logic and the Foundations of Mathematics*, North-Holland Pub. Co.

Ghilezan, S., Ivetic, J., Lescanne, P. and Likavec, S. (2011). Intersection types for the resource control lambda calculi. In: Antonio Cerone, P. P. (ed.) *Proceedings of the 8th International Colloquium on Theoretical Aspects of Computing (ICTAC)*, Lecture Notes in Computer Science, vol. 6916, Springer-Verlag, 116–134.

Girard, J.-Y. (1987). Linear logic. *Theoretical Computer Science* **50** 1–102.

Girard, J.-Y. (1996). Proof-nets: The parallel syntax for proof-theory. In: Aglianò, P. and Ursini, A. (eds.) *Logic and Algebra*, Lecture Notes in Pure and Applied Mathematics, vol. 180, CRC Press, 97–124.

Herbelin, H. (1995). A lambda-calculus structure isomorphic to Gentzen-style sequent calculus structure. In: Pacholski, L. and Tiuryn, J. (eds.) *The 8th International Workshop on Computer Science Logic (CSL)*, Lecture Notes in Computer Science, vol. 933, Springer-Verlag, 61–75.

Kesner, D. and Ventura, D. (2014a). Quantitative types for intuitionistic calculi. Technical Report hal-00980868, Paris Cité Sorbonne.

Kesner, D. and Ventura, D. (2014b). Quantitative types for the linear substitution calculus. In: *IFIP Theoretical Computer Science (TCS)*, LNCS, Springer-Verlag, 296–310.

Kesner, D. and Ventura, D. (2015). A resource aware computational interpretation for herbelin's syntax. In: Leucker, M., Rueda, C. and Valencia, F. D. (eds.) *12th International Colloquium Theoretical Aspects of Computing – ICTAC 2015*, Lecture Notes in Computer Science, vol. 9399, Springer-Verlag, 388–403.

Kfoury, A. (1996). A linearization of the lambda-calculus and consequences. Technical report, Boston Universsity.

Kfoury, A. and Wells, J. B. (2004). Principality and type inference for intersection types using expansion variables. *Theoretical Computer Science* **311** (1–3) 1–70.

Kikuchi, K. (2014). Uniform proofs of normalisation and approximation for intersection types. In *Proceedings of the 7th Workshop on Intersection Types and Related Systems (ITRS)*, Vienna, Austria.

Krivine, J.-L. (1993). *Lambda-Calculus, Types and Models*, Ellis Horwood.

Lengrand, S., Lescanne, P., Dougherty, D., Dezani-Ciancaglini, M. and van Bakel, S. (2004). Intersection types for explicit substitutions. *Information and Computation* **189** (1) 17–42.

Miller, D., Nadathur, G., Pfenning, F. and Scedrov, A. (1991). Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic* **51** (1–2) 125–157.

Milner, R. (2007). Local bigraphs and confluence: Two conjectures: (extended abstract). *Electronic Notes in Theoretical Computer Science* **175** (3) 65–73.

Neergaard, P. M. and Mairson, H. G. (2004). Types, potency, and idempotency: why nonlinearity and amnesia make a type system work. In: Okasaki, C. and Fisher, K. (eds.) *Proceedings of the 9th ACM SIGPLAN International Conference on Functional Programming (ICFP)*, ACM, 138–149.

Ong, L. and Ramsay, S. J. (2011). Verifying higher-order functional programs with pattern matching algebraic data types. In: Ball, T. and Sagiv, M. (eds.) *Proceedings of the 38th Annual ACM Symposium on Principles of Programming Languages (POPL)*, ACM, 587–598.

Pagani, M. and Ronchi Della Rocca, S. (2011). Solvability in resource lambda-calculus. In: Ong, L. (ed.) *Foundations of Software Science and Computation Structures (FOSSACS)*, Lecture Notes in Computer Science, vol. 6014, Springer-Verlag, 358–373.

Pottinger, G. (1977). Normalization as a homomorphic image of cut-elimination. *Annals of Mathematical Logic* **12** 323–357.

Pottinger, G. (1980). A type assignment for the strongly normalizable λ-terms. In: Seldin, J. P. and Hindley, J. R. (eds.) *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Academic Press, 561–578.

Prawitz, D. (1965). *Natural Deduction: A Proof-Theoretical Study*. Phd thesis, Stockholm University.

Regnier, L. (1994). Une équivalence sur les lambda-termes. *Theoretical Computer Science* **2** (126) 281–292.

Urzyczyn, P. (1999). The emptiness problem for intersection types. *Journal of Symbolic Logic* **64** (3) 1195–1215.

Valentini, S. (2001). An elementary proof of strong normalization for intersection types. *Archive of Mathematical Logic* **40** (7) 475–488.

van Bakel, S. (1992). Complete restrictions of the intersection type discipline. *Theoretical Computer Science* **102** (1)135–163.

Zucker, J. (1974). The correspondence between cut-elimination and normalization I. *Annals of Mathematical Logic* **7** (1) 1–112.