# A journey through resource control lambda calculi and explicit substitution using intersection types (an account)

Silvia Ghilezan, Jelena Ivetic, Pierre Lescanne, Silvia Likavec

# A journey through resource control lambda calculi and explicit substitution using intersection types

S. Ghilezan [*1], J. Ivetić [†1], P. Lescanne [‡2], and S. Likavec [§3]

[1]University of Novi Sad, Faculty of Technical Sciences, Serbia
[2]University of Lyon, École Normal Supérieure de Lyon, France
[3]Dipartimento di Informatica, Università di Torino, Italy

**Abstract**

In this paper we invite the reader to a journey through three lambda calculi with resource control: the lambda calculus, the sequent lambda calculus, and the lambda calculus with explicit substitution. All three calculi enable explicit control of resources due to the presence of weakening and contraction operators. Along this journey, we propose intersection type assignment systems for all three resource control calculi. We recognise the need for three kinds of variables all requiring different kinds of intersection types. Our main contribution is the characterisation of strong normalisation of reductions in all three calculi, using the techniques of reducibility, head subject expansion, a combination of well-orders and suitable embeddings of terms.

**Keywords:** lambda calculus; resource control; sequent calculus; explicit substitution; intersection types; strong normalisation; typeability

## Introduction

It is well known that simply typed $\lambda$-calculus captures the computational content of intuitionistic natural deduction through Curry-Howard correspondence [34]. This connection between logic and computation can be extended to other calculi and logical systems [27]: Herbelin's $\overline{\lambda}$-calculus [33], Pinto and Dyckhoff's $\lambda\pi\sigma$-calculus [49] and Espírito Santo's $\lambda^{\mathsf{Gtz}}$-calculus [20] correspond to intuitionistic sequent calculus. In the

[*]Email: gsilvia@uns.ac.rs
[†]Email: jelenaivetic@uns.ac.rs
[‡]Email: pierre.lescanne@ens-lyon.fr
[§]Email: likavec@di.unito.it

realm of classical logic, Parigot's $\lambda\mu$-calculus [48] corresponds to classical natural deduction, whereas Barbanera and Berardi's symmetric calculus [3] and Curien and Herbelin's $\overline{\lambda}\mu\widetilde{\mu}$-calculus [14] correspond to its sequent version. Extending first, the $\lambda x$ calculus of explicit substitution and later $\lambda$-calculus and $\lambda^{Gtz}$-calculus with explicit operators for erasure (a.k.a. weakening) and duplication (a.k.a. contraction) brings the same correspondence to intuitionistic natural deduction and intuitionistic sequent calculus with explicit structural rules of weakening and contraction on the logical side [24], as investigated in [35, 36, 26].

On the other hand, let us consider type assignment systems for various calculi. To overcome the limitations of the simple type discipline in which the only forming operator is an arrow $\rightarrow$, a new type forming operator $\cap$ was introduced in [12, 13, 50, 55]. The newly obtained intersection type assignment systems enabled complete characterisation of termination of term calculi [60, 23, 25]. The extension of Curry-Howard correspondence to other formalisms brought the need for intersection types into many different settings [18, 39, 43, 46].

Our work is inspired by and extends Kesner and Lengrand's [35] work on resource operators for $\lambda$-calculus with explicit substitution. Their linear $\lambda$lxr-calculus introduces operators for linear substitution, erasure and duplication, preserving at the same time confluence and full composition of explicit substitutions of its predecessor $\lambda x$ [8, 54]. The simply typed version of this calculus corresponds to the intuitionistic fragment of Linear Logic's proof-nets, according to Curry-Howard correspondence, and it enjoys strong normalisation and subject reduction. Resource control in sequent $\lambda$-calculus was proposed by Ghilezan et al. in [26], whereas resource control both in $\lambda$-calculus and $\lambda$x-calculus was further developed in [36, 37].

In order to control all resources, in the spirit of $\lambda I$-calculus (see e.g. [4]), void lambda abstraction is not acceptable, so in order to have $\lambda x.M$ the variable $x$ has to occur in $M$. But if $x$ is not used in a term $M$, one can perform an *erasure (a.k.a weakening)* by using the expression $x \odot M$. In this way, the term $M$ does not contain the variable $x$, but the term $x \odot M$ does. Similarly, a variable should not occur twice. If nevertheless, we want to have two positions for the same variable, we have to duplicate it explicitly, using fresh names. This is done by using the operator $x <^{x_1}_{x_2} M$, called *duplication (a.k.a contraction)* which creates two fresh variables $x_1$ and $x_2$.

Explicit control of erasure and duplication leads to decomposing of reduction steps into more atomic steps, thus revealing the details of computation which are usually left implicit. Since erasing and duplicating of (sub)terms essentially changes the structure of a program, it is important to see how this mechanism really works and to be able to control this part of computation. We chose a direct approach to term calculi rather than taking a more common path through linear logic [1, 7]. In practice, for instance in the description of compilers by rules with binders [52, 53], the implementation of substitutions of linear variables by inlining[1] is simple and efficient when substitution of duplicated variables requires the cumbersome and time consuming mechanism of pointers and it is therefore important to tightly control duplication. On the other hand, precise control of erasing does not require a garbage collector and prevents memory

[1] *Inlining* is the technics which consists in copying at compile time the text of a function instead of implementing a call to that function.

leaking.

Our main goal is to characterize the termination of reductions for term calculi with explicit control of duplication and erasure, in different frameworks: natural deduction, sequent style and with explicit substitution. We revisit the syntax and reduction rules of three term calculi with explicit operators for weakening and contraction: $\lambda_{\textcircled{R}}$ (the extension of the $\lambda$-calculus), $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ (the extension of the sequent lambda calculus $\lambda^{\mathsf{Gtz}}$) and $\lambda_{\textcircled{R}}^{x}$ (the extension of the $\lambda x$-calculus with explicit substitution). We then introduce intersection types into all three calculi $\lambda_{\textcircled{R}}$, $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ and $\lambda_{\textcircled{R}}^{x}$. Our intersection type assignment systems $\lambda_{\textcircled{R}}\cap$, $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}\cap$ and $\lambda_{\textcircled{R}}^{x}\cap$ integrate intersection into logical rules, thus preserving syntax-directedness of the system. We assign restricted form of intersection types to terms, namely strict types, therefore minimizing the need for pre-order on types. By using these intersection type assignment systems we prove that terms in all three calculi enjoy strong normalisation if and only if they are typeable. To the best of our knowledge, together with the conference version of this paper [28], this is the first treatment of intersection types in the presence of resource control operators. Intersection types fit naturally to resource control. Indeed, the control allows us to consider three types of variables: variables as placeholders (the traditional view of $\lambda$-calculus), variables to be duplicated and variables to be erased because they are irrelevant. For each kind of a variable, there is a kind of type associated to it, namely a strict type for a *placeholder*, an intersection for a variable *to-be-duplicated*, and a specific type for an *erased* variable.

We first prove in Section 1 that terms typeable in $\lambda_{\textcircled{R}}$-calculus are strongly normalising by adapting the reducibility method for explicit resource control operators. Then we prove that all strongly normalising terms are typeable in $\lambda_{\textcircled{R}}$-calculus by using typeability of normal forms and head subject expansion.

Further, we prove strong normalisation for $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ and $\lambda_{\textcircled{R}}^{x}$, in Section 2 and Section 3, respectively, by using a combination of well-orders and a suitable embeddings of $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$-terms and $\lambda_{\textcircled{R}}^{x}$-terms into $\lambda_{\textcircled{R}}$-terms which preserve typeability and enable the simulation of all reductions and equations by the operational semantics of the $\lambda_{\textcircled{R}}$-calculus. Finally, we prove that strong normalisation implies typeability in $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ and $\lambda_{\textcircled{R}}^{x}$ using head subject expansion.

**Related work** The idea to control the use of variables can be traced back to Church's $\lambda I$-calculus [4]. Currently there are several different lines of research in resource aware term calculi. Van Oostrom [61] and later Kesner and Lengrand [35], applying ideas from linear logic [31], proposed to extend the $\lambda$-calculus and the $\lambda x$-calculus, with operators to control the use of variables (resources). Generalising this approach, Kesner and Renaud [36, 37] developed the *prismoid of resources*, a system of eight calculi parametric over the explicit and implicit treatment of substitution, erasure and duplication. Resource control in sequent calculus corresponding to classical logic was proposed in [62]. On the other hand, process calculi and their relation to $\lambda$-calculus by Boudol [9] initialised investigations in resource aware non-deterministic $\lambda$-calculus with multiplicities and a generalised notion of application [10]. The theory was connected to linear logic via differential $\lambda$-calculus in [19] and typed with non-idempotent intersection types in [47]. In this paper we follow the notation of [62] and [28], which

is related to [61].

This paper is an extended and revised version of [28]. In addition to $\lambda_{\circledR}$-calculus and $\lambda_{\circledR}^{\mathsf{Gtz}}$-calculus presented in [28], this extended version adds the treatment of the $\lambda_{\circledR}^{\mathsf{x}}$-calculus, the resource lambda calculus with explicit substitution, together with the characterization of strong normalisation for this calculus. Also, the proof that typeability implies strong normalisation in $\lambda_{\circledR}$-calculus is improved.

**Outline of the paper** In Section 1 we first give the syntax and reduction rules for $\lambda_{\circledR}$-calculus, followed by the intersection type assignment system and the characterisation of strong normalisation. Section 2 deals with $\lambda_{\circledR}^{\mathsf{Gtz}}$-calculus, its syntax, reduction rules, intersection type assignment system and the characterisation of strong normalisation. Section 3 introduces $\lambda_{\circledR}^{\mathsf{x}}$-calculus with its syntax, reduction rules and intersection type assignment system, again followed by the characterisation of strong normalisation. Finally, we conclude in Section 4 with some directions for future work.

# Contents



# 1 Intersection types for the resource control lambda calculus $\lambda_{\circledR}$

In this section we focus on the resource control lambda calculus $\lambda_{\circledR}$. First we revisit its syntax and operational semantics; further we introduce intersection type assignment system and finally we prove that typebility in the proposed system completely characterises the set of strongly normalising $\lambda_{\circledR}$-terms.

## 1.1 Resource control lambda calculus $\lambda_{\mathbb{R}}$

The *resource control* lambda calculus, $\lambda_{\mathbb{R}}$, is an extension of the $\lambda$-calculus with explicit operators for weakening and contraction. It corresponds to the $\lambda_{\mathsf{cw}}$-calculus of Kesner and Renaud, proposed in [36] as a vertex of "the prismoid of resources", where substitution is implicit. We use a notation along the lines of [62] and close to [61]. It is slightly modified w.r.t. [36] in order to emphasize the correspondence between this calculus and its sequent counterpart.

First of all, we introduce the syntactic category of *pre-terms* of $\lambda_{\mathbb{R}}$-calculus given by the following abstract syntax:

$$\text{Pre-terms} \qquad f \quad ::= \quad x \,|\, \lambda x.f \,|\, ff \,|\, x \odot f \,|\, x <^{x_1}_{x_2} f$$

where $x$ ranges over a denumerable set of term variables. $\lambda x.f$ is an *abstraction*, $ff$ is an *application*, $x \odot f$ is a *weakening* and $x <^{x_1}_{x_2} f$ is a *contraction*. The contraction operator is assumed to be insensitive to the order of the arguments $x_1$ and $x_2$, i.e. $x <^{x_1}_{x_2} f = x <^{x_2}_{x_1} f$.

The set of free variables of a pre-term $f$, denoted by $Fv(f)$, is defined as follows:

$$Fv(x) = x; \quad Fv(\lambda x.f) = Fv(f) \setminus \{x\}; \quad Fv(fg) = Fv(f) \cup Fv(g);$$
$$Fv(x \odot f) = \{x\} \cup Fv(f); \quad Fv(x <^{x_1}_{x_2} f) = \{x\} \cup Fv(f) \setminus \{x_1, x_2\}.$$

In $x <^{x_1}_{x_2} f$, the contraction binds the variables $x_1$ and $x_2$ in $f$ and introduces a free variable $x$. The operator $x \odot f$ also introduces a free variable $x$. In order to avoid parentheses, we let the scope of all binders extend to the right as much as possible.

The set of $\lambda_{\mathbb{R}}$-*terms*, denoted by $\Lambda_{\mathbb{R}}$ and ranged over by $M, N, P, M_1, \ldots$. is a subset of the set of pre-terms, defined in Figure 1.

$$\frac{}{x \in \Lambda_{\mathbb{R}}} \qquad \frac{f \in \Lambda_{\mathbb{R}} \;\; x \in Fv(f)}{\lambda x.f \in \Lambda_{\mathbb{R}}}$$

$$\frac{f \in \Lambda_{\mathbb{R}} \;\; g \in \Lambda_{\mathbb{R}} \;\; Fv(f) \cap Fv(g) = \emptyset}{fg \in \Lambda_{\mathbb{R}}}$$

$$\frac{f \in \Lambda_{\mathbb{R}} \;\; x \notin Fv(f)}{x \odot f \in \Lambda_{\mathbb{R}}} \qquad \frac{f \in \Lambda_{\mathbb{R}} \;\; x_1 \neq x_2 \;\; x_1, x_2 \in Fv(f) \;\; x \notin Fv(f) \setminus \{x_1, x_2\}}{x <^{x_1}_{x_2} f \in \Lambda_{\mathbb{R}}}$$

Figure 1: $\Lambda_{\mathbb{R}}$: $\lambda_{\mathbb{R}}$-terms

Informally, we say that a term is a pre-term in which in every subterm every free variable occurs exactly once, and every binder binds (exactly one occurrence of) a free variable. Our notion of terms corresponds to the notion of linear terms in [35]. In that sense, only linear expressions are in the focus of our investigation. In other words, terms are well-formed in $\lambda_{\mathbb{R}}$ if and only if bound variables appear actually in the term and variables occur at most once. These conditions will be assumed throughout the paper without mentioning them explicitly. This assumption is not a restriction, since every traditional term has a corresponding $\lambda_{\mathbb{R}}$-term, as illustrated by the following example.

**Example 1.** *Pre-terms* $\lambda x.y$ *and* $\lambda x.xx$ *are not* $\lambda_{\textcircled{R}}$*-terms, on the other hand pre-terms* $\lambda x.(x \odot y)$ *and* $\lambda x.x <_{x_2}^{x_1} (x_1 x_2)$ *are their corresponding* $\lambda_{\textcircled{R}}$*-terms.*

In the sequel, we use the notation $X \odot M$ for $x_1 \odot ... \, x_n \odot M$ and $X <_Z^Y M$ for $x_1 <_{z_1}^{y_1} ... \, x_n <_{z_n}^{y_n} M$, where $X$, $Y$ and $Z$ are lists of size $n$, consisting of all distinct variables $x_1,...,x_n,y_1,...,y_n,z_1,...,z_n$. If $n = 0$, i.e., if $X$ is the empty list, then $X \odot M = X <_Z^Y M = M$. Note that due to the equivalence relation defined in Figure 4, we can use these notations also for sets of variables of the same size.

In what follows we use Barendregt's convention [4] for variables: in the same context a variable cannot be both free and bound. This applies to binders like $\lambda x.M$ which binds $x$ in $M$, $x <_{x_2}^{x_1} M$ which binds $x_1$ and $x_2$ in $M$, and also to the implicit substitution $M[N/x]$ which can be seen as a binder for $x$ in $M$.

The set $\textcircled{R}$ of reduction rules $\rightarrow_{\lambda_{\textcircled{R}}}$ of the $\lambda_{\textcircled{R}}$-calculus is presented in Figure 2.

| | | | |
|---|---|---|---|
| $(\beta)$ | $(\lambda x.M)N$ | $\rightarrow$ | $M[N/x]$ |
| $(\gamma_1)$ | $x <_{x_2}^{x_1} (\lambda y.M)$ | $\rightarrow$ | $\lambda y.x <_{x_2}^{x_1} M$ |
| $(\gamma_2)$ | $x <_{x_2}^{x_1} (MN)$ | $\rightarrow$ | $(x <_{x_2}^{x_1} M)N$, if $x_1,x_2 \notin Fv(N)$ |
| $(\gamma_3)$ | $x <_{x_2}^{x_1} (MN)$ | $\rightarrow$ | $M(x <_{x_2}^{x_1} N)$, if $x_1,x_2 \notin Fv(M)$ |
| $(\omega_1)$ | $\lambda x.(y \odot M)$ | $\rightarrow$ | $y \odot (\lambda x.M)$, $x \neq y$ |
| $(\omega_2)$ | $(x \odot M)N$ | $\rightarrow$ | $x \odot (MN)$ |
| $(\omega_3)$ | $M(x \odot N)$ | $\rightarrow$ | $x \odot (MN)$ |
| $(\gamma\omega_1)$ | $x <_{x_2}^{x_1} (y \odot M)$ | $\rightarrow$ | $y \odot (x <_{x_2}^{x_1} M)$, $y \neq x_1,x_2$ |
| $(\gamma\omega_2)$ | $x <_{x_2}^{x_1} (x_1 \odot M)$ | $\rightarrow$ | $M[x/x_2]$ |

Figure 2: The set $\textcircled{R}$ of reduction rules of the $\lambda_{\textcircled{R}}$-calculus

The reduction rules are divided into four groups. The main computational step is $\beta$ reduction. The group of $(\gamma)$ reductions perform propagation of contraction into the expression. Similarly, $(\omega)$ reductions extract weakening out of expressions. This discipline allows us to optimize the computation by delaying duplication of terms on the one hand, and by performing erasure of terms as soon as possible on the other. Finally, the rules in $(\gamma\omega)$ group explain the interaction between explicit resource operators that are of different nature.

The inductive definition of the meta operator $[\ /\ ]$, representing the implicit substitution of free variables, is given in Figure 3. In order to obtain well formed terms as the results of substitution, $Fv(M) \cap Fv(N) = \emptyset$ must hold in this definition. Moreover, notice that for the expression $M[N/x]$ to make sense, $M$ must contain exactly one occurrence of the free variable $x$ and $M$ and $N$ must share no variable but $x$.[2] Indeed a substitution is always created by a $\beta$-reduction and, in the term $(\lambda x.M)N$, $x$ has to appear exactly once in $M$ and the other variables of $Fv(M) \cup Fv(N)$ as well. Barendregt convention on variable says that $x$ should not occur freely in $N$. Also, if the terms $N_1$ and $N_2$ are obtained from the term $N$ by renaming all the free variables in $N$ by fresh variables, then $M[N_1/x_1,N_2/x_2]$ denotes a parallel substitution.

[2]We prefer $x$ not to belong to $M$ in order to respect Barendregt convention on variable.

$$
\begin{array}{rcl}
x[N/x] & \triangleq & N \\
(\lambda y.M)[N/x] & \triangleq & \lambda y.M[N/x],\ x \neq y \\
(MP)[N/x] & \triangleq & M[N/x]P,\ x \notin Fv(P) \\
(MP)[N/x] & \triangleq & MP[N/x],\ x \notin Fv(M) \\
(y \odot M)[N/x] & \triangleq & y \odot M[N/x],\ x \neq y \\
(x \odot M)[N/x] & \triangleq & Fv(N) \odot M \\
(y <^{y_1}_{y_2} M)[N/x] & \triangleq & y <^{y_1}_{y_2} M[N/x],\ x \neq y \\
(x <^{x_1}_{x_2} M)[N/x] & \triangleq & Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M[N_1/x_1, N_2/x_2]
\end{array}
$$

Figure 3: Substitution in $\lambda_{\circledR}$-calculus

$$
\begin{array}{lrcl}
(\varepsilon_1) & x \odot (y \odot M) & \equiv_{\lambda_{\circledR}} & y \odot (x \odot M) \\
(\varepsilon_2) & x <^{x_1}_{x_2} M & \equiv_{\lambda_{\circledR}} & x <^{x_2}_{x_1} M \\
(\varepsilon_3) & x <^{y}_{z} (y <^{u}_{v} M) & \equiv_{\lambda_{\circledR}} & x <^{y}_{u} (y <^{z}_{v} M) \\
(\varepsilon_4) & x <^{x_1}_{x_2} (y <^{y_1}_{y_2} M) & \equiv_{\lambda_{\circledR}} & y <^{y_1}_{y_2} (x <^{x_1}_{x_2} M),\ x \neq y_1, y_2,\ y \neq x_1, x_2
\end{array}
$$

Figure 4: Equivalences in $\lambda_{\circledR}$-calculus

**Definition 2** (Parallel substitution)**.** $M[N/x,P/z] = (M[N/x])[P/z]$ for $x,z \in Fv(M)$ and $(Fv(M) \setminus \{x\}) \cap Fv(N) = (Fv(M) \setminus \{z\}) \cap Fv(P) = Fv(N) \cap Fv(P) = \emptyset$.

In the $\lambda_{\circledR}$-calculus, one works modulo equivalencies given in Figure 4.

Notice that because we work with $\lambda_{\circledR}$ terms, no variable is lost during the computation, which is stated by the following proposition.

**Proposition 3.** *If $M \to M'$ then $Fv(M) = Fv(M')$.*

*Proof.* The proof is by case analysis on the reduction rules. □

The following lemma explains how to compose implicit substitutions.

**Lemma 4.**

- *If $z \in FV(N)$ then $(M[N/x])[P/z] = M[N[P/z]/x]$.*
- *If $z \in FV(M)$ then $(M[N/x])[P/z] = (M[P/z])[N/x]$*

*Proof.*

Notice that for the expressions to make sense, one must have $x \in Fv(M)$ and $(Fv(M) \setminus \{x\}) \cap Fv(N) = \emptyset$, $(Fv(N) \setminus \{z\}) \cap Fv(P) = \emptyset$ and $(Fv(M) \setminus \{x\}) \cap Fv(P) = \emptyset$.

- $(x[N/x])[P/z] \triangleq N[P/z]$ and $x[N[P/z]/x] \triangleq N[P/z]$
- $((\lambda y.M)[N/x])[P/z] \triangleq (\lambda y.M[N/x])[P/z] \triangleq \lambda y.(M[N/x])[P/z] =_{IH} \lambda y.M[N[P/z]/x] \triangleq (\lambda y.M)[N[P/z]/x],\ x,z \neq y$

- $x \notin Fv(Q)$ (the case $x \notin Fv(M)$ is analogous)
  $((MQ)[N/x])[P/z] \triangleq (M[N/x]Q)[P/z] =_{z \in Fv(N)} (M[N/x])[P/z]Q =_{IH} M[N[P/z]/x]Q =$
  $(MQ)[N[P/z]/x]$

- $((y \odot M)[N/x])[P/z] \triangleq (y \odot M[N/x])[P/z] \triangleq y \odot M[N/x][P/z] =_{IH} y \odot M[N[P/z]/x] \triangleq$
  $(y \odot M)[N[P/z]/x],\ \ y \neq x, z$

- $((x \odot M)[N/x])[P/z] \triangleq (Fv(N) \odot M)[P/z] =_{z \in Fv(N)}$
  $(z \odot \{Fv(N) \setminus \{z\}\} \odot M)[P/z] = \{Fv(P) \cup Fv(N) \setminus z\} \odot M = Fv(N[P/z]) \odot M \triangleq$
  $(x \odot M)[N[P/z]/x]$

- $((y <^{y_1}_{y_2} M)[N/x])[P/z] \triangleq (y <^{y_1}_{y_2} M[N/x])[P/z] \triangleq y <^{y_1}_{y_2} M[N/x][P/z] =_{IH} y <^{y_1}_{y_2}$
  $M[[P/z]N/x] =_{IH} (y <^{y_1}_{y_2} M)[[P/z]N/x],\ \ x \neq y$

- $((x <^{x_1}_{x_2} M)[N/x])[P/z] \triangleq (Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M[N_1/x_1, N_2/x_2])[P/z] \triangleq$
  $Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M[N_1/x_1][N_2/x_2][P/z] =$
  $z <^{z_1}_{z_2} Fv(N) \setminus \{z\} <^{Fv(N_1)\setminus\{z_1\}}_{Fv(N_2)\setminus\{z_2\}} M[N_1/x_1][N_2/x_2][P/z] \triangleq$
  $Fv(P) <^{Fv(P_1)}_{Fv(P_2)} Fv(N) \setminus \{z\} <^{Fv(N_1)\setminus\{z_1\}}_{Fv(N_2)\setminus\{z_2\}} M[N_1/x_1][N_2/x_2][P_1/z_1][P_2/z_2] =_{IH}$
  $Fv(P) \cup Fv(N) \setminus \{z\} <^{Fv(P_1)\cup Fv(N_1)\setminus\{z_1\}}_{Fv(P_2)\cup Fv(N_2)\setminus\{z_2\}} M[N_1[P_1/z_1]/x_1, N_2[P_2/z_2]/x_2] \triangleq$
  $(x <^{x_1}_{x_2} M)[N[P/z]/x]$.
  We used the fact that $z_1 \in Fv(N_1)$ and $z_2 \in Fv(N_2)$.

□

In the following lemma, by $\rightarrow^*$ we denote the reflexive and transitive closure of the reductions and equivalences of $\lambda_{\circledR}$-calculus, i.e., $\rightarrow^* \triangleq (\rightarrow_{\lambda_{\circledR}} \cup \equiv_{\lambda_{\circledR}})^*$.

**Lemma 5.**

*(i)* $M[y \odot N/x] \rightarrow^* y \odot M[N/x]$

*(ii)* $y <^{y_1}_{y_2} M[N/x] \rightarrow^* M[y <^{y_1}_{y_2} N/x]$, *for* $y_1, y_2 \notin Fv(M)$.

*Proof.* The proof is by induction on the structure of the term $M$.

(i)
- $M = x$. Then $M[y \odot N/x] = x[y \odot N/x] \triangleq y \odot N \triangleq y \odot x[N/x] = y \odot M[N/x]$.
- $M = \lambda z.P$. Then $M[y \odot N/x] = (\lambda z.P)[y \odot N/x] \triangleq \lambda z.P[y \odot N/x] \rightarrow_{IH} \lambda z.(y \odot$
  $P[N/x]) \rightarrow_{\omega_1} y \odot (\lambda z.P[N/x]) \triangleq y \odot (\lambda z.P)[N/x] = y \odot M[N/x]$.
- $M = PQ$. We will treat the case when $x \notin Fv(Q)$. The case when $x \notin Fv(P)$ is analogous.
  Then $M[y \odot N/x] = (PQ)[y \odot N/x] \triangleq P[y \odot N/x]Q \rightarrow_{IH} (y \odot P[N/x])Q$
  $\rightarrow_{\omega_2} y \odot (P[N/x]Q) \triangleq y \odot (PQ)[N/x] = y \odot M[N/x]$.
- $M = z \odot P$. Then $M[y \odot N/x] = (z \odot P)[y \odot N/x] \triangleq z \odot P[y \odot N/x] \rightarrow_{IH}$
  $z \odot y \odot P[N/x] \equiv_{\varepsilon_1} y \odot z \odot P[N/x] = y \odot M[N/x]$.
- $M = x \odot P$. Then $M[y \odot N/x] = (x \odot P)[y \odot N/x] \triangleq Fv(y \odot N) \odot P = y \odot$
  $Fv(N) \odot P \triangleq y \odot (x \odot P)[N/x] = y \odot M[N/x]$, since $x \notin Fv(P)$.

– $M = z <^{z_1}_{z_2} P$. Then $M[y \odot N/x] = (z <^{z_1}_{z_2} P)[y \odot N/x] \triangleq z <^{z_1}_{z_2} P[y \odot N/x] \rightarrow_{IH}$
$z <^{z_1}_{z_2} (y \odot P[N/x]) \rightarrow_{\gamma\omega_1} y \odot (z <^{z_1}_{z_2} P[N/x]) = y \odot M[N/x]$.

– $M = x <^{x_1}_{x_2} P$. Then $M[y \odot N/x] = (x <^{x_1}_{x_2} P)[y \odot N/x] \triangleq$
$Fv(y \odot N) <^{Fv(y_1 \odot N_1)}_{Fv(y_2 \odot N_2)} P[y_1 \odot N_1/x_1, y_2 \odot N_2/x_2] \rightarrow_{IH}$
$Fv(y \odot N) <^{Fv(y_1 \odot N_1)}_{Fv(y_2 \odot N_2)} y_1 \odot y_2 \odot P[N_1/x_1, N_2/x_2] =$
$Fv(N) <^{Fv(N_1)}_{Fv(N_2)} y <^{y_1}_{y_2} y_1 \odot y_2 \odot P[N_1/x_1, N_2/x_2] \rightarrow_{\gamma\omega_2}$
$Fv(N) <^{Fv(N_1)}_{Fv(N_2)} y \odot P[N_1/x_1, N_2/x_2] \rightarrow_{\gamma\omega_1}$
$y \odot Fv(N) <^{Fv(N_1)}_{Fv(N_2)} P[N_1/x_1, N_2/x_2] \triangleq y \odot (x <^{x_1}_{x_2} P[N/x]) = y \odot M[N/x]$.

(ii) – $M = x$. Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} x[N/x] \triangleq y <^{y_1}_{y_2} N \triangleq x[y <^{y_1}_{y_2} N/x] =$
$M[y <^{y_1}_{y_2} N/x]$.

– $M = \lambda z.P$. Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (\lambda z.P)[N/x] \triangleq y <^{y_1}_{y_2} \lambda z.P[N/x]$
$\rightarrow_{\gamma_1} \lambda z.y <^{y_1}_{y_2} P[N/x] \rightarrow_{IH} \lambda z.P[y <^{y_1}_{y_2} N/x] \triangleq (\lambda z.P)[y <^{y_1}_{y_2} N/x] =$
$M[y <^{y_1}_{y_2} N/x]$.

– $M = PQ$, $x \notin Fv(Q)$. The case when $x \notin Fv(P)$ is analogous.
Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (PQ)[N/x] \triangleq y <^{y_1}_{y_2} P[N/x]Q \rightarrow_{\gamma_2} (y <^{y_1}_{y_2} P[N/x])Q$
$\rightarrow_{IH} P[y <^{y_1}_{y_2} N/x]Q \triangleq (PQ)[y <^{y_1}_{y_2} N/x] = M[y <^{y_1}_{y_2} N/x]$.

– $M = z \odot P$, where $z \neq x, y_1, y_2$. Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (z \odot P)[N/x] \triangleq$
$y <^{y_1}_{y_2} z \odot P[N/x] \rightarrow_{\gamma\omega_1} z \odot y <^{y_1}_{y_2} P[N/x] \rightarrow_{IH} z \odot P[y <^{y_1}_{y_2} N/x] \triangleq (z \odot P)[y <^{y_1}_{y_2}$
$N/x] = M[y <^{y_1}_{y_2} N/x]$.

– $M = x \odot P$. Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (x \odot P)[N/x] \triangleq y <^{y_1}_{y_2} Fv(N) \odot P$.
Since $y_1, y_2 \in Fv(N)$ we have that $y <^{y_1}_{y_2} y_1 \odot y_2 \odot Fv(N) \setminus \{y_1, y_2\} \odot P \rightarrow_{\gamma\omega_2}$
$y \odot Fv(N) \setminus \{y_1, y_2\} \odot P$.
On the other hand, $M[y <^{y_1}_{y_2} N/x] = (x \odot P)[y <^{y_1}_{y_2} N/x] \triangleq Fv(y <^{y_1}_{y_2} N) \odot P =$
$y \odot Fv(N) \setminus \{y_1, y_2\} \odot P$, so the proposition is proved.

– $M = z <^{z_1}_{z_2} P$. Then $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (z <^{z_1}_{z_2} P)[N/x] \triangleq$
$y <^{y_1}_{y_2} z <^{z_1}_{z_2} P[N/x] \equiv_{\lambda_{\circledR}} z <^{z_1}_{z_2} y <^{y_1}_{y_2} P[N/x] \rightarrow_{IH} z <^{z_1}_{z_2} P[y <^{y_1}_{y_2} N/x] \triangleq$
$(z <^{z_1}_{z_2} P)[y <^{y_1}_{y_2} N/x] = M[y <^{y_1}_{y_2} N/x]$.

– $y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} (x <^{x_1}_{x_2} P)[N/x] \triangleq$
$y <^{y_1}_{y_2} Fv(N) <^{Fv(N_1)}_{Fv(N_2)} P[N_1/x_1, N_2/x_2] =$
$y <^{y_1}_{y_2} y_1 <^{y'_1}_{y'_2} y_2 <^{y''_1}_{y''_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\circledR}}$
$y <^{y_1}_{y'_1} y_1 <^{y_2}_{y'_2} y_2 <^{y''_1}_{y''_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\circledR}}$
$y <^{y_1}_{y'_1} y_1 <^{y_2}_{y''_1} y_2 <^{y'_2}_{y''_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\circledR}}$
$y <^{y_1}_{y_2} y_1 <^{y'_1}_{y''_1} y_2 <^{y'_2}_{y''_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\circledR}}$
$y <^{y_1}_{y_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} y_2 <^{y'_2}_{y''_2} y_1 <^{y'_1}_{y''_1} P[N_1/x_1, N_2/x_2] \rightarrow_{IHx2}$
$y <^{y_1}_{y_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[(y_1 <^{y'_1}_{y''_1} N_1)/x_1, (y_2 <^{y'_2}_{y''_2} N_2)/x_2]$
On the other hand, rewriting the right hand side yields:
$M[y <^{y_1}_{y_2} N/x] = (x <^{x_1}_{x_2} P)[y <^{y_1}_{y_2} N/x] \triangleq$

$Fv(y <^{y_1}_{y_2} N) <^{Fv(y_3 <^{y_1}_{y_2} N'_1)}_{Fv(y_4 <^{y_1}_{y_2} N'_2)} P[(y_3 <^{y_1}_{y_2} N'_1)/x_1, (y_4 <^{y_1}_{y_2} N'_2)/x_2]$
By renaming $y_1 \to y'_1$ and $y_2 \to y''_1$ in $y_3 <^{y_1}_{y_2} N'_1$ and $y_1 \to y'_2$ and $y_2 \to y''_2$ in $y_4 <^{y_1}_{y_2} N'_2$ we get
$Fv(y <^{y_1}_{y_2} N) <^{Fv(y_3 <^{y'_1}_{y''_1} N_1)}_{Fv(y_4 <^{y'_2}_{y''_2} N_2)} P[(y_3 <^{y'_1}_{y''_1} N_1)/x_1, (y_4 <^{y'_2}_{y''_2} N_2)/x_2]$
where $N'_1[y'_1/y_1, y''_1/y_2] = N_1$ and $N'_2[y'_2/y_1, y''_2/y_2] = N_2$.

Finally, by renaming $y_3 \to y_1$ and $y_4 \to y_2$ we get
$Fv(y <^{y_1}_{y_2} N) <^{Fv(y_1 <^{y'_1}_{y''_1} N_1)}_{Fv(y_2 <^{y'_2}_{y''_2} N_2)} P[(y_1 <^{y'_1}_{y''_1} N_1)/x_1, (y_2 <^{y'_2}_{y''_2} N_2)/x_2] =$
$y <^{y_1}_{y_2} Fv(N) \setminus \{y_1, y_2\} <^{Fv(N_1)\setminus\{y'_1, y''_1\}}_{Fv(N_2)\setminus\{y'_2, y''_2\}} P[(y_1 <^{y'_1}_{y''_1} N_1)/x_1, (y_2 <^{y'_2}_{y''_2} N_2)/x_2]$
which completes the proof.

□

Since the last case of the previous lemma is a bit tricky, let us illustrate it with the following example.

**Example 6.** Let $M = x <^{x_1}_{x_2} x_1x_2$ and $N = y_1y_2$. Then
$y <^{y_1}_{y_2} M[N/x] = y <^{y_1}_{y_2} x <^{x_1}_{x_2} x_1x_2[(y_1y_2)/x] \triangleq$
$y <^{y_1}_{y_2} Fv(y_1y_2) <^{Fv(z_1z_2)}_{Fv(w_1w_2)} x_1x_2[(z_1z_2)/x_1, (w_1w_2)/x_2] =$
$y <^{y_1}_{y_2} y_1 <^{z_1}_{w_1} y_2 <^{z_2}_{w_2} x_1x_2[(z_1z_2)/x_1, (w_1w_2)/x_2] \equiv_{\lambda_{\circledR}, (3\times\varepsilon_3)}$
$y <^{y_1}_{y_2} y_1 <^{z_1}_{z_2} y_2 <^{w_1}_{w_2} (z_1z_2)(w_1w_2) = M_1$.
On the other hand:
$x <^{x_1}_{x_2} x_1x_2[y <^{y_1}_{y_2} y_1y_2/x] \triangleq$
$Fv(y <^{y_1}_{y_2} y_1y_2) <^{Fv(y_3 <^{y_1}_{y_2} y_1y_2)}_{Fv(y_4 <^{y_1}_{y_2} y_1y_2)} (x_1x_2)[(y_3 <^{y_1}_{y_2} y_1y_2)/x_1, (y_4 <^{y_1}_{y_2} y_1y_2)/x_2] =$
$y <^{y_3}_{y_4} (y_3 <^{y_1}_{y_2} y_1y_2)(y_4 <^{y_1}_{y_2} y_1y_2)$.
By renaming $y_1 \to z_1$, $y_2 \to z_2$ in the first bracket, and $y_1 \to w_1$, $y_2 \to w_2$ in the second one we obtain: $y <^{y_3}_{y_4} (y_3 <^{z_1}_{z_2} z_1z_2)(y_4 <^{w_1}_{w_2} w_1w_2)$.
By renaming $y_3 \to y_1$, $y_4 \to y_2$ we get $y <^{y_1}_{y_2} (y_1 <^{z_1}_{z_2} z_1z_2)(y_2 <^{w_1}_{w_2} w_1w_2) = M_2$.
Finally, $M_1 \to_{\gamma_2, \gamma_3} M_2$.

## 1.2 Intersection types for $\lambda_{\circledR}$

In this subsection we introduce an intersection type assignment system which assigns *strict types* to $\lambda_{\circledR}$-terms. Strict types were proposed in [60] and used in [22] for characterisation of strong normalisation in $\lambda^{\mathsf{Gtz}}$-calculus.

The syntax of types is defined as follows:

| | | | |
|---|---|---|---|
| Strict types | $\sigma$ | $::=$ | $p \mid \alpha \to \sigma$ |
| Types | $\alpha$ | $::=$ | $\cap_i^n \sigma_i$ |

where $p$ ranges over a denumerable set of type atoms, and $\cap_i^n \sigma_i$ stands for $\sigma_1 \cap \ldots \cap \sigma_n$, $n \geq 0$. Particularly, if $n = 0$, then $\cap_i^0 \sigma_i$ represents the *neutral element* for the intersection operator, denoted by $\top$.
We denote types with $\alpha, \beta, \gamma$..., strict types with $\sigma, \tau, \upsilon$... and the set of all types by Types. We assume that the intersection operator is idempotent, commutative and associative. We also assume that intersection has priority over the arrow operator. Hence, we will omit parenthesis in expressions like $(\cap_i^n \tau_i) \to \sigma$.

**Definition 7.**

(i) A *basic type assignment* is an expression of the form $x : \alpha$, where $x$ is a term variable and $\alpha$ is a type.

(ii) A *basis* $\Gamma$ is a set $\{x_1 : \alpha_1, \ldots, x_n : \alpha_n\}$ of basic type assignments, where all term variables are different. $Dom(\Gamma) = \{x_1, \ldots, x_n\}$. A basis extension $\Gamma, x : \alpha$ denotes the set $\Gamma \cup \{x : \alpha\}$, where $x \notin Dom(\Gamma)$.

(iii) A *bases intersection* is defined as:

$$\Gamma \sqcap \Delta = \{x : \alpha \cap \beta \mid x : \alpha \in \Gamma \,\&\, x : \beta \in \Delta \,\&\, Dom(\Gamma) = Dom(\Delta)\}.$$

(iv) $\Gamma^\top = \{x : \top \mid x \in Dom(\Gamma)\}$.

In what follows we assume that the bases intersection has priority over the basis extension, hence the parenthesis in $\Gamma, (\Delta_1 \sqcap \ldots \sqcap \Delta_n)$ will be omitted. It is easy to show that $\Gamma^\top \sqcap \Delta = \Delta$ for arbitrary bases $\Gamma$ and $\Delta$ that can be intersected, hence $\Gamma^\top$ can be considered the neutral element for the bases intersection.

$$\frac{}{x : \sigma \vdash x : \sigma}\ (Ax)$$

$$\frac{\Gamma, x : \alpha \vdash M : \sigma}{\Gamma \vdash \lambda x.M : \alpha \to \sigma}\ (\to_I) \qquad \frac{\Gamma \vdash M : \cap_i^n \tau_i \to \sigma \quad \Delta_0 \vdash N : \tau_0 \ \ldots \ \Delta_n \vdash N : \tau_n}{\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash MN : \sigma}\ (\to_E)$$

$$\frac{\Gamma, x : \alpha, y : \beta \vdash M : \sigma}{\Gamma, z : \alpha \cap \beta \vdash z <_y^x M : \sigma}\ (Cont) \qquad \frac{\Gamma \vdash M : \sigma}{\Gamma, x : \top \vdash x \odot M : \sigma}\ (Weak)$$

Figure 5: $\lambda_{\circledR}\cap$: $\lambda_{\circledR}$-calculus with intersection types

The type assignment system $\lambda_{\circledR}\cap$ is given in Figure 5. Notice that in the syntax of $\lambda_{\circledR}$ there are three kinds of variables according to the way they are introduced, namely as a placeholder, as a result of a contraction or as a result of a weakening. Each kind of a variable receives a specific type. Variables as placeholders have a strict type, variables resulting from a contraction have an intersection type and variables resulting from a weakening have a $\top$ type. Moreover, notice that intersection types occur only in two inference rules. In the rule $(Cont)$ the intersection type is created, this being *the only* place where this happens. This is justified because it corresponds to the duplication of

a variable. In other words, the control on the duplication of variables entails the control on the introduction of intersections in building the type of the term in question. In the rule $(\to_E)$, intersection appears on the right hand side of $\vdash$ sign which corresponds to the usage of the intersection type after it has been created by the rule $(Cont)$ or by the rule $(Weak)$ if $n = 0$. In this inference rule, the role of $\Delta_0$ should be noticed. It is needed only when $n = 0$ to ensure that $N$ has a type, i.e. that $N$ is strongly normalizing. Then, in the bottom of the rule, the types of the free variables of $N$ can be forgotten, hence all the free variables of $N$ receive the type $\top$. All the free variables of the term must occur in the environment (see Lemma 8), therefore useless variables occur with the type $\top$. If $n$ is not 0, then $\Delta_0$ can be any of the other environments and the type of $N$ the associated type. Since $\Delta^\top$ is a neutral element for $\sqcap$, then $\Delta^\top$ disappears in the bottom of the rule. The case for $n = 0$ resembles the rules $(drop)$ and/or *(K-cup)* in [42] and was used to present the two cases, $n = 0$ and $n \neq 0$ in a uniform way. In the rule $(Weak)$ the choice of the type of $x$ is $\top$, since this corresponds to a variable which does not occur anywhere in $M$. The remaining rules, namely $(Ax)$ and $(\to_I)$ are traditional, i.e. they are the same as in the simply typed $\lambda$-calculus. Noticed however that the type of the variable in $(Ax)$ is a strict type.

**Lemma 8** (Domain Correspondence for $\lambda_{\circledR}\cap$)**.** *Let $\Gamma \vdash M : \sigma$ be a typing judgment. Then $x \in Dom(\Gamma)$ if and only if $x \in Fv(M)$.*

*Proof.* The rules of Figure 5 belong to three categories.

1. *The rules that introduce a variable*. These rules are *(Ax)*, $(Cont)$ and $(Weak)$. One sees that the variable is introduced in the environment if and only it is introduced in the term as a free variable.

2. *The rules that remove variables*. These rules are $(\to_I)$ and $(Cont)$. One sees that the variables are removed from the environment if and only if they are removed from the term as a free variable.

3. *The rule that does not introduce and does not remove a variable*. This rule is $(\to_E)$.

Notice that $(Cont)$ introduces and removes variables. □

The Generation Lemma makes somewhat more precise the Domain Correspondence Lemma.

**Lemma 9** (Generation lemma for $\lambda_{\circledR}\cap$)**.**

*(i)* $\Gamma \vdash \lambda x.M : \tau$ *iff there exist* $\alpha$ *and* $\sigma$ *such that* $\tau \equiv \alpha \to \sigma$ *and* $\Gamma, x : \alpha \vdash M : \sigma$.

*(ii)* $\Gamma \vdash MN : \sigma$ *iff and there exist* $\Delta_i$ *and* $\tau_i$, $i = 0, \ldots, n$ *such that* $\Gamma' \vdash M : \cap_i^n \tau_i \to \sigma$ *and for all* $i \in \{0, \ldots, n\}$, $\Delta_i \vdash N : \tau_i$ *and* $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.

*(iii)* $\Gamma \vdash z <_y^x M : \sigma$ *iff there exist* $\Gamma', \alpha, \beta$ *such that* $\Gamma = \Gamma', z : \alpha \cap \beta$ *and* $\Gamma', x : \alpha, y : \beta \vdash M : \sigma$.

*(iv)* $\Gamma \vdash x \odot M : \sigma$ *iff* $\Gamma = \Gamma', x : \top$ *and* $\Gamma' \vdash M : \sigma$.

*Proof.* The proof is straightforward since all the rules are syntax directed. □

The proposed system satisfies the following properties.

**Lemma 10** (Substitution lemma for $\lambda_{\circledR}\cap$)**.** *If* $\Gamma, x : \cap_i^n \tau_i \vdash M : \sigma$ *and for all* $i \in \{0,\ldots,n\}$, $\Delta_i \vdash N : \tau_i$, *then* $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash M[N/x] : \sigma$.

*Proof.* The proof is by induction on the structure of the term $M$. We only show the interesting cases.

- Base case $M = x$. By the axiom $x : \tau \vdash x : \tau$ where $\tau = \cap_i^1 \tau_i$, i.e. $n = 1$, hence the second assumption is $\Delta \vdash N : \tau$ which proves the case since $N \triangleq x[N/x]$.
- $M = x \odot P$. Now we assume $\Gamma, x : \cap_i^0 \tau_i \vdash x \odot P : \sigma$ and $\Delta_i \vdash N : \tau_i$ for all $i \in \{0,\ldots,0\}$, in other words $\Gamma, x : \top \vdash x \odot P : \sigma$ and $\Delta_0 \vdash N : \tau_0$ (i.e. $N$ is typeable). By Generation lemma 9$(iv)$ we get $\Gamma \vdash P : \sigma$. Since $Dom(\Delta_0) = Fv(N)$ by applying the $(Weak)$ rule multiple times we get $\Gamma, \Delta_0^\top \vdash Fv(N) \odot P \vdash \sigma$ which is exactly what we want to prove.
- $M = x <_{x_2}^{x_1} P$. From $\Gamma, x : \cap_i^n \tau_i \vdash x <_{x_2}^{x_1} P : \sigma$, by Generation lemma 9$(iii)$ we get that $\cap_i^n \tau_i = \cap_{i=1}^m \tau_i \cap \cap_{i=m+1}^n \tau_i$ for some $m < n$ and $\Gamma, x_1 : \cap_i^m \tau_i, x_2 : \cap_{i=m+1}^n \tau_i \vdash P : \sigma$. From the other assumption $\Delta_i \vdash N : \tau_i$ for all $i \in \{1,\ldots,n\}$, by renaming the variables in $\Delta_i$ (i.e. the free variables of $N$) we get two different sets of sequents: $\Delta'_j \vdash N_1 : \tau_j$ for $j = 1,\ldots,m$ and $\Delta''_k \vdash N_2 : \tau_k$ for $k = m+1,\ldots,n$. By applying IH twice, we get $\Gamma, \Delta'^\top_0 \sqcap \Delta'_1 \sqcap \ldots \sqcap \Delta'_m, \Delta''^\top_0 \sqcap \Delta''_{m+1} \sqcap \ldots \sqcap \Delta''_n \vdash (P[N_1/x_1])[N_2/x_2] : \sigma$. Now, we apply the definition of the parallel substitution, and perform contraction on all pairs of corresponding (i.e. obtained by the renaming of the same variable) elements of $\Delta'_j$ and $\Delta''_k$ by introducing again the original names of the free variables of $N$ from $\Delta_i$ and finally get what we need:

$$\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash Fv(N) <_{Fv(N_2)}^{Fv(N_1)} P[N_1/x_1, N_2/x_2] : \sigma.$$

□

**Proposition 11** (Subject reduction and equivalence)**.** *For every* $\lambda_{\circledR}$*-term* $M$*: if* $\Gamma \vdash M : \sigma$ *and* $M \to M'$ *or* $M \equiv M$, *then* $\Gamma \vdash M' : \sigma$.

*Proof.* The proof is done by the case analysis on the applied reduction. Since the property is stable by context, we can without losing generality assume that the reduction takes place at the outermost position of the term. Here we just show several cases. We will use GL as an abbreviation for Generation lemma 9.

- Case $(\beta)$: Let $\Gamma \vdash (\lambda x.M)N : \sigma$. We want to show that $\Gamma \vdash M[N/x] : \sigma$. From $\Gamma \vdash (\lambda x.M)N : \sigma$ and from GL(ii) it follows that $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$, and that there is a type $\cap_i^n \tau_i$ such that for all $i = 0,\ldots,n$, $\Delta_i \vdash N : \tau_i$, and $\Gamma' \vdash \lambda x.M : \cap_i^n \tau_i \to \sigma$. Further, by GL(i) we have that $\Gamma', x : \cap_i^n \tau_i \vdash M : \sigma$. Now, all the assumptions of Substitution lemma 10 hold, yielding $\Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash M[N/x] : \sigma$ which is exactly what we need, since $\Gamma = \Delta_0^\top \sqcap \Gamma', \Delta_1 \sqcap \ldots \sqcap \Delta_n$.

- Case ($\gamma\omega_2$): Let $\Gamma \vdash x <^{x_1}_{x_2} x_1 \odot M : \sigma$. We are showing that $\Gamma \vdash M[x/x_2] : \sigma$. From the first sequent by GL(iii) we have that $\Gamma = \Gamma', x : \alpha \cap \beta$ and $\Gamma', x_1 : \alpha, x_2 : \beta \vdash x_1 \odot M : \sigma$. Further, by GL(iv) we conclude that $\alpha \equiv \top$, $x : \top \cap \beta \equiv \beta$ and $\Gamma', x_2 : \beta \vdash M : \sigma$. Since $\beta = \cap_i^n \tau_i$ for some $n \geq 0$, by applying Substitution lemma 10 to $\Gamma', x_2 : \beta \vdash M : \sigma$ and $x : \tau_i \vdash x : \tau_i$, $i = 0, \ldots, n$ we get $\Gamma \vdash M[x/x_2] : \sigma$.

- The other rules are easy since they do not essentially change the structure of the term.

□

Due to this property, equivalent terms have the same type.

## 1.3 Typeability ⇒ SN in $\lambda_{\circledR}\cap$

In various type assignment systems, the *reducibility method* can be used to prove many reduction properties of typeable terms. It was first introduced by Tait [58] for proving the strong normalisation of simply typed λ-calculus, and developed further to prove *strong normalisation* of various calculi in [59, 30, 41, 25, 29], *confluence* (the Church-Rosser property) of βη-reduction in [40, 57, 44, 45, 29] and to characterise certain classes of λ-terms such as strongly normalising, normalising, head normalising, and weak head normalising terms (and their persistent versions) by their typeability in various intersection type systems in [23, 17, 15, 16].

The main idea of the reducibility method is to interpret types by suitable sets of lambda terms which satisfy some realizability properties and prove the soundness of type assignment with respect to these interpretations. A consequence of soundness is that every typeable term belongs to the interpretation of its type, hence satisfying a desired reduction property.

In the remainder of the paper we consider $\Lambda_{\circledR}$ as the *applicative structure* whose domain are $\lambda_{\circledR}$-terms and where the application is just the application of $\lambda_{\circledR}$-terms. The set of *strongly normalizing terms* is defined as the smallest subset of $\Lambda_{\circledR}$ such that:

$$\frac{M' \in \mathcal{SN} \qquad M \to M'}{M \in \mathcal{SN}}$$

**Definition 12.** For $\mathcal{M}, \mathcal{N} \subseteq \Lambda_{\circledR}$, we define $\mathcal{M} \longrightarrow \mathcal{N} \subseteq \Lambda_{\circledR}$ as

$$\mathcal{M} \longrightarrow \mathcal{N} = \{N \in \Lambda_{\circledR} \mid \forall M \in \mathcal{M} \quad NM \in \mathcal{N})\}.$$

**Definition 13.** The type interpretation $[\![-]\!] : \mathsf{Types} \to 2^{\Lambda_{\circledR}}$ is defined by:

(*I*1) $[\![p]\!] = \mathcal{SN}$, where $p$ is a type atom;

(*I*2) $[\![\alpha \to \sigma]\!] = [\![\alpha]\!] \longrightarrow [\![\sigma]\!]$;

(*I*3) $[\![\cap_i^n \sigma_i]\!] = \cap_i^n [\![\sigma_i]\!]$ and $[\![\cap_i^0 \sigma_i]\!] = \mathcal{SN}$.

Next, we introduce the notions of *variable property, reduction property, expansion property, weakening property* and *contraction property.* Variable property and expansion property correspond to the saturation property given in [5], whereas reduction property corresponds to the property CR 2 in Chapter 6 of [32]. To this aim we will use the following notation: recall that Ⓡ denotes the set of reductions given in Figure 2. If $\mu \in Ⓡ$, then $redex_\mu^\varsigma$ denotes a redex, that is a term which is an instance by the meta-substitution $\varsigma$ of the left hand side of the reduction $\mu$. Whereas $contr_\mu^\varsigma$ denotes the instance of the right hand side of the same reduction $\mu$ by the same meta-substitution $\varsigma$.[3]

**Definition 14.**

- A set $\mathcal{X} \subseteq \Lambda_Ⓡ$ satisfies the *variable property*, notation $\mathsf{VAR}(\mathcal{X})$, if $\mathcal{X}$ contains all the terms of the form $xM_1 \dots M_n$ for $M_i \in \mathcal{SN}$.
- A set $\mathcal{X} \subseteq \Lambda_Ⓡ$ satisfies the *reduction property*, notation $\mathsf{RED}(\mathcal{X})$, if $\mathcal{X}$ is stable by reduction, in other words $M \in \mathcal{X}$ and $M \to M'$ imply $M' \in \mathcal{X}$.
- A set $\mathcal{X} \subseteq \Lambda_Ⓡ$ satisfies the *expansion property*, notation $\mathsf{EXP}_\mu(\mathcal{X})$ where $\mu$ is a rule in Ⓡ, if[4]:

$$\frac{M_1 \in \mathcal{SN} \dots M_n \in \mathcal{SN} \qquad contr_\mu^\varsigma\, M_1 \dots M_n \in \mathcal{X}}{redex_\mu^\varsigma\, M_1 \dots M_n \in \mathcal{X}.}\ \mathsf{EXP}_\mu(\mathcal{X})$$

- A set $\mathcal{X} \subseteq \Lambda_Ⓡ$ satisfies the *weakening property*, notation $\mathsf{WEAK}(\mathcal{X})$ if:

$$\frac{M \in \mathcal{X}}{x \odot M \in \mathcal{X}.}\ \mathsf{WEAK}(\mathcal{X})$$

- A set $X \subseteq \Lambda_Ⓡ$ satisfies the *contraction property*, notation $\mathsf{CONT}(\mathcal{X})$ if:

$$\frac{M \in \mathcal{X}}{x <_z^y M \in \mathcal{X}.}\ \mathsf{CONT}(\mathcal{X})$$

**Remark.** In the previous definition (Definition 14) it is not necessary to explicitly write the conditions about free variables since we work with $\lambda_Ⓡ$-terms.

**Definition 15** (Ⓡ-Saturated set)**.** A set $\mathcal{X} \subseteq \Lambda_Ⓡ$ is called Ⓡ*-saturated*, if

- $\mathcal{X} \subseteq \mathcal{SN}$ and
- $\mathcal{X}$ satisfies the variable, reduction, expansion, weakening and contraction properties.

**Proposition 16.** *Let* $\mathcal{M}, \mathcal{N} \subseteq \Lambda_Ⓡ$.

[3] Meta-substitution is a substitution that assigns values to meta-variables.

[4] Notice that we do not need a condition that $N \in \mathcal{SN}$ in $\mathsf{EXP}_\beta(\mathcal{X})$, as in ordinary $\lambda$-calculus, since we only work with linear terms, hence if the contractum $M[N/x] \in \mathcal{SN}$, then $N \in \mathcal{SN}$.

*(i)* $\mathcal{SN}$ *is* ®*-saturated.*

*(ii) If* $\mathcal{M}$ *and* $\mathcal{N}$ *are* ®*-saturated, then* $\mathcal{M} \longrightarrow \mathcal{N}$ *is* ®*-saturated.*

*(iii) If* $\mathcal{M}$ *and* $\mathcal{N}$ *are* ®*-saturated, then* $\mathcal{M} \cap \mathcal{N}$ *is* ®*-saturated.*

*(iv) For all types* $\varphi \in Types$, $[\![\varphi]\!]$ *is* ®*-saturated.*

*Proof.* (i)

- $\mathcal{SN} \subseteq \mathcal{SN}$, $\mathsf{VAR}(\mathcal{SN})$ and $\mathsf{RED}(\mathcal{SN})$ trivially hold.
- $\mathsf{EXP}_\beta(\mathcal{SN})$. Suppose that $M[N/x]M_1 \ldots M_n \in \mathcal{SN}$ and $M_1, \ldots, M_n \in \mathcal{SN}$. Since $M[N/x]$ is a subterm of a term in $\mathcal{SN}$, we know that $M \in \mathcal{SN}$. Also, since $M[N/x] \in \mathcal{SN}$ and $M$ is linear, $N \in \mathcal{SN}$. By assumption, $M_1, \ldots, M_n \in \mathcal{SN}$, so the reductions inside of these terms terminate. After finitely many reduction steps, we obtain $(\lambda x.M)NM_1 \ldots M_n \to \ldots \to (\lambda x.M')N'M'_1 \ldots M'_n$ where $M \to M'$, $N \to N'$, $M_1 \to M'_1, \ldots, M_n \to M'_n$. After contracting $(\lambda x.M')N'M'_1 \ldots M'_n$ to $M'[N'/x]M'_1 \ldots M'_n$, we obtain a reduct of $M[N/x]M_1 \ldots M_n \in \mathcal{SN}$. Hence, $(\lambda x.M)NM_1 \ldots M_n \in \mathcal{SN}$.
- $\mathsf{EXP}_\mu(\mathcal{SN})$. Analogous to $\mathsf{EXP}_\beta(\mathcal{SN})$.
- $\mathsf{WEAK}(\mathcal{SN})$. Suppose that $M \in \mathcal{SN}$ and $x \notin Fv(M)$. Then trivially $x \odot M \in \mathcal{SN}$, since no new redexes are formed.
- $\mathsf{CONT}(\mathcal{SN})$. Suppose that $M \in \mathcal{SN}$, $y \neq z$, $y,z \in Fv(M)$, $x \notin Fv(M) \setminus \{y,z\}$. We prove $x <^y_z M \in \mathcal{SN}$ by induction on the structure of $M$.
  - $M = \lambda w.N$. Then $N \in \mathcal{SN}$ and $x <^y_z M = x <^y_z (\lambda w.N) \to_{\gamma_1} \lambda w.x <^y_z N \in \mathcal{SN}$, since $x <^y_z N \in \mathcal{SN}$ by IH.
  - $M = PQ$. Then $P,Q \in \mathcal{SN}$ and if $y,z \notin Fv(Q)$, $x <^y_z M = x <^y_z (PQ) \to_{\gamma_2} (x <^y_z P)Q \in \mathcal{SN}$, since by IH $x <^y_z P \in \mathcal{SN}$.
    The case of $\to_{\gamma_3}$ reduction is analogous.
  - $M = w \odot N$. Then $x <^y_z M = x <^y_z (w \odot N) \to_{\gamma\omega_1} w \odot (x <^y_z N)$. By IH $x <^y_z N \in \mathcal{SN}$ and $w \odot (x <^y_z N)$ does not introduce any new redexes.
  - $M = y \odot N$. Then $x <^y_z M = x <^y_z (y \odot N) \to_{\gamma\omega_2} N[x/z] \in \mathcal{SN}$, since $N \in \mathcal{SN}$ by IH.

(ii)

- $\mathcal{M} \longrightarrow \mathcal{N} \subseteq \mathcal{SN}$. Suppose that $M \in \mathcal{M} \longrightarrow \mathcal{N}$. Then, for all $N \in \mathcal{M}$, $MN \in \mathcal{N}$. Since $\mathcal{M}$ is ®-saturated, $\mathsf{VAR}(\mathcal{M})$ holds so $x \in \mathcal{M}$ and $Mx \in \mathcal{N} \subseteq \mathcal{SN}$. From here we can deduce that $M \in \mathcal{SN}$.
- $\mathsf{VAR}(\mathcal{M} \longrightarrow \mathcal{N})$. Suppose that $x \in \mathtt{var}$, and $M_1, \ldots, M_n \in \mathcal{SN}, n \geq 0$, such that $x \cap Fv(M_1) \cap \ldots \cap Fv(M_n) = \emptyset$. We need to show that $xM_1 \ldots M_n \in \mathcal{M} \longrightarrow \mathcal{N}$, i.e. $\forall N \in \mathcal{M}$, $xM_1 \ldots M_nN \in \mathcal{N}$. This holds since by IH $\mathcal{M} \subseteq \mathcal{SN}$ and $\mathcal{N}$ is ®-saturated, i.e. $\mathsf{VAR}(\mathcal{N})$ holds.

- $\mathsf{RED}(\mathcal{M} \longrightarrow \mathcal{N})$. Let $M \in \mathcal{M} \longrightarrow \mathcal{N}$ and $M'$ be such that $M \to M'$ and let $N \in \mathcal{M}$. We know that $MN \in \mathcal{N}$ and $MN \to M'N$. By IH, $M'N \in \mathcal{N}$ hence $M' \in \mathcal{M} \longrightarrow \mathcal{N}$.

- $\mathsf{EXP}_\beta(\mathcal{M} \longrightarrow \mathcal{N})$. Suppose that $M[N/x]M_1 \ldots M_n \in \mathcal{M} \longrightarrow \mathcal{N}$ and $M_1, \ldots, M_n \in \mathcal{SN}$. This means that for all $P \in \mathcal{M}$, $M[N/x]M_1 \ldots M_nP \in \mathcal{N}$. But $\mathcal{N}$ is ®-saturated, so $\mathsf{EXP}_\beta(\mathcal{N})$ holds and we have that for all $P \in \mathcal{N}$, $(\lambda x.M)NM_1 \ldots M_nP \in \mathcal{N}$. This means that $(\lambda x.M)NM_1 \ldots M_n \in \mathcal{M} \longrightarrow \mathcal{N}$.

- $\mathsf{EXP}_\mu(\mathcal{M} \longrightarrow \mathcal{N})$. Analogous to $\mathsf{EXP}_\beta(\mathcal{M} \longrightarrow \mathcal{N})$.

- $\mathsf{WEAK}(\mathcal{M} \longrightarrow \mathcal{N})$. Suppose that $M \in \mathcal{M} \longrightarrow \mathcal{N}$ and $x \notin Fv(M)$. This means that for all $N \in \mathcal{M}, MN \in \mathcal{N}$. But $\mathcal{N}$ is ®-saturated, i.e. $\mathsf{WEAK}(\mathcal{N})$ holds, hence $x \odot (MN) \in \mathcal{N}$. Also $\mathsf{EXP}_{\omega_2}(\mathcal{N})$ holds so we obtain for all $N \in \mathcal{M}, (x \odot M)N \in \mathcal{N}$, i.e $x \odot M \in \mathcal{M} \longrightarrow \mathcal{N}$.

- $\mathsf{CONT}(\mathcal{M} \longrightarrow \mathcal{N})$. Let $M \in \mathcal{M} \longrightarrow \mathcal{N}$. We want to prove that $x <^y_z M \in \mathcal{M} \longrightarrow \mathcal{N}$ for $y \neq z$, $y,z \in Fv(M)$ and $x \notin Fv(M)$. Let $P$ be any term in $\mathcal{M}$. We have to prove that $(x <^y_z M)P \in \mathcal{N}$. Since $M \in \mathcal{M} \longrightarrow \mathcal{N}$, we know that $MP \in \mathcal{N}$. By IH $x <^y_z (MP) \in \mathcal{N}$. By reduction $\gamma_2$ and hence by $\mathsf{RED}(\mathcal{N})$ we have $(x <^y_z M)P \in \mathcal{N}$. Therefore $x <^y_z M \in \mathcal{M} \longrightarrow \mathcal{N}$.

(iii)

- $\mathcal{M} \cap \mathcal{N} \subseteq \mathcal{SN}$ is straightforward, since $M, N \subseteq \mathcal{SN}$ by IH.

- $\mathsf{VAR}(\mathcal{M} \cap \mathcal{N})$. Since $\mathsf{VAR}(\mathcal{M})$ and $\mathsf{VAR}(\mathcal{N})$ hold, we have that $\forall M_1, \ldots, M_n \in \mathcal{SN}, n \geq 0$: $xM_1 \ldots M_n \in \mathcal{M}$ and $xM_1 \ldots M_n \in \mathcal{N}$. We deduce that $\forall M_1, \ldots, M_n \in \mathcal{SN}, n \geq 0$: $xM_1 \ldots M_n \in \mathcal{M} \cap \mathcal{N}$, i.e. $\mathsf{VAR}(\mathcal{M} \cap \mathcal{N})$ holds.

- $\mathsf{RED}(\mathcal{M} \cap \mathcal{N})$ is straightforward.

- $\mathsf{EXP}_\beta(\mathcal{M} \cap \mathcal{N})$ and $\mathsf{EXP}_\mu(\mathcal{M} \cap \mathcal{N})$ are straightforward.

- $\mathsf{WEAK}(\mathcal{M} \cap \mathcal{N})$. Let $M \in \mathcal{M} \cap \mathcal{N}$ and $x \notin Fv(M)$. Then $M \in \mathcal{M}$ and $M \in \mathcal{N}$. Since both $\mathcal{M}$ and $\mathcal{N}$ are ®-saturated $\mathsf{WEAK}(\mathcal{M})$ and $\mathsf{WEAK}(\mathcal{N})$ hold, hence by IH $x \odot M \in \mathcal{M}$ and $x \odot M \in \mathcal{N}$, i.e. $x \odot M \in \mathcal{M} \cap \mathcal{N}$.

- $\mathsf{CONT}(\mathcal{M} \cap \mathcal{N})$. Suppose that $M \in \mathcal{M} \cap \mathcal{N}$, $y \neq z$, $y,z \in Fv(M)$, $x \notin Fv(M) \setminus \{y,z\}$. Since both $\mathcal{M}$ and $\mathcal{N}$ are ®-saturated $\mathsf{CONT}(\mathcal{M})$ and $\mathsf{CONT}(\mathcal{N})$ hold, hence by IH $x <^y_z M \in \mathcal{M}$ and $x <^y_z M \in \mathcal{N}$, i.e. $x <^y_z M \in \mathcal{M} \cap \mathcal{N}$.

(iv) By induction on the construction of $\varphi \in \mathsf{Types}$.

- If $\varphi \equiv p$, $p$ a type atom, then $[\![\varphi]\!] = \mathcal{SN}$, so it is ®-saturated using (i).

- If $\varphi \equiv \alpha \to \sigma$, then $[\![\varphi]\!] = [\![\alpha]\!] \longrightarrow [\![\sigma]\!]$. Since $[\![\alpha]\!]$ and $[\![\sigma]\!]$ are ®-saturated by IH, we can use (ii).

- If $\varphi \equiv \cap^n_i \sigma_i$, then $[\![\varphi]\!] = [\![\cap^n_i \sigma_i]\!] = \cap^n_i [\![\sigma_i]\!]$ and for all $i = 1, \ldots, n$, $[\![\sigma_i]\!]$ are ®-saturated by IH, so we can use (iii). If $\varphi \equiv \cap^0_i \sigma_i$, then $[\![\varphi]\!] = \mathcal{SN}$

□

We further define a *valuation of terms* $[\![-]\!]_\rho : \Lambda_{\circledR} \to \Lambda_{\circledR}$ and the *semantic satisfiability relation* $\models$ connecting the type interpretation with the term valuation.

**Definition 17.** Let $\rho : \mathtt{var} \to \Lambda_{\circledR}$ be a valuation of term variables in $\Lambda_{\circledR}$. For $M \in \Lambda_{\circledR}$, with $Fv(M) = \{x_1, \ldots, x_n\}$ the *term valuation* $[\![-]\!]_\rho : \Lambda_{\circledR} \to \Lambda_{\circledR}$ is defined as follows:

$$[\![M]\!]_\rho = M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n].$$

providing that $x \neq y \;\Rightarrow\; Fv(\rho(x)) \cap Fv(\rho(y)) = \emptyset$.

*Notation:* $\rho(N/x)$ is the valuation defined as: $\rho(N/x)(y) = \begin{cases} \rho(y) & \text{if } x \neq y \\ N & \text{otherwise} \end{cases}$

**Lemma 18.**

*(i)* $[\![MN]\!]_\rho = [\![M]\!]_\rho [\![N]\!]_\rho$

*(ii)* $[\![\lambda x.M]\!]_\rho N \to [\![M]\!]_{\rho(N/x)}$.

*(iii)* $[\![x \odot M]\!]_\rho = Fv(\rho(x)) \odot [\![M]\!]_\rho$.

*(iv)* $[\![z <^x_y M]\!]_\rho = Fv(N) <^{Fv(N_1)}_{Fv(N_2)} [\![M]\!]_{\rho(N_1/x, N_2/y)}$
*where $N = \rho(z)$, $N_1$, $N_2$ are obtained from $N$ by renaming its free variables.*

*Proof.*

(i) Straightforward from the definition of substitution given in Figure 3.

(ii) If $Fv(\lambda x.M) = \{x_1, \ldots, x_n\}$, then
$[\![\lambda x.M]\!]_\rho N = (\lambda x.M)[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n]N \to$
$(M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n])[N/x] = M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n, N/x] =$
$[\![M]\!]_{\rho(N/x)}$,

(iii) If $Fv(M) = \{x_1, \ldots, x_n\}$, then
$[\![x \odot M]\!]_\rho = (x \odot M)[\rho(x)/x, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] =$
$Fv(\rho(x)) \odot M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] = Fv(\rho(x)) \odot [\![M]\!]_\rho$.

(iv) If $Fv(M) = \{x_1, \ldots, x_n\}$, then
$[\![z <^x_y M]\!]_\rho = (z <^x_y M)[N/z, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] =$
$Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M[N_1/x, N_2/y, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] =$
$= Fv(N) <^{Fv(N_1)}_{Fv(N_2)} [\![M]\!]_{\rho(N_1/x, N_2/y)}$.

□

**Definition 19.**

(i) $\rho \models M : \sigma \quad \Longleftrightarrow \quad [\![M]\!]_\rho \in [\![\sigma]\!]$;

(ii) $\rho \models \Gamma \quad \Longleftrightarrow \quad (\forall (x : \alpha) \in \Gamma) \quad \rho(x) \in [\![\alpha]\!]$;

(iii) $\Gamma \models M : \sigma \iff (\forall \rho, \rho \models \Gamma \Rightarrow \rho \models M : \sigma)$.

**Lemma 20.** *Let* $\Gamma \vDash M : \sigma$ *and* $\Delta \vDash M : \tau$*, then*

$$\rho \vDash \Gamma \sqcap \Delta \text{ if and only if } \rho \vDash \Gamma \text{ and } \rho \vDash \Delta.$$

*Proof.* The proof is a straightforward consequence of the Definition 7 of bases intersection $\sqcap$. □

**Proposition 21** (Soundness of $\lambda_{\circledR\cap}$)**.** *If* $\Gamma \vdash M : \sigma$*, then* $\Gamma \models M : \sigma$.

*Proof.* By induction on the derivation of $\Gamma \vdash M : \sigma$.

- If the last rule applied is $(Ax)$, i.e. $x : \sigma \vdash x : \sigma$ the proof is trivial.
- The last rule applied is $(\rightarrow_I)$, i.e.,

  $$\Gamma, x : \alpha \vdash M : \sigma \Rightarrow \Gamma \vdash \lambda x.M : \alpha \rightarrow \sigma.$$

  By the IH $\Gamma, x : \alpha \models M : \sigma$. Suppose that $\rho \models \Gamma$ and we want to show that $\rho \models \lambda x.M : \alpha \rightarrow \sigma$. We have to show that

  $$[\![\lambda x.M]\!]_\rho \in [\![\alpha \rightarrow \sigma]\!] = [\![\alpha]\!] \longrightarrow [\![\sigma]\!] \text{ i.e. } \forall N \in [\![\alpha]\!].\ [\![\lambda x.M]\!]_\rho N \in [\![\sigma]\!].$$

  Suppose that $N \in [\![\alpha]\!]$. We have that $\rho(N/x) \models \Gamma, x : \alpha$ since $\rho \models \Gamma$, $x \notin \Gamma$ and $\rho(N/x)(x) = N \in [\![\alpha]\!]$. By IH $\rho(N/x) \models M : \sigma$, hence we can conclude that $[\![M]\!]_{\rho(N/x)} \in [\![\sigma]\!]$. Using Lemma 18(ii) we get $[\![\lambda x.M]\!]_\rho N \rightarrow [\![M]\!]_{\rho(N/x)}$. Since $[\![M]\!]_{\rho(N/x)} \in [\![\sigma]\!]$ and $[\![\sigma]\!]$ is $\circledR$-saturated, we obtain $[\![\lambda x.M]\!]_\rho N \in [\![\sigma]\!]$.
- The last rule applied is $(\rightarrow_E)$, i.e.

  $$\Gamma \vdash M : \cap_i^n \tau_i \rightarrow \sigma,\ \Delta_0 \vdash N : \tau_0 \ \ldots\ \Delta_n \vdash N : \tau_n \Rightarrow \Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash MN : \sigma.$$

  Let $\rho$ be any valuation.
  Assuming that $\Gamma \vdash M : \cap_i^n \tau_i \rightarrow \sigma, \Delta_1 \vdash N : \tau_1, \ldots, \Delta_n \vdash N : \tau_n$, we have to prove that if $\rho \vDash \Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$, then $\rho \vDash MN : \sigma$.

  By IH, $\Gamma \models M : \cap_i^n \tau_i \rightarrow \sigma$ and $\Delta_0 \models N : \tau_0, \ldots, \Delta_n \models N : \tau_n$. Assume that $\rho \models \Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$. This means that $\rho \models \Gamma$ and $\rho \models \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$. From $\rho \models \Gamma$ we deduce by Definition 19 (iii) $\rho \models M : \cap_i^n \tau_i \rightarrow \sigma$ and by Definition 19 (i) $[\![M]\!]_\rho \in [\![\cap_i^n \tau_i \rightarrow \sigma]\!]$. By Definition 17 $[\![M]\!]_\rho \in \bigcap_i^n [\![\tau]\!] \longrightarrow [\![\sigma]\!]$.
  Using Lemma 20 $\rho \models \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$ implies $(\rho \models \Delta_0^\top) \wedge (\bigwedge_{i=1}^n \rho \models \Delta_i)$, hence by Definition 19 (i) and (iii) we get $([\![N]\!]_\rho \in [\![\top]\!]) \wedge \bigwedge_{i=1}^n ([\![N]\!]_\rho \in [\![\tau_i]\!])$, i.e. $[\![N]\!]_\rho \in \mathcal{SN} \cap \cap_i^n [\![\tau_i]\!] = \cap_i^n [\![\tau_i]\!]$, since $[\![\tau_i]\!] \subseteq \mathcal{SN}$ by Proposition 16(iv). By Definition 12 of $\longrightarrow$, $[\![MN]\!]_\rho = [\![M]\!]_\rho\, [\![N]\!]_\rho \in [\![\sigma]\!]$ and by Definition 19 (i) $\rho \vDash MN : \sigma$.
- The last rule applied is $(Weak)$, i.e.,

  $$\Gamma \vdash M : \sigma \Rightarrow \Gamma, x : \top \vdash x \odot M : \sigma.$$

  By the IH $\Gamma \models M : \sigma$. Suppose that $\rho \models \Gamma, x : \top \Leftrightarrow \rho \models \Gamma$ and $\rho \models x : \top$. From $\rho \models \Gamma$ we obtain $[\![M]\!]_\rho \in [\![\sigma]\!]$. Using multiple times the weakening property $\mathsf{WEAK}$ and Lemma 18(iii) we obtain $Fv(\rho(x)) \odot [\![M]\!]_\rho = [\![x \odot M]\!]_\rho \in [\![\sigma]\!]$, since $Fv(\rho(x)) \cap Fv([\![M]\!]_\rho) = \emptyset$.

- The last rule applied is $(Cont)$, i.e.,

$$\Gamma, x:\alpha, y:\beta \vdash M:\sigma \;\Rightarrow\; \Gamma, z:\alpha\cap\beta \vdash z <^{x}_{y} M:\sigma.$$

By the IH $\Gamma, x:\alpha, y:\beta \models M:\sigma$. Suppose that $\rho \models \Gamma, z:\alpha\cap\beta$.This means that $\rho \models \Gamma$ and $\rho \models z:\alpha\cap\beta \Leftrightarrow \rho(z) \in [\![\alpha]\!]$ and $\rho(z) \in [\![\beta]\!]$. For the sake of simplicity let $\rho(z) \equiv N$. We define a new valuation $\rho'$ such that $\rho' = \rho(N_1/x, N_2/y)$, where $N_1$ and $N_2$ are obtained by renaming the free variables of $N$. Then $\rho' \models \Gamma, x:\alpha, y:\beta$ since $x, y \notin Dom(\Gamma)$, $N_1 \in [\![\alpha]\!]$ and $N_2 \in [\![\beta]\!]$. By the IH $[\![M]\!]_{\rho'} = [\![M]\!]_{\rho(N_1/x,N_2/y)} \in [\![\sigma]\!]$. Using the contraction property CONT we have that $Fv(N) <^{Fv(N_1)}_{Fv(N_2)}$ $[\![M]\!]_{\rho(N_1/x,N_2/y)} = [\![z <^{x}_{y} M]\!]_{\rho} \in [\![\sigma]\!]$.

□

**Theorem 22** ($\mathcal{SN}$ for $\lambda_{\circledR}\cap$)**.** *If $\Gamma \vdash M:\sigma$, then $M$ is strongly normalizing, i.e. $M \in \mathcal{SN}$.*

*Proof.* Suppose $\Gamma \vdash M:\sigma$. By Proposition 21 $\Gamma \models M:\alpha$. According to Definition 19(iii), this means that $(\forall \rho \models \Gamma) \quad \rho \models M:\sigma$. We can choose a particular $\rho_0(x) = x$ for all $x \in \mathtt{var}$. By Proposition 16(iv), $[\![\beta]\!]$ is ®-saturated for each type $\beta$, hence $x = [\![x]\!]_{\rho_0} \in [\![\beta]\!]$ (variable condition for $n = 0$). Therefore, $\rho_0 \models \Gamma$ and we can conclude that $[\![M]\!]_{\rho_0} \in [\![\sigma]\!]$. On the other hand, $M = [\![M]\!]_{\rho_0}$ and $[\![\sigma]\!] \subseteq \mathcal{SN}$ (Proposition 16), hence $M \in \mathcal{SN}$. □

## 1.4 SN ⇒ Typeability in $\lambda_{\circledR}\cap$

We want to prove that if a $\lambda_{\circledR}$-term is SN, then it is typeable in the system $\lambda_{\circledR}\cap$. We proceed in two steps: 1) we show that all $\lambda_{\circledR}$-normal forms are typeable and 2) we prove the head subject expansion. First, let us observe the structure of the $\lambda_{\circledR}$-normal forms, given by the following abstract syntax:

$$\begin{array}{rcl} M_{nf} & ::= & x \,|\, \lambda x.M_{nf} \,|\, \lambda x.x \odot M_{nf} \,|\, xM^1_{nf} \ldots M^n_{nf} \,| \\ & & x <^{x_1}_{x_2} M_{nf}N_{nf}, \text{ with } x_1 \in Fv(M_{nf}), x_2 \in Fv(N_{nf}) \\ W_{nf} & ::= & x \odot M_{nf} \,|\, x \odot W_{nf} \end{array}$$

Notice that it is necessary to distinguish normal forms $W_{nf}$ since the term $\lambda x.y \odot M_{nf}$ is not a normal form, since $\lambda x.y \odot M_{nf} \to_{\omega_1} y \odot \lambda x.M_{nf}$.

**Proposition 23.** *$\lambda_{\circledR}$-normal forms are typeable in the system $\lambda_{\circledR}\cap$.*

*Proof.* By induction on the structure of $M_{nf}$ and $W_{nf}$. □

**Lemma 24** (Inverse substitution lemma)**.** *Let $\Gamma \vdash M[N/x]:\sigma$ and $N$ typeable. Then, there are $\Delta_j$ and $\tau_j$, $j = 0, \ldots, n$ such that $\Delta_j \vdash N:\tau_j$, and $\Gamma', x:\cap^n_i \tau_i \vdash M:\sigma$, where $\Gamma = \Gamma'$, $\Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.*

*Proof.* By induction on the structure of $M$. □

**Proposition 25** (Head subject expansion)**.** *For every $\lambda_{\circledR}$-term $M$: if $M \to M'$, $M$ is a contracted redex and $\Gamma \vdash M':\sigma$, then $\Gamma \vdash M:\sigma$, provided that if $M \equiv (\lambda x.N)P \to_{\beta} N[P/x] \equiv M'$, $P$ is typeable.*

*Proof.* By case study according to the applied reduction. □

**Theorem 26** (SN ⇒ typeability)**.** *All strongly normalising $\lambda_{\circledR}$-terms are typeable in the $\lambda_{\circledR}\cap$ system.*

*Proof.* The proof is by induction on the length of the longest reduction path out of a strongly normalising term $M$, with a subinduction on the size of $M$.

- If $M$ is a normal form, then $M$ is typeable by Proposition 23.
- If $M$ is itself a redex, let $M'$ be the term obtained by contracting the redex $M$. $M'$ is also strongly normalising, hence by IH it is typeable. Then $M$ is typeable, by Proposition 25. Notice that, if $M \equiv (\lambda x.N)P \to_\beta N[P/x] \equiv M'$, then, by IH, $P$ is typeable, since the length of the longest reduction path out of $P$ is smaller than that of $M$, and the size of $P$ is smaller than the size of $M$.
- Next, suppose that $M$ is not itself a redex nor a normal form. Then $M$ is of one of the following forms: $\lambda x.N$, $\lambda x.x \odot N$, $xM_1 \dots M_n$, $x \odot N$, or $x <^{x_1}_{x_2} NP$, $x_1 \in Fv(N)$, $x_2 \in Fv(P)$ (where $M_1, \dots, M_n$, $N$, and $NP$ are *not* normal forms). $M_1, \dots, M_n$ and $NP$ are typeable by IH, as subterms of $M$. Then, it is easy to build the typing for $M$. For instance, let us consider the case $x <^{x_1}_{x_2} NP$ with $x_1 \in Fv(N)$, $x_2 \in Fv(P)$. By induction $NP$ is typeable, hence $N$ is typeable with say $\Gamma, x_1 : \beta \vdash N : \cap_i^n \tau_i \to \sigma$ and $P$ is typeable with say $\Delta_j, x_2 : \gamma_j \vdash P : \tau_j$, for all $j = 0, \dots, n$. Then using the rule $(E \to)$ we obtain $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n, x_1 : \beta, x_2 : \cap_i^n \gamma_i \vdash NP : \sigma$. Finally, the rule $(Cont)$ yields $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n, x : \beta \cap (\cap_i^n \gamma_i) \vdash x <^{x_1}_{x_2} NP : \sigma$.

□

Finally, we can give a characterisation of strong normalisation in $\lambda_{\circledR}$-calculus.

**Theorem 27.** *In $\lambda_{\circledR}$-calculus, the term $M$ is strongly normalising if and only if it is typeable in $\lambda_{\circledR}\cap$.*

*Proof.* Immediate consequence of Theorems 22 and 26. □

# 2 Intersection types for the sequent resource control lambda calculus $\lambda^{\mathsf{Gtz}}_{\circledR}$

In this section we focus on the sequent resource control lambda calculus $\lambda^{\mathsf{Gtz}}_{\circledR}$. First we revisit its syntax and operational semantics; further we introduce an intersection type assignment system and finally we prove that typeability in the proposed system completely characterises the set of strongly normalising $\lambda^{\mathsf{Gtz}}_{\circledR}$-expressions.

## 2.1 Resource control sequent lambda calculus $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$

The *resource control lambda Gentzen* calculus $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ is derived from the $\lambda^{\mathsf{Gtz}}$-calculus (more precisely its confluent sub-calculus $\lambda_{V}^{\mathsf{Gtz}}$) by adding the explicit operators for weakening and contraction. It is proposed in [26]. The abstract syntax of $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ pre-expressions is the following:

$$\begin{array}{lrcl} \text{Pre-values} & F & ::= & x \,|\, \lambda x.f \,|\, x \odot f \,|\, x <_{x_2}^{x_1} f \\ \text{Pre-terms} & f & ::= & F \,|\, fc \\ \text{Pre-contexts} & c & ::= & \widehat{x}.f \,|\, f :: c \,|\, x \odot c \,|\, x <_{x_2}^{x_1} c \end{array}$$

where $x$ ranges over a denumerable set of term variables.

A *pre-value* can be a variable, an abstraction, a weakening or a contraction; a *pre-term* is either a value or a cut (an application). A *pre-context* is one of the following: a selection, a context constructor (usually called cons), a weakening on pre-context or a contraction on a pre-context. Pre-terms and pre-contexts are together referred to as the *pre-expressions* and will be ranged over by $E$. Pre-contexts $x \odot c$ and $x <_{x_2}^{x_1} c$ behave exactly like corresponding pre-terms $x \odot f$ and $x <_{x_2}^{x_1} f$ in the untyped calculus, so they will mostly not be treated separately. The set of free variables of a pre-expression is defined analogously to the free variables in $\lambda_{\textcircled{R}}$-calculus with the following additions:

$$Fv(fc) = Fv(f) \cup Fv(c); \quad Fv(\widehat{x}.f) = Fv(f) \setminus \{x\}; \quad Fv(f :: c) = Fv(f) \cup Fv(c).$$

Like in $\lambda_{\textcircled{R}}$-calculus, the set of $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$-expressions (namely values, terms and contexts), denoted by $\Lambda_{\textcircled{R}}^{\mathsf{Gtz}} \cup \Lambda_{\textcircled{R},C}^{\mathsf{Gtz}}$, is a subset of the set of pre-expressions, defined in Figure 6. Values are denoted by $T$, terms by $t, u, v...$, contexts by $k, k', ...$ and expressions by $e, e'$.

The computation over the set of $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$-expressions reflects the cut-elimination process. Four groups of reductions in $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$-calculus are given in Figure 7.

The first group consists of $\beta_g$, $\pi$, $\sigma$ and $\mu$ reductions from the $\lambda^{\mathsf{Gtz}}$. New reductions are added to deal with explicit contraction ($\gamma$ reductions) and weakening ($\omega$ reductions). The groups of $\gamma$ and $\omega$ reductions consist of rules that perform propagation of contraction into the expression and extraction of weakening out of the expression. This discipline allows us to optimize the computation by delaying the duplication of terms on the one hand, and by performing the erasure of terms as soon as possible on the other. The equivalencies in $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$ are the ones given in Figure 4, except for the fact that they refer to $\lambda_{\textcircled{R}}^{\mathsf{Gtz}}$-expressions.

The meta-substitution $t[u/x]$ is defined as in Figure 3 with the following additions:

$$\begin{array}{rclcrcl} (tk)[u/x] & = & t[u/x]k, \ x \notin Fv(k) & \qquad & (tk)[u/x] & = & tk[u/x], \ x \notin Fv(t) \\ (\widehat{y}.t)[u/x] & = & \widehat{y}.t[u/x] & & & & \\ (t :: k)[u/x] & = & t[u/x] :: k, \ x \notin Fv(k) & & (t :: k)[u/x] & = & t :: k[u/x], \ x \notin Fv(t) \end{array}$$

In the $\pi$ rule, the meta-operator @, called *append*, joins two contexts and is defined as:

$$\begin{array}{rclcrcl} (\widehat{x}.t)@k' & = & \widehat{x}.tk' & \qquad & (u :: k)@k' & = & u :: (k@k') \\ (x \odot k)@k' & = & x \odot (k@k') & & (x <_z^y k)@k' & = & x <_z^y (k@k'). \end{array}$$

$$\frac{}{x \in \Lambda^{\mathsf{Gtz}}_{\circledR}} \qquad \frac{f \in \Lambda^{\mathsf{Gtz}}_{\circledR} \;\; x \in Fv(f)}{\lambda x.f \in \Lambda^{\mathsf{Gtz}}_{\circledR}}$$

$$\frac{f \in \Lambda^{\mathsf{Gtz}}_{\circledR} \;\; c \in \Lambda^{\mathsf{Gtz}}_{\circledR,C} \;\; Fv(f) \cap Fv(c) = \emptyset}{fc \in \Lambda^{\mathsf{Gtz}}_{\circledR}}$$

$$\frac{f \in \Lambda^{\mathsf{Gtz}}_{\circledR} \;\; x \in Fv(f)}{\widehat{x}.f \in \Lambda^{\mathsf{Gtz}}_{\circledR,C}} \qquad \frac{f \in \Lambda^{\mathsf{Gtz}}_{\circledR} \;\; c \in \Lambda^{\mathsf{Gtz}}_{\circledR,C} \;\; Fv(f) \cap Fv(c) = \emptyset}{f :: c \in \Lambda^{\mathsf{Gtz}}_{\circledR,C}}$$

$$\frac{E \in \Lambda^{\mathsf{Gtz}}_{\circledR} \cup \Lambda^{\mathsf{Gtz}}_{\circledR,C} \;\; x \notin Fv(E)}{x \odot E \in \Lambda^{\mathsf{Gtz}}_{\circledR} \cup \Lambda^{\mathsf{Gtz}}_{\circledR,C}}$$

$$\frac{E \in \Lambda^{\mathsf{Gtz}}_{\circledR} \cup \Lambda^{\mathsf{Gtz}}_{\circledR,C} \;\; x_1 \neq x_2 \;\; x_1, x_2 \in Fv(E) \;\; x \notin Fv(E) \setminus \{x_1, x_2\}}{x <^{x_1}_{x_2} E \in \Lambda^{\mathsf{Gtz}}_{\circledR} \cup \Lambda^{\mathsf{Gtz}}_{\circledR,C}}$$

Figure 6: $\Lambda^{\mathsf{Gtz}}_{\circledR} \cup \Lambda^{\mathsf{Gtz}}_{\circledR,C}$: $\lambda^{\mathsf{Gtz}}_{\circledR}$-expressions

## 2.2 Intersection types for $\lambda^{\mathsf{Gtz}}_{\circledR}$

The type assignment system $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$ that assigns strict types to $\lambda^{\mathsf{Gtz}}_{\circledR}$-expressions is given in Figure 8. Due to the sequent flavour of the $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculus, here we distinguish two sorts of type assignments:

- $\Gamma \vdash t : \sigma$ for typing a term and
- $\Gamma; \beta \vdash k : \sigma$, a type assignment with a *stoup*, for typing a context.

A stoup is a place for the last formula in the antecedent, after the semi-colon. The formula in the stoup is the place where computation will continue.

The syntax of types and the related definitions are the same as in $\lambda_{\circledR}\cap$. The $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$ system is also syntax-directed i.e. the intersection is incorporated into already existing rules of the simply-typed system. In the style of sequent calculus, left intersection introduction is managed by the contraction rules $(Cont_t)$ and $(Cont_k)$, whereas the right intersection introduction is performed by the cut rule $(Cut)$ and left arrow introduction rule $(\to_L)$. In these two rules $Dom(\Gamma_1) = \ldots = Dom(\Gamma_n)$. The role of $\Gamma_0^{\top}$ has been already explained in subsection 1.2.

The Generation lemma induced by the proposed system is the following:

**Lemma 28** (Generation lemma for $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$)**.**

*(i)* $\Gamma \vdash \lambda x.t : \tau$ *iff there exist* $\alpha$ *and* $\sigma$ *such that* $\tau \equiv \alpha \to \sigma$ *and* $\Gamma, x : \alpha \vdash t : \sigma$.

*(ii)* $\Gamma; \gamma \vdash t :: k : \rho$ *iff* $\Gamma = {\Gamma'_0}^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n, \Delta$, $\gamma \equiv \cap^m_j(\cap^n_i \sigma_i \to \tau_j)$, $\Delta; \cap^m_j \tau_j \vdash k : \rho$ *and* $\Gamma'_l \vdash t : \sigma_l$ *for all* $l \in \{0, \ldots, n\}$.

$$
\begin{array}{lrcl}
(\beta_g) & (\lambda x.t)(u :: k) & \to & u(\widehat{x}.tk) \\
(\sigma) & T(\widehat{x}.v) & \to & v[T/x] \\
(\pi) & (tk)k' & \to & t(k@k') \\
(\mu) & \widehat{x}.xk & \to & k \\
\\
(\gamma_1) & x <^{x_1}_{x_2} (\lambda y.t) & \to & \lambda y.x <^{x_1}_{x_2} t \\
(\gamma_2) & x <^{x_1}_{x_2} (tk) & \to & (x <^{x_1}_{x_2} t)k, \quad \text{if } x_1, x_2 \notin Fv(k) \\
(\gamma_3) & x <^{x_1}_{x_2} (tk) & \to & t(x <^{x_1}_{x_2} k), \quad \text{if } x_1, x_2 \notin Fv(t) \\
(\gamma_4) & x <^{x_1}_{x_2} (\widehat{y}.t) & \to & \widehat{y}.(x <^{x_1}_{x_2} t) \\
(\gamma_5) & x <^{x_1}_{x_2} (t :: k) & \to & (x <^{x_1}_{x_2} t) :: k, \quad \text{if } x_1, x_2 \notin Fv(k) \\
(\gamma_6) & x <^{x_1}_{x_2} (t :: k) & \to & t :: (x <^{x_1}_{x_2} k), \quad \text{if } x_1, x_2 \notin Fv(t) \\
\\
(\omega_1) & \lambda x.(y \odot t) & \to & y \odot (\lambda x.t), \quad x \neq y \\
(\omega_2) & (x \odot t)k & \to & x \odot (tk) \\
(\omega_3) & t(x \odot k) & \to & x \odot (tk) \\
(\omega_4) & \widehat{x}.(y \odot t) & \to & y \odot (\widehat{x}.t), \quad x \neq y \\
(\omega_5) & (x \odot t) :: k & \to & x \odot (t :: k) \\
(\omega_6) & t :: (x \odot k) & \to & x \odot (t :: k) \\
\\
(\gamma\omega_1) & x <^{x_1}_{x_2} (y \odot e) & \to & y \odot (x <^{x_1}_{x_2} e) \qquad x_1 \neq y \neq x_2 \\
(\gamma\omega_2) & x <^{x_1}_{x_2} (x_1 \odot e) & \to & e[x/x_2]
\end{array}
$$

Figure 7: Reduction rules of $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculus

*(iii)* $\Gamma \vdash tk : \sigma$ *iff* $\Gamma = {\Gamma'_0}^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n, \Delta$, *and there exist* $\tau_j, j = 0, \ldots, n$ *such that for all* $j \in \{0, \ldots, n\}$ *the following holds:* $\Gamma'_j \vdash t : \tau_j$, *and* $\Delta; \cap_i^n \tau_i \vdash k : \sigma$.

*(iv)* $\Gamma; \alpha \vdash \widehat{x}.t : \sigma$ *iff* $\Gamma, x : \alpha \vdash t : \sigma$.

*(v)* $\Gamma \vdash z <^x_y t : \sigma$ *iff there exist* $\Gamma', \alpha, \beta$ *such that* $\Gamma = \Gamma', z : \alpha \cap \beta$ *and* $\Gamma', x : \alpha, y : \beta \vdash t : \sigma$.

*(vi)* $\Gamma \vdash x \odot t : \sigma$ *iff* $\Gamma = \Gamma', x : \top$ *and* $\Gamma' \vdash t : \sigma$.

*(vii)* $\Gamma; \gamma \vdash z <^x_y k : \sigma$ *iff there exist* $\Gamma', \alpha, \beta$ *such that* $\Gamma = \Gamma', z : \alpha \cap \beta$ *and* $\Gamma', x : \alpha, y : \beta; \gamma \vdash k : \sigma$.

*(viii)* $\Gamma; \gamma \vdash x \odot k : \sigma$ *iff* $\Gamma = \Gamma', x : \top$ *and* $\Gamma'; \gamma \vdash k : \sigma$.

The proposed system satisfies the following properties.

**Lemma 29.**

*(i) If* $\Gamma \vdash t : \sigma$, *then* $Dom(\Gamma) = Fv(t)$.

*(ii) If* $\Gamma; \alpha \vdash k : \sigma$, *then* $Dom(\Gamma) = Fv(k)$.

*Proof.* Similar to the proof of Lemma 8. □

**Lemma 30** (Substitution lemma for $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$)**.**

$$\frac{}{x:\sigma \vdash x:\sigma}\ (Ax)$$

$$\frac{\Gamma, x:\alpha \vdash t:\sigma}{\Gamma \vdash \lambda x.t : \alpha \to \sigma}\ (\to_R) \qquad \frac{\Gamma, x:\alpha \vdash t:\sigma}{\Gamma;\alpha \vdash \widehat{x}.t : \sigma}\ (Sel)$$

$$\frac{\Gamma_0 \vdash t:\sigma_0 \quad \ldots \quad \Gamma_n \vdash t:\sigma_n \quad \Delta;\cap_j^m \tau_j \vdash k:\rho}{\Gamma_0^\top \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n, \Delta;\cap_j^m(\cap_i^n \sigma_i \to \tau_j) \vdash t :: k : \rho}\ (\to_L)$$

$$\frac{\Gamma_0 \vdash t:\sigma_0 \quad \ldots \quad \Gamma_n \vdash t:\sigma_n \quad \Delta;\cap_i^n \sigma_i \vdash k:\tau}{\Gamma_0^\top \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n, \Delta \vdash tk : \tau}\ (Cut)$$

$$\frac{\Gamma, x:\alpha, y:\beta \vdash t:\sigma}{\Gamma, z:\alpha\cap\beta \vdash z <_y^x t:\sigma}\ (Cont_t) \qquad \frac{\Gamma \vdash t:\sigma}{\Gamma, x:\top \vdash x \odot t:\sigma}\ (Weak_t)$$

$$\frac{\Gamma, x:\alpha, y:\beta;\gamma \vdash k:\sigma}{\Gamma, z:\alpha\cap\beta;\gamma \vdash z <_y^x k:\sigma}\ (Cont_k) \qquad \frac{\Gamma;\gamma \vdash k:\sigma}{\Gamma, x:\top;\gamma \vdash x \odot k:\sigma}\ (Weak_k)$$

Figure 8: $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$: $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculus with intersection types

*(i)* *If* $\Gamma, x:\cap_i^n \tau_i \vdash t:\sigma$ *and for all* $j=0,\ldots,n$, $\Delta_j \vdash u:\tau_j$, *then* $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash t[u/x]:\sigma$.

*(ii)* *If* $\Gamma, x:\cap_i^n \tau_i;\alpha \vdash k:\sigma$ *and for all* $j=0,\ldots,n$, $\Delta_j \vdash u:\tau_j$, *then* $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n;\alpha \vdash k[u/x]:\sigma$.

*Proof.* By mutual induction on the structure of terms and contexts. □

**Proposition 31** (Append lemma)**.** *If* $\Gamma_j;\alpha \vdash k:\tau_j$ *for all* $j=0,\ldots,n$, *and* $\Delta;\cap_i^n \tau_i \vdash k':\sigma$, *then* $\Gamma_0^\top \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n, \Delta;\alpha \vdash k@k':\sigma$.

*Proof.* By induction on the structure of the context $k$. □

**Proposition 32** (Subject equivalence for $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$)**.**

*(i)* *For every* $\lambda^{\mathsf{Gtz}}_{\circledR}$*-term* $t$*: if* $\Gamma \vdash t:\sigma$ *and* $t \equiv_{\lambda^{\mathsf{Gtz}}_{\circledR}} t'$, *then* $\Gamma \vdash t':\sigma$.

*(ii)* *For every* $\lambda^{\mathsf{Gtz}}_{\circledR}$*-context* $k$*: if* $\Gamma;\alpha \vdash k:\sigma$ *and* $k \equiv_{\lambda^{\mathsf{Gtz}}_{\circledR}} k'$, *then* $\Gamma;\alpha \vdash k':\sigma$.

*Proof.* By case analysis on the applied equivalence. □

**Proposition 33** (Subject reduction for $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$)**.**

*(i)* *For every* $\lambda^{\mathsf{Gtz}}_{\circledR}$*-term* $t$*: if* $\Gamma \vdash t:\sigma$ *and* $t \to t'$, *then* $\Gamma \vdash t':\sigma$.

*(ii)* *For every* $\lambda^{\mathsf{Gtz}}_{\circledR}$*-context* $k$*: if* $\Gamma;\alpha \vdash k:\sigma$ *and* $k \to k'$, *then* $\Gamma;\alpha \vdash k':\sigma$.

*Proof.* By case analysis on the applied reduction, using Lemmas 30 and 31 for the cases of $(\sigma)$ and $(\pi)$ rule, respectively. □

## 2.3 Typeability ⇒ SN in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$

In this section, we prove the strong normalisation of the $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculus with intersection types. The termination is proved by showing that the reduction on the set $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}} \cup \Lambda^{\mathsf{Gtz}}_{\mathcal{R},C}$ of the typeable $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions is included in a particular well-founded relation, which we define as the lexicographic product of three well-founded component relations. The first one is based on the mapping of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions into $\lambda_{\mathcal{R}}$-terms. We show that this mapping preserves types and that every $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reduction can be simulated either by a $\lambda_{\mathcal{R}}$-reduction or by an equality and each $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-equivalence can be simulated by an $\lambda_{\mathcal{R}}$-equivalence. The other two well-founded orders are based on the introduction of quantities designed to decrease a global measure associated with specific $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions during the computation.

**Definition 34.** The mapping $\lfloor\ \rfloor : \Lambda^{\mathsf{Gtz}}_{\mathcal{R}} \to \Lambda_{\mathcal{R}}$ is defined together with the auxiliary mapping $\lfloor\ \rfloor_{\mathsf{k}} : \Lambda^{\mathsf{Gtz}}_{\mathcal{R},C} \to (\Lambda_{\mathcal{R}} \to \Lambda_{\mathcal{R}})$ in the following way:

$$\begin{array}{lcllcl}
\lfloor x\rfloor & = & x & \lfloor \widehat{x}.t\rfloor_{\mathsf{k}}(M) & = & (\lambda x.\lfloor t\rfloor)M \\
\lfloor \lambda x.t\rfloor & = & \lambda x.\lfloor t\rfloor & \lfloor t :: k\rfloor_{\mathsf{k}}(M) & = & \lfloor k\rfloor_{\mathsf{k}}(M\lfloor t\rfloor) \\
\lfloor x \odot t\rfloor & = & x \odot \lfloor t\rfloor & \lfloor x \odot k\rfloor_{\mathsf{k}}(M) & = & x \odot \lfloor k\rfloor_{\mathsf{k}}(M) \\
\lfloor x <^y_z t\rfloor & = & x <^y_z \lfloor t\rfloor & \lfloor x <^y_z k\rfloor_{\mathsf{k}}(M) & = & x <^y_z \lfloor k\rfloor_{\mathsf{k}}(M) \\
\lfloor tk\rfloor & = & \lfloor k\rfloor_{\mathsf{k}}(\lfloor t\rfloor) & & &
\end{array}$$

**Lemma 35.**

*(i)* $Fv(t) = Fv(\lfloor t\rfloor)$, *for* $t \in \Lambda^{\mathsf{Gtz}}_{\mathcal{R}}$.

*(ii)* $\lfloor v[t/x]\rfloor = \lfloor v\rfloor[\lfloor t\rfloor/x]$, *for* $v,t \in \Lambda^{\mathsf{Gtz}}_{\mathcal{R}}$.

We prove that the mappings $\lfloor\ \rfloor$ and $\lfloor\ \rfloor_{\mathsf{k}}$ preserve types. In the sequel, the notation $\Lambda_{\mathcal{R}(\Gamma\vdash_{\lambda_{\mathcal{R}}}\sigma)}$ stands for $\{M \mid M \in \Lambda_{\mathcal{R}} \ \&\ \Gamma \vdash_{\lambda_{\mathcal{R}}} M : \sigma\}$.

**Proposition 36** (Type preservation by $\lfloor\ \rfloor$)**.**

*(i) If* $\Gamma \vdash t : \sigma$, *then* $\Gamma \vdash_{\lambda_{\mathcal{R}}} \lfloor t\rfloor : \sigma$.

*(ii) If* $\Gamma; \cap_i^n \tau_i \vdash k : \sigma$, *then* $\lfloor k\rfloor_{\mathsf{k}} : \Lambda_{\mathcal{R}(\Delta_j\vdash_{\lambda_{\mathcal{R}}}\tau_j)} \to \Lambda_{\mathcal{R}(\Gamma,\Delta\vdash_{\lambda_{\mathcal{R}}}\sigma)}$, *for all* $j \in \{0,\ldots,n\}$ *and for some* $\Delta = \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.

*Proof.* The proposition is proved by simultaneous induction on derivations. We distinguish cases according to the last typing rule used.

- Cases $(Ax)$, $(\to_R)$, $(Weak_t)$ and $(Cont_t)$ are easy, because the intersection type assignment system of $\lambda_{\mathcal{R}}$ has exactly the same rules.

- Case $(Sel)$: the derivation ends with the rule

$$\frac{\Gamma, x:\alpha \vdash t:\sigma}{\Gamma;\alpha \vdash \widehat{x}.t:\sigma}\ (Sel)$$

By IH we have that $\Gamma, x:\alpha \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \sigma$, where $\alpha = \cap_i^n \tau_i$. For any $M \in \Lambda_{\circledR}$ such that $\Delta_j \vdash_{\lambda_{\circledR}} M : \tau_i$, for all $j \in \{0,\ldots,n\}$, we have

$$\frac{\dfrac{\Gamma, x:\cap_i^n \tau_i \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \sigma}{\Gamma \vdash_{\lambda_{\circledR}} \lambda x.\lfloor t \rfloor : \cap_i^n \tau_i \to \sigma}\ (\to_I) \quad \Delta_0 \vdash_{\lambda_{\circledR}} M:\tau_0 \ \ldots\ \Delta_n \vdash_{\lambda_{\circledR}} M:\tau_n}{\Gamma, \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash_{\lambda_{\circledR}} (\lambda x.\lfloor t \rfloor)M : \sigma}\ (\to_E)$$

Since $(\lambda x.\lfloor t \rfloor)M = \lfloor \widehat{x}.t \rfloor_{\mathsf{k}}(M)$, we conclude that $\lfloor \widehat{x}.t \rfloor_{\mathsf{k}} : \Lambda_{\circledR(\Delta_j \vdash_{\lambda_{\circledR}} \tau_j)} \to \Lambda_{\circledR(\Gamma, \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n \vdash_{\lambda_{\circledR}} \sigma)}$.

- Case $(\to_L)$: the derivation ends with the rule

$$\frac{\Gamma_0 \vdash t:\sigma_0 \ \ldots\ \Gamma_n \vdash t:\sigma_n \quad \Delta;\cap_j^m \tau_j \vdash k:\rho}{\Gamma, \Delta; \cap_j^m (\cap_i^n \sigma_i \to \tau_j) \vdash t :: k : \rho}\ (\to_L)$$

for $\Gamma = \Gamma_0^{\top} \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n$. By IH we have that $\Gamma_l \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \sigma_l$, for $l \in \{0,\ldots,n\}$. For any $M \in \Lambda_{\circledR}$ such that $\Gamma'_j \vdash_{\Lambda_{\circledR}} M : \cap_i^n \sigma_i \to \tau_j$, $j = 1,\ldots,m$ we have

$$\frac{\Gamma'_j \vdash_{\lambda_{\circledR}} M : \cap_i^n \sigma_i \to \tau_j \quad \Gamma_0 \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \sigma_0 \ \ldots\ \Gamma_n \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \sigma_n}{\Gamma_0^{\top} \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n, \Gamma'_j \vdash_{\lambda_{\circledR}} M \lfloor t \rfloor : \tau_j}\ (\to_E)$$

From the right-hand side premise in the $(\to_L)$ rule, by IH, we get that $\lfloor k \rfloor_{\mathsf{k}}$ is the function with the scope $\lfloor k \rfloor_{\mathsf{k}} : \Lambda_{\circledR(\Gamma'''_j \vdash_{\lambda_{\circledR}} \tau_j)} \to \Lambda_{\circledR(\Gamma''', \Gamma'' \vdash_{\lambda_{\circledR}} \rho)}$, for some $\Gamma''' = \Gamma_0'''^{\top} \sqcap \Gamma_1''' \sqcap \ldots \sqcap \Gamma_n'''$. For $\Gamma''' \equiv \Gamma, \Gamma'$ and by taking $M\lfloor t \rfloor$ as the argument of the function $\lfloor k \rfloor_{\mathsf{k}}$, we get $\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\circledR}} \lfloor k \rfloor_{\mathsf{k}}(M\lfloor t \rfloor) : \rho$. Since $\lfloor k \rfloor_{\mathsf{k}}(M\lfloor t \rfloor) = \lfloor t :: k \rfloor_{\mathsf{k}}(M)$, we have that $\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\circledR}} \lfloor t :: k \rfloor_{\mathsf{k}}(M) : \rho$. This holds for any $M$ of the appropriate type, yielding
$\lfloor t :: k \rfloor_{\mathsf{k}} : \Lambda_{\circledR(\Gamma' \vdash_{\lambda_{\circledR}} \cap_i^n \sigma_i \to \tau_j)} \to \Lambda_{\circledR(\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\circledR}} \rho)}$, which is exactly what we need. Case $(Cut)$: the derivation ends with the rule

$$\frac{\Gamma_0 \vdash t:\tau_0 \ldots \Gamma_n \vdash t:\tau_n \quad \Delta;\cap \tau_i^n \vdash k:\sigma}{\Gamma_0^{\top} \sqcap \Gamma_1 \sqcap \ldots \sqcap \Gamma_n, \Delta \vdash tk : \sigma}\ (Cut)$$

By IH we have that $\Gamma_j \vdash_{\lambda_{\circledR}} \lfloor t \rfloor : \tau_j$ and $\lfloor k \rfloor_{\mathsf{k}} : \Lambda_{\circledR(\Gamma'_j \vdash_{\lambda_{\circledR}} \tau_j)} \to \Lambda_{\circledR(\Gamma', \Delta \vdash_{\lambda_{\circledR}} \sigma)}$ for all $j = 0,\ldots,n$ and for $\Gamma' = \Gamma_0^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n$. Hence, for any $M \in \Lambda_{\circledR}$ such that $\Gamma'_j \vdash_{\lambda_{\circledR}} M : \tau_j$, $\Gamma', \Delta \vdash_{\lambda_{\circledR}} \lfloor k \rfloor_{\mathsf{k}}(M) : \sigma$ holds. By taking $M \equiv \lfloor t \rfloor$ and $\Gamma' \equiv \Gamma$, we get $\Gamma, \Delta \vdash_{\lambda_{\circledR}} \lfloor k \rfloor_{\mathsf{k}}(\lfloor t \rfloor) : \sigma$. But $\lfloor k \rfloor_{\mathsf{k}}(\lfloor t \rfloor) = \lfloor tk \rfloor$, so the proof is done.

- Case $(Weak_k)$: the derivation ends with the rule

$$\frac{\Gamma;\beta \vdash k : \sigma}{\Gamma, x : \top;\beta \vdash x \odot k : \sigma}\ (Weak_k)$$

By IH we have that $\lfloor k \rfloor_{\mathsf{k}}$ is the function with the scope $\lfloor k \rfloor_{\mathsf{k}} : \Lambda_{\mathbb{R}(\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} \tau_j)} \to \Lambda_{\mathbb{R}(\Gamma,\Gamma_0'^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n \vdash_{\lambda_{\mathbb{R}}} \sigma)}$, meaning that for each $M \in \Lambda_{\mathbb{R}}$ such that $\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} M : \tau_j$ for all $j \in \{0,\ldots,n\}$ holds $\Gamma_0'^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n, \Gamma \vdash_{\lambda_{\mathbb{R}}} \lfloor k \rfloor_{\mathsf{k}}(M) : \sigma$. Now, we can apply $(Weak)$ rule:

$$\frac{\Gamma, \Gamma_0'^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n \vdash \lfloor k \rfloor_{\mathsf{k}}(M) : \sigma}{\Gamma, \Gamma_0'^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n, x : \top \vdash x \odot \lfloor k \rfloor_{\mathsf{k}}(M) : \sigma}\ (Weak)$$

Since $x \odot \lfloor k \rfloor_{\mathsf{k}}(M) = \lfloor x \odot k \rfloor_{\mathsf{k}}(M)$, this means that $\lfloor x \odot k \rfloor_{\mathsf{k}} : \Lambda_{\mathbb{R}(\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} \tau_j)} \to \Lambda_{\mathbb{R}(\Gamma,\Gamma_0'^{\top} \sqcap \Gamma'_1 \sqcap \ldots \sqcap \Gamma'_n, x:\top \vdash_{\lambda_{\mathbb{R}}} \sigma)}$, which is exactly what we wanted to get.

- Case $(Cont_k)$: similar to the case $(Weak_k)$, relying on the rule $(Cont)$ in $\lambda_{\mathbb{R}}$.

□

For the given encoding $\lfloor\ \rfloor$, we show that each $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reduction step can be simulated by an $\lambda_{\mathbb{R}}$-reduction or by an equality. In order to do so, we prove the following lemmas. The proofs of Lemma 38 and Lemma 39, according to [21], use Regnier's $\sigma$ reductions, investigated in [51].

$$\begin{array}{rcl} ((\lambda x.M)N)P & \to & (\lambda x.(MN))P \ \ x \notin P \\ (\lambda xy.M)N & \to & \lambda y.((\lambda x.M)N) \ \ y \notin N \\ M((\lambda x.P)N) & \to & (\lambda x.MP)N \ \ x \notin M \end{array}$$

**Lemma 37.** *If $M \to_{\lambda_{\mathbb{R}}} M'$, then $\lfloor k \rfloor_{\mathsf{k}}(M) \to_{\lambda_{\mathbb{R}}} \lfloor k \rfloor_{\mathsf{k}}(M')$.*

**Lemma 38.** $\lfloor k \rfloor_{\mathsf{k}}((\lambda x.P)N) \to_{\lambda_{\mathbb{R}}} (\lambda x.\lfloor k \rfloor_{\mathsf{k}}(P))N$.

**Lemma 39.** *If $M \in \Lambda^{\mathbb{R}}$ and $k,k' \in \Lambda^{\mathsf{Gtz}}_{\mathbb{R},C}$, then $\lfloor k' \rfloor_{\mathsf{k}} \circ \lfloor k \rfloor_{\mathsf{k}}(M) \to_{\lambda_{\mathbb{R}}} \lfloor k@k' \rfloor_{\mathsf{k}}(M)$.*

**Lemma 40.**

*(i) If $x \notin Fv(k)$, then $(\lfloor k \rfloor_{\mathsf{k}}(M))[N/x] = \lfloor k \rfloor_{\mathsf{k}}(M[N/x])$.*

*(ii) If $x,y \notin Fv(k)$, then $z <^x_y (\lfloor k \rfloor_{\mathsf{k}}(M)) \to_{\lambda_{\mathbb{R}}} \lfloor k \rfloor_{\mathsf{k}}(z <^x_y M)$.*

*(iii) $\lfloor k \rfloor_{\mathsf{k}}(x \odot M) \to_{\lambda_{\mathbb{R}}} x \odot \lfloor k \rfloor_{\mathsf{k}}(M)$.*

Now we can prove that the reduction rules of $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$ can be simulated by the reduction rules or an equality in the $\lambda_{\mathbb{R}}$-calculus. Moreover, the equivalences of $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-calculus are preserved in $\lambda_{\mathbb{R}}$-calculus.

**Theorem 41** (Simulation of $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reduction by $\lambda_{\mathbb{R}}$-reduction)**.**

*(i) If a term $t \to_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} t'$, then $\lfloor t \rfloor \to_{\lambda_{\mathbb{R}}} \lfloor t' \rfloor$.*

*(ii)* *If a context* $k \to_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} k'$ *by* $\gamma_6$ *or* $\omega_6$ *reduction, then* $\lfloor k \rfloor_{\mathsf{k}}(M) \equiv \lfloor k' \rfloor_{\mathsf{k}}(M)$*, for any* $M \in \Lambda^{\mathbb{R}}$.

*(iii)* *If a context* $k \to_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} k'$ *by some other reduction, then* $\lfloor k \rfloor_{\mathsf{k}}(M) \to_{\lambda_{\mathbb{R}}} \lfloor k' \rfloor_{\mathsf{k}}(M)$*, for any* $M \in \Lambda^{\mathbb{R}}$.

*(iv)* *If* $t \equiv_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} t'$*, then* $\lfloor t \rfloor \equiv_{\lambda_{\mathbb{R}}} \lfloor t' \rfloor$*, and if* $k \equiv_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} k'$*, then* $\lfloor k \rfloor_{\mathsf{k}}(M) \equiv_{\lambda_{\mathbb{R}}} \lfloor k' \rfloor_{\mathsf{k}}(M)$*, for any* $M \in \Lambda_{\mathbb{R}}$.

*Proof.* The proof goes by case analysis on the outermost reduction or equivalence performed, using Definition 34. □

The previous proposition shows that $\beta_{\mathsf{g}}$, $\pi$, $\sigma$, $\mu$, $\gamma_1$ - $\gamma_5$, $\omega_1$ - $\omega_5$, $\gamma\omega_1$ and $\gamma\omega_2$ $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reductions are interpreted by $\lambda_{\mathbb{R}}$-reductions and that $\gamma_6$ and $\omega_6$ $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reductions are interpreted by an identity in the $\lambda_{\mathbb{R}}$. Since the set of equivalences of the two calculi coincide, they are trivially preserved. If one wants to prove that there is no infinite sequence of $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reductions one has to prove that there cannot exist an infinite sequence of $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reductions which are all interpreted as equalities. To prove this, one shows that if a term is reduced with such a $\lambda^{\mathsf{Gtz}}_{\mathbb{R}}$-reduction, it is reduced for another order that forbids infinite decreasing chains. This order is itself composed of several orders, free of infinite decreasing chains (Definition 45).

**Definition 42.** The functions $\mathcal{S}(\,)$, $\|\ \|_{\mathsf{c}}$, $\|\ \|_{\mathsf{w}} : \Lambda^{\mathsf{Gtz}}_{\mathbb{R}} \to \mathbb{N}$ are defined as follows:

$$\begin{array}{rclrcl}
\mathcal{S}(x) & = & 1 & \mathcal{S}(tk) & = & \mathcal{S}(t)+\mathcal{S}(k) \\
\mathcal{S}(\lambda x.t) & = & 1+\mathcal{S}(t) & \mathcal{S}(\widehat{x}.t) & = & 1+\mathcal{S}(t) \\
\mathcal{S}(x \odot e) & = & 1+\mathcal{S}(e) & \mathcal{S}(t :: k) & = & \mathcal{S}(t)+\mathcal{S}(k) \\
\mathcal{S}(x <^{y}_{z} e) & = & 1+\mathcal{S}(e) & & &
\end{array}$$

$$\begin{array}{rclrcl}
\|x\|_{\mathsf{c}} & = & 0 & \|x\|_{\mathsf{w}} & = & 1 \\
\|\lambda x.t\|_{\mathsf{c}} & = & \|t\|_{\mathsf{c}} & \|\lambda x.t\|_{\mathsf{w}} & = & 1+\|t\|_{\mathsf{w}} \\
\|x \odot e\|_{\mathsf{c}} & = & \|e\|_{\mathsf{c}} & \|x \odot e\|_{\mathsf{w}} & = & 0 \\
\|x <^{y}_{z} e\|_{\mathsf{c}} & = & \|e\|_{\mathsf{c}}+\mathcal{S}(e) & \|x <^{y}_{z} e\|_{\mathsf{w}} & = & 1+\|e\|_{\mathsf{w}} \\
\|tk\|_{\mathsf{c}} & = & \|t\|_{\mathsf{c}}+\|k\|_{\mathsf{c}} & \|tk\|_{\mathsf{w}} & = & 1+\|t\|_{\mathsf{w}}+\|k\|_{\mathsf{w}} \\
\|\widehat{x}.t\|_{\mathsf{c}} & = & \|t\|_{\mathsf{c}} & \|\widehat{x}.t\|_{\mathsf{w}} & = & 1+\|t\|_{\mathsf{w}} \\
\|t :: k\|_{\mathsf{c}} & = & \|t\|_{\mathsf{c}}+\|k\|_{\mathsf{c}} & \|t :: k\|_{\mathsf{w}} & = & 1+\|t\|_{\mathsf{w}}+\|k\|_{\mathsf{w}}
\end{array}$$

**Lemma 43.** *For all* $e, e' \in \Lambda^{\mathsf{Gtz}}_{\mathbb{R}}$*:*

*(i)* *If* $e \to_{\gamma_6} e'$*, then* $\|e\|_{\mathsf{c}} > \|e'\|_{\mathsf{c}}$.

*(ii)* *If* $e \to_{\omega_6} e'$*, then* $\|e\|_{\mathsf{c}} = \|e'\|_{\mathsf{c}}$.

*(iii)* *If* $e \equiv_{\lambda^{\mathsf{Gtz}}_{\mathbb{R}}} e'$*, then* $\|e\|_{\mathsf{c}} = \|e'\|_{\mathsf{c}}$.

**Lemma 44.**

*(i)* *For all* $e, e' \in \Lambda^{\mathsf{Gtz}}_{\mathbb{R}}$*: If* $e \to_{\omega_6} e'$*, then* $\|e\|_{\mathsf{w}} > \|e'\|_{\mathsf{w}}$.

*(ii) If* $e \equiv_{\lambda^{\mathsf{Gtz}}_{\circledR}} e'$, *then* $\|e\|_{\mathsf{w}} = \|e'\|_{\mathsf{w}}$.

Now we can define the following orders based on the previously introduced mapping and norms.

**Definition 45.** We define the following strict orders and equivalencies on $\Lambda^{\mathsf{Gtz}}_{\circledR}\cap$:

(i) $t >_{\lambda_{\circledR}} t'$ iff $\lfloor t\rfloor \to^{+}_{\lambda_{\circledR}} \lfloor t'\rfloor$; $t =_{\lambda_{\circledR}} t'$ iff $\lfloor t\rfloor \equiv_{\lambda_{\circledR}} \lfloor t'\rfloor$
$k >_{\lambda_{\circledR}} k'$ iff $\lfloor k\rfloor_{\mathsf{k}}(M) \to^{+}_{\lambda_{\circledR}} \lfloor k'\rfloor(M)$ for every $\lambda_{\circledR}$ term $M$ ;
$k =_{\lambda_{\circledR}} k'$ iff $\lfloor k\rfloor_{\mathsf{k}}(M) \equiv_{\lambda_{\circledR}} \lfloor k'\rfloor_{\mathsf{k}}(M)$ or $\lfloor k\rfloor_{\mathsf{k}}(M) \equiv \lfloor k'\rfloor(M)$ for every $\lambda_{\circledR}$ term $M$;

(ii) $e >_c e'$ iff $\|e\|_{\mathsf{c}} > \|e'\|_{\mathsf{c}}$; $e =_c e'$ iff $\|e\|_{\mathsf{c}} = \|e'\|_{\mathsf{c}}$;

(iii) $e >_w e'$ iff $\|e\|_{\mathsf{w}} > \|e'\|_{\mathsf{w}}$; $e =_w e'$ iff $\|e\|_{\mathsf{w}} = \|e'\|_{\mathsf{w}}$;

The lexicographic product of two orders $>_1$ and $>_2$ is defined as [2]:

$$a >_1 \times_{lex} >_2 b \Leftrightarrow a >_1 b \text{ or } (a =_1 b \text{ and } a >_2 b).$$

**Definition 46.** We define the relation $\gg$ on $\Lambda^{\mathsf{Gtz}}_{\circledR}$ as the lexicographic product:

$$\gg \;=\; >_{\lambda_{\circledR}} \times_{lex} >_c \times_{lex} >_w .$$

The following propositions proves that the reduction relation on the set of typed $\lambda^{\mathsf{Gtz}}_{\circledR}$-expressions is included in the given lexicographic product $\gg$.

**Proposition 47.** *For each* $e \in \Lambda^{\mathsf{Gtz}}_{\circledR}$*: if* $e \to e'$, *then* $e \gg e'$.

*Proof.* By case analysis on the kind of reduction and the structure of $\gg$.
If $e \to e'$ by $\beta_{\mathsf{g}}$, $\sigma$, $\pi$, $\mu$, $\gamma_1$, $\gamma_2$, $\gamma_3$, $\gamma_4$ $\gamma_5$, $\gamma\omega_1$, $\gamma\omega_2$, $\omega_1$, $\omega_2$, $\omega_3$ $\omega_4$ or $\omega_5$ reduction, then $e >_{\lambda_{\circledR}} e'$ by Proposition 41.
If $e \to e'$ by $\gamma_6$, then $e =_{\lambda_{\circledR}} e'$ by Proposition 41, and $e >_c e'$ by Lemma 43.
Finally, if $e \to e'$ by $\omega_6$, then $e =_{\lambda_{\circledR}} e'$ by Proposition 41, $e =_c e'$ by Lemma 43 and $e >_w e'$ by Lemma 44. □

Strong normalisation of $\to$ is another terminology for the well-foundness of the relation $\to$ and it is well-known that a relation included in a well-founded relation is well-founded and that the lexicographic product of well-founded relations is well-founded.

**Theorem 48** (Strong normalisation of the $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$)**.** *Each expression in* $\Lambda^{\mathsf{Gtz}}_{\circledR}\cap$ *is strongly normalising.*

*Proof.* The reduction $\to$ is well-founded on $\Lambda^{\mathsf{Gtz}}_{\circledR}\cap$ as it is included (Proposition 47) in the relation $\gg$ which is well-founded as the lexicographic product of the well-founded relations $>_{\lambda_{\circledR}}$, $>_c$ and $>_w$. Relation $>_{\lambda_{\circledR}}$ is based on the interpretation $\lfloor\ \rfloor : \Lambda^{\mathsf{Gtz}}_{\circledR} \to \Lambda_{\circledR}$. By Proposition 36 typeability is preserved by the interpretation $\lfloor\ \rfloor$ and $\to_{\lambda_{\circledR}}$ is strongly normalising (i.e., well-founded) on $\Lambda_{\circledR}\cap$ (Section 1.3), hence $>_{\lambda_{\circledR}}$ is well-founded on $\Lambda^{\mathsf{Gtz}}_{\circledR}\cap$. Similarly, $>_c$ and $>_w$ are well-founded, as they are based on interpretations into the well-founded relation $>$ on the set $\mathbb{N}$ of natural numbers. □

## 2.4 SN $\Rightarrow$ Typeability in $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$

Now, we want to prove that if a $\lambda^{\mathsf{Gtz}}_{\circledR}$-term is SN, then it is typeable in the system $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$. We follow the procedure used in Section 1.4. The proofs are similar to the ones in Section 1.4.

The abstract syntax of $\lambda^{\mathsf{Gtz}}_{\circledR}$-normal forms is the following:

$$\begin{array}{rcl} t_{nf} & ::= & x \,|\, \lambda x.t_{nf} \,|\, \lambda x.x \odot t_{nf} \,|\, x(t_{nf} :: k_{nf}) \,|\, x <^{y}_{z} y(t_{nf} :: k_{nf}) \\ k_{nf} & ::= & \widehat{x}.t_{nf} \,|\, \widehat{x}.x \odot t_{nf} \,|\, t_{nf} :: k_{nf} \,|\, x <^{y}_{z} (t_{nf} :: k_{nf}),\ y \in Fv(t_{nf}), z \in Fv(k_{nf}) \\ w_{nf} & ::= & x \odot e_{nf} \,|\, x \odot w_{nf} \end{array}$$

We use $e_{nf}$ for any $\lambda^{\mathsf{Gtz}}_{\circledR}$-expression in the normal form.

**Proposition 49.** *$\lambda^{\mathsf{Gtz}}_{\circledR}$-normal forms are typeable in the system $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$.*

*Proof.* By mutual induction on the structure of $t_{nf}$, $k_{nf}$ and $w_{nf}$. □

The following two lemmas explain the behavior of the meta operators $[\ /\ ]$ and $@$ during expansion.

**Lemma 50** (Inverse substitution lemma)**.**

*(i) Let $\Gamma \vdash t[u/x] : \sigma$ and $u$ typeable. Then, there exist $\Delta_j$ and $\tau_j$, $j = 0,\ldots,n$ such that $\Delta_j \vdash u : \tau_j$ and $\Gamma', x : \cap_i^n \tau_i \vdash t : \sigma$, where $\Gamma = \Gamma', \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.*

*(ii) Let $\Gamma;\gamma \vdash k[u/x] : \sigma$ and $u$ typeable. Then, there are $\Delta_j$ and $\tau_j$, $j = 0,\ldots,n$ such that $\Delta_j \vdash u : \tau_j$ and $\Gamma', x : \cap_i^n \tau_i;\gamma \vdash k : \sigma$, where $\Gamma = \Gamma', \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.*

*Proof.* By mutual induction on the structure of terms and contexts. □

**Lemma 51** (Inverse append lemma)**.** *If $\Gamma;\alpha \vdash k@k' : \sigma$, then there are $\Delta_j$ and $\tau_j$, $j = 0,\ldots,n$ such that $\Delta_j;\alpha \vdash k : \tau_j$ and $\Gamma';\cap_i^n \tau_i \vdash k' : \sigma$, where $\Gamma = \Gamma', \Delta_0^{\top} \sqcap \Delta_1 \sqcap \ldots \sqcap \Delta_n$.*

*Proof.* By induction on the structure of the context $k$. □

Now we prove that the type of a term is preserved during the expansion.

**Proposition 52** (Head subject expansion)**.** *For every $\lambda^{\mathsf{Gtz}}_{\circledR}$-term $t$: if $t \to t'$, $t$ is contracted redex and $\Gamma \vdash t' : \sigma$, then $\Gamma \vdash t : \sigma$.*

*Proof.* By case study according to the applied reduction. □

**Theorem 53** (SN $\Rightarrow$ typeability)**.** *All strongly normalising $\lambda^{\mathsf{Gtz}}_{\circledR}$ terms are typeable in the $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$ system.*

*Proof.* Analogous to the proof of Theorem 26. □

Now we give a characterisation of strong normalisation in $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculus.

**Theorem 54.** *In $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculus, the term $t$ is strongly normalising if and only if it is typeable in $\lambda^{\mathsf{Gtz}}_{\circledR}\cap$.*

*Proof.* Immediate consequence of Theorems 48 and 53. □

# 3 Intersection types for the resource control lambda calculus with explicit substitution $\lambda^{x}_{\mathbb{R}}$

## 3.1 Resource control lambda calculus with explicit substitution $\lambda^{x}_{\mathbb{R}}$

The *resource control* lambda calculus with explicit substitution $\lambda^{x}_{\mathbb{R}}$, is an extension of the $\lambda_{x}$-calculus with explicit operators for weakening and contraction. It corresponds to the $\lambda$lxr-calculus of Kesner and Lengrand, proposed in [35], and also represents a vertex of "the prismoid of resources".

The *pre-terms* of $\lambda^{x}_{\mathbb{R}}$-calculus are given by the following abstract syntax:

$$\text{Pre-terms} \qquad f \quad ::= \quad x \,|\, \lambda x.f \,|\, ff \,|\, f\langle x := f\rangle \,|\, x \odot f \,|\, x <^{x_1}_{x_2} f$$

The only point of difference with respect to $\lambda_{\mathbb{R}}$-calculus is the operator of explicit substitution $\langle := \rangle$.

The set of free variables of a pre-term $f$, denoted by $Fv(f)$, is defined as follows:

$$Fv(x) = x; \quad Fv(\lambda x.f) = Fv(f) \setminus \{x\};$$
$$Fv(fg) = Fv(f) \cup Fv(g); \quad Fv(f\langle x := g\rangle) = (Fv(f) \setminus \{x\}) \cup Fv(g)$$
$$Fv(x \odot f) = \{x\} \cup Fv(f); \quad Fv(x <^{x_1}_{x_2} f) = \{x\} \cup Fv(f) \setminus \{x_1, x_2\}.$$

In $f\langle x := g\rangle$, the substitution binds the variable $x$ in $f$.

The set of $\lambda^{x}_{\mathbb{R}}$-*terms*, denoted by $\Lambda^{x}_{\mathbb{R}}$ and ranged over by $M, N, P, M_1, \ldots$. is a subset of the set of pre-terms, defined by the rules in Figure 9.

$$\frac{}{x \in \Lambda^{x}_{\mathbb{R}}} \qquad \frac{f \in \Lambda^{x}_{\mathbb{R}} \;\; x \in Fv(f)}{\lambda x.f \in \Lambda^{x}_{\mathbb{R}}}$$

$$\frac{f \in \Lambda^{x}_{\mathbb{R}} \;\; g \in \Lambda^{x}_{\mathbb{R}} \;\; Fv(f) \cap Fv(g) = \emptyset}{fg \in \Lambda^{x}_{\mathbb{R}}}$$

$$\frac{f \in \Lambda^{x}_{\mathbb{R}} \;\; g \in \Lambda^{x}_{\mathbb{R}} \;\; x \in Fv(f) \;\; (Fv(f) \setminus \{x\}) \cap Fv(g) = \emptyset}{f\langle x := g\rangle \in \Lambda^{x}_{\mathbb{R}}}$$

$$\frac{f \in \Lambda^{x}_{\mathbb{R}} \;\; x \notin Fv(f)}{x \odot f \in \Lambda^{x}_{\mathbb{R}}} \qquad \frac{f \in \Lambda^{x}_{\mathbb{R}} \;\; x_1 \neq x_2, \;\; x_1, x_2 \in Fv(f) \;\; x \notin Fv(f) \setminus \{x_1, x_2\}}{x <^{x_1}_{x_2} f \in \Lambda^{x}_{\mathbb{R}}}$$

Figure 9: $\Lambda^{x}_{\mathbb{R}}$: $\lambda^{x}_{\mathbb{R}}$-terms

The notion of terms corresponds to the notion of linear terms in [35].

The reduction rules of $\lambda^{x}_{\mathbb{R}}$-calculus are presented in Figure 10.

In the $\lambda^{x}_{\mathbb{R}}$, one works modulo equivalencies given in Figure 11.

## 3.2 Intersection types for $\lambda^{x}_{\mathbb{R}}$

In this subsection we introduce intersection type assignment system which assigns *strict types* to $\lambda^{x}_{\mathbb{R}}$-terms. The system is syntax-directed, hence significantly different

| | | | |
|---|---|---|---|
| $(\beta_x)$ | $(\lambda x.M)N$ | $\to$ | $M\langle x := N\rangle$ |
| $(\sigma_1)$ | $x\langle x := N\rangle$ | $\to$ | $N$ |
| $(\sigma_2)$ | $(\lambda y.M)\langle x := N\rangle$ | $\to$ | $\lambda y.M\langle x := N\rangle$ |
| $(\sigma_3)$ | $(MP)\langle x := N\rangle$ | $\to$ | $M\langle x := N\rangle P$, if $x \notin Fv(P)$ |
| $(\sigma_4)$ | $(MP)\langle x := N\rangle$ | $\to$ | $MP\langle x := N\rangle$, if $x \notin Fv(M)$ |
| $(\sigma_5)$ | $(x \odot M)\langle x := N\rangle$ | $\to$ | $Fv(N) \odot M$ |
| $(\sigma_6)$ | $(y \odot M)\langle x := N\rangle$ | $\to$ | $y \odot M\langle x := N\rangle$, if $x \neq y$ |
| $(\sigma_7)$ | $(x <^{x_1}_{x_2} M)\langle x := N\rangle$ | $\to$ | $Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M\langle x_1 := N_1\rangle\langle x_2 := N_2\rangle$ |
| $(\sigma_8)$ | $(M\langle x := N\rangle)\langle y := P\rangle$ | $\to$ | $M\langle x := N\langle y := P\rangle\rangle$, if $y \notin Fv(M) \setminus \{x\}$ |
| $(\gamma_1)$ | $x <^{x_1}_{x_2} (\lambda y.M)$ | $\to$ | $\lambda y.x <^{x_1}_{x_2} M$ |
| $(\gamma_2)$ | $x <^{x_1}_{x_2} (MN)$ | $\to$ | $(x <^{x_1}_{x_2} M)N$, if $x_1, x_2 \notin Fv(N)$ |
| $(\gamma_3)$ | $x <^{x_1}_{x_2} (MN)$ | $\to$ | $M(x <^{x_1}_{x_2} N)$, if $x_1, x_2 \notin Fv(M)$ |
| $(\gamma_4)$ | $x <^{x_1}_{x_2} (M\langle y := N\rangle)$ | $\to$ | $M\langle y := x <^{x_1}_{x_2} N\rangle$, if $x_1, x_2 \notin Fv(M) \setminus \{y\}$ |
| $(\omega_1)$ | $\lambda x.(y \odot M)$ | $\to$ | $y \odot (\lambda x.M)$, $x \neq y$ |
| $(\omega_2)$ | $(x \odot M)N$ | $\to$ | $x \odot (MN)$ |
| $(\omega_3)$ | $M(x \odot N)$ | $\to$ | $x \odot (MN)$ |
| $(\omega_4)$ | $M\langle y := x \odot N\rangle$ | $\to$ | $x \odot (M\langle y := N\rangle)$ |
| $(\gamma\omega_1)$ | $x <^{x_1}_{x_2} (y \odot M)$ | $\to$ | $y \odot (x <^{x_1}_{x_2} M)$, $y \neq x_1, x_2$ |
| $(\gamma\omega_2)$ | $x <^{x_1}_{x_2} (x_1 \odot M)$ | $\to$ | $M\langle x_2 := x\rangle$ |

Figure 10: Reduction rules of $\lambda^{x}_{\circledR}$-calculus

| | | | |
|---|---|---|---|
| $(\varepsilon_1)$ | $x \odot (y \odot M)$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $y \odot (x \odot M)$ |
| $(\varepsilon_2)$ | $x <^{x_1}_{x_2} M$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $x <^{x_2}_{x_1} M$ |
| $(\varepsilon_3)$ | $x <^{y}_{z} (y <^{u}_{v} M)$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $x <^{y}_{u} (y <^{z}_{v} M)$ |
| $(\varepsilon_4)$ | $x <^{x_1}_{x_2} (y <^{y_1}_{y_2} M)$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $y <^{y_1}_{y_2} (x <^{x_1}_{x_2} M)$, $x \neq y_1, y_2$, $y \neq x_1, x_2$ |
| $(\varepsilon_5)$ | $M\langle x := N\rangle\langle y := P\rangle$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $M\langle y := P\rangle\langle x := N\rangle$, $x \notin Fv(P)$, $y \notin Fv(M)$ |
| $(\varepsilon_6)$ | $(y <^{y_1}_{y_2} M)\langle x := N\rangle$ | $\equiv_{\lambda^{x}_{\circledR}}$ | $y <^{y_1}_{y_2} M\langle x := N\rangle$, $x \neq y$, $y_1, y_2 \notin Fv(N)$ |

Figure 11: Equivalences in $\lambda^{x}_{\circledR}$-calculus

from the one proposed in [42].

The syntax of types and the definitions of type assignment, basis, etc. are the same as in the case of the system $\lambda_{\mathbb{R}}\cap$. The type assignment system $\lambda^{x}_{\mathbb{R}}\cap$ is given in Figure 12. The only difference with respect to the $\lambda_{\mathbb{R}}\cap$ is the presence of one new type assignment rule, namely $(Subst)$ for typing the explicit substitution. The rules $(\to_E)$ and $(Subst)$ are constructed in the same manner, as explained in subsection 1.2.

$$\frac{}{x:\sigma\vdash x:\sigma}\ (Ax)$$

$$\frac{\Gamma,x:\alpha\vdash M:\sigma}{\Gamma\vdash\lambda x.M:\alpha\to\sigma}\ (\to_I) \qquad \frac{\Gamma\vdash M:\cap_i^n\tau_i\to\sigma \quad \Delta_0\vdash N:\tau_0 \quad \dots \quad \Delta_n\vdash N:\tau_n}{\Gamma,\Delta_0^{\top}\sqcap\Delta_1\sqcap\dots\sqcap\Delta_n\vdash MN:\sigma}\ (\to_E)$$

$$\frac{\Gamma,x:\cap_i^n\tau_i\vdash M:\sigma \quad \Delta_0\vdash N:\tau_0 \quad \dots \quad \Delta_n\vdash N:\tau_n}{\Gamma,\Delta_0^{\top}\sqcap\Delta_1\sqcap\dots\sqcap\Delta_n\vdash M\langle x:=N\rangle:\sigma}\ (Subst)$$

$$\frac{\Gamma,x:\alpha,y:\beta\vdash M:\sigma}{\Gamma,z:\alpha\cap\beta\vdash z<^{x}_{y}M:\sigma}\ (Cont) \qquad \frac{\Gamma\vdash M:\sigma}{\Gamma,x:\top\vdash x\odot M:\sigma}\ (Weak)$$

Figure 12: $\lambda^{x}_{\mathbb{R}}\cap$: $\lambda^{x}_{\mathbb{R}}$-calculus with intersection types

**Proposition 55** (Generation lemma for $\lambda^{x}_{\mathbb{R}}\cap$)**.**

*(i)* $\Gamma\vdash\lambda x.M:\tau$ *iff there exist* $\alpha$ *and* $\sigma$ *such that* $\tau\equiv\alpha\to\sigma$ *and* $\Gamma,x:\alpha\vdash M:\sigma$.

*(ii)* $\Gamma\vdash MN:\sigma$ *iff there exist* $\Delta_j$ *and* $\tau_j$, $j=0,\dots,n$ *such that* $\Delta_j\vdash N:\tau_j$ *and* $\Gamma'\vdash M:\cap_i^n\tau_i\to\sigma$, *moreover* $\Gamma=\Gamma',\Delta_0^{\top}\sqcap\Delta_1\sqcap\dots\sqcap\Delta_n$.

*(iii)* $\Gamma\vdash M\langle x:=N\rangle:\sigma$ *iff there exist a type* $\alpha=\cap_{j=0}^n\tau_j$, *such that for all* $j\in\{0,\dots,n\}$, $\Delta_j\vdash N:\tau_j$ *and* $\Gamma',x:\cap_i^n\tau_i\vdash M:\sigma$, *moreover* $\Gamma=\Gamma',x:\alpha,\Delta_0^{\top}\sqcap\Delta_1\sqcap\dots\sqcap\Delta_n$.

*(iv)* $\Gamma\vdash z<^{x}_{y}M:\sigma$ *iff there exist* $\Gamma',\alpha,\beta$ *such that* $\Gamma=\Gamma',z:\alpha\cap\beta$ *and* $\Gamma',x:\alpha,y:\beta\vdash M:\sigma$.

*(v)* $\Gamma\vdash x\odot M:\sigma$ *iff* $\Gamma=\Gamma',x:\top$ *and* $\Gamma'\vdash M:\sigma$.

The proposed system also satisfies preservation of free variables, bases intersection and subject reduction and equivalence.

# 4 Conclusions

In this paper, we have proposed intersection type assignment systems for:

- resource control lambda calculus $\lambda_{\mathbb{R}}$, which corresponds to $\lambda_{\mathtt{CW}}$ of [36];

- resource control sequent lambda calculus $\lambda^{\mathsf{Gtz}}_{\circledR}$ of [26] and
- resource control calculus with explicit substitution $\lambda^{x}_{\circledR}$ of [35].

The three intersection type assignment systems proposed here give a complete characterization of strongly normalizing terms for these three calculi. The strong normalisation of typeable resource control lambda terms is proved directly by an appropriate modification of the reducibility method, whereas the same property for resource control sequent lambda terms is proved by well-founded lexicographic order based on suitable embedding into the former calculus and the strong normalisation of the calculus with explicit substitution is given by its interpretation in the resource control lambda calculus. This paper expands the range of the intersection type techniques and combines different methods in the strict types environment. It should be noticed that the strict control on the way variables are introduced determines the way terms are typed in a given environment. Basically, in a given environment no irrelevant intersection types are introduced. The flexibility on the choice of a type for a term, as it is used in rule $(\rightarrow_E)$ in Figure 5, comes essentially from the choice one has in invoking the axiom. Unlike the approach of introducing non-idempotent intersection types into the calculus with some kind of resource management [47], our intersection is idempotent. As a consequence, our type assignment system corresponds to full intuitionistic logic, while non-idempotent intersection type assignment systems correspond to intuitionistic linear logic.

The three presented calculi $\lambda_{\circledR}$, $\lambda^{\mathsf{Gtz}}_{\circledR}$ and $\lambda^{x}_{\circledR}$ are good candidates to investigate the computational content of substructural logics [56], both in natural deduction and sequent calculus. The motivation for these logics comes from philosophy (Relevant Logics), linguistics (Lambek Calculus) to computing (Linear Logic). Since the basic idea of resource control is to explicitly handle structural rules, the control operators could be used to handle the absence of (some) structural rules in substructural logics such as weakening, contraction, commutativity, associativity. This would be an interesting direction for further research. Another direction will involve the investigation of the use of intersection types, being a powerful means for building models of lambda calculus [6, 16], in constructing models for sequent lambda calculi. Finally, one may wonder how the strict control on the duplication and the erasure of variables influences the type reconstruction of terms [11, 38].

**Acknowledgements:** We would like to thank the ICTAC 2011 anonymous referees for their careful reading and many valuable comments, which helped us improve the final version of the paper. We would also like to thank Dragiša Žunić for participating in the earlier stages of the work.

# References

[1] S. Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111(1&2):3–57, 1993.

[2] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, UK, 1998.

[3] F. Barbanera and S. Berardi. A symmetric lambda calculus for classical program extraction. *Information and Computation*, 125(2):103–117, 1996.

[4] H. P. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. North-Holland, Amsterdam, revised edition, 1984.

[5] H. P. Barendregt. Lambda calculi with types. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, UK, 1992.

[6] H. P. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48(4):931–940 (1984), 1983.

[7] N. Benton, G. Bierman, V. de Paiva, and M. Hyland. A term calculus for intuitionistic linear logic. In Marc Bezem and Jan Friso Groote, editors, *1st International Conference on Typed Lambda Calculus, TLCA '93*, volume 664 of *Lecture Notes in Computer Science*, pages 75–90. Springer, 1993.

[8] R. Bloo and K. H. Rose. Preservation of strong normalisation in named lambda calculi with explicit substitution and garbage collection. In *Computer Science in the Netherlands, CSN '95*, pages 62–72, 1995.

[9] G. Boudol. The lambda-calculus with multiplicities (abstract). In E. Best, editor, *4th International Conference on Concurrency Theory, CONCUR '93*, volume 715 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 1993.

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

[11] G. Boudol and P. Zimmer. On type inference in the intersection type discipline. *Electronic Notes in Theoretical Computer Science*, 136:23–42, 2005.

[12] M. Coppo and M. Dezani-Ciancaglini. A new type-assignment for lambda terms. *Archiv für Mathematische Logik*, 19:139–156, 1978.

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

[14] P.-L. Curien and H. Herbelin. The duality of computation. In *5th International Conference on Functional Programming, ICFP'00*, pages 233–243. ACM Press, 2000.

[15] M. Dezani-Ciancaglini and S. Ghilezan. Two behavioural lambda models. In H. Geuvers and F. Wiedijk, editors, *Types for Proofs and Programs*, volume 2646 of *Lecture Notes in Computer Science*, pages 127–147. Springer, 2003.

[16] M. Dezani-Ciancaglini, S. Ghilezan, and S. Likavec. Behavioural Inverse Limit Models. *Theoretical Computer Science*, 316(1–3):49–74, 2004.

[17] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional characterization of λ-terms using intersection types. In *25th International Symposium on Mathematical Foundations of Computer Science, MFCS '00*, volume 1893 of *Lecture Notes in Computer Science*, pages 304–314. Springer, 2000.

[18] D. J. Dougherty, S. Ghilezan, and P. Lescanne. Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: extending the Coppo-Dezani heritage. *Theoretical Computer Science*, 398:114–128, 2008.

[19] T. Ehrhard and L. Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309(1-3):1–41, 2003.

[20] J. Espírito Santo. Completing Herbelin's programme. In S. Ronchi Della Rocca, editor, *9th International Conference on Typed Lambda Calculi and Applications, TLCA '07*, volume 4583 of *Lecture Notes in Computer Science*, pages 118–132. Springer, 2007.

[21] J. Espírito Santo, S. Ghilezan, and J. Ivetić. Characterising strongly normalising intuitionistic sequent terms. In *International Workshop TYPES'07 (Selected Papers)*, volume 4941 of *Lecture Notes in Computer Science*, pages 85–99. Springer, 2008.

[22] J. Espírito Santo, J. Ivetić, and S. Likavec. Characterising strongly normalising intuitionistic terms. *Fundamenta Informaticae*, 2011. To appear.

[23] J. Gallier. Typing untyped λ-terms, or reducibility strikes again! *Annals of Pure and Applied Logic*, 91:231–270, 1998.

[24] G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935.

[25] S. Ghilezan. Strong normalization and typability with intersection types. *Notre Dame Journal of Formal Logic*, 37(1):44–52, 1996.

[26] S. Ghilezan, J. Ivetić, P. Lescanne, and D. Žunić. Intuitionistic sequent-style calculus with explicit structural rules. In *8th International Tbilisi Symposium on Language, Logic and Computation*, volume 6618 of *LNAI*, pages 101–124, 2011.

[27] S. Ghilezan and S. Likavec. Computational interpretations of logics. In Z. Ognjanović, editor, *Collection of Papers, special issue Logic in Computer Science 20(12)*, pages 159–215. Mathematical Institute of Serbian Academy of Sciences and Arts, 2009.

[28] Silvia Ghilezan, Jelena Ivetić, Pierre Lescanne, and Silvia Likavec. Intersection types for the resource control lambda calculi. In Antonio Cerone and Pekka Pihlajasaari, editors, *8th International Colloquium on Theoretical Aspects of Computing, ICTAC '11*, volume 6916 of *Lecture Notes in Computer Science*, pages 116–134. Springer, 2011.

[29] Silvia Ghilezan and Silvia Likavec. Reducibility: A Ubiquitous Method in Lambda Calculus with Intersection Types. In Steffen van Bakel, editor, *ITRS '02*, volume 70 of *Electronic Notes in Theoretical Computer Science*, pages 106–123, 2002.

[30] J.-Y. Girard. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'elimination des coupures dans l'analyse et la théorie des types. In J. E. Fenstad, editor, *2nd Scandinavian Logic Symposium*, pages 63–92. North-Holland, 1971.

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

[32] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoret Computer Science*. Cambridge University Press, 1989.

[33] H. Herbelin. A lambda calculus structure isomorphic to Gentzen-style sequent calculus structure. In L. Pacholski and J. Tiuryn, editors, *Computer Science Logic, CSL '94*, volume 933 of *Lecture Notes in Computer Science*, pages 61–75. Springer, 1995.

[34] W. A. Howard. The formulas-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, London, 1980.

[35] D. Kesner and S. Lengrand. Resource operators for lambda-calculus. *Information and Computation*, 205(4):419–473, 2007.

[36] D. Kesner and F. Renaud. The prismoid of resources. In R. Královič and D. Niwiński, editors, *34th International Symposium on Mathematical Foundations of Computer Science, MFCS '09*, volume 5734 of *Lecture Notes in Computer Science*, pages 464–476. Springer, 2009.

[37] D. Kesner and F. Renaud. A prismoid framework for languages with resources. *Theoretical Computer Science*, 412(37):4867–4892, 2011.

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

[39] K. Kikuchi. Simple proofs of characterizing strong normalisation for explicit substitution calculi. In F. Baader, editor, *18th International Conference on Term Rewriting and Applications, RTA'07*, volume 4533 of *Lecture Notes in Computer Science*, pages 257–272. Springer, 2007.

[40] G. Koletsos. Church-Rosser theorem for typed functionals. *Journal of Symbolic Logic*, 50:782–790, 1985.

[41] J.-L. Krivine. *Lambda-calcul types et modèles*. Masson, Paris, 1990.

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

[43] Ralph Matthes. Characterizing strongly normalizing terms of a calculus with generalized applications via intersection types. In *ICALP Satellite Workshops*, pages 339–354, 2000.

[44] J. C. Mitchell. Type systems for programming languages. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, Volume B, pages 415–431. Elsevier, Amsterdam, 1990.

[45] J. C. Mitchell. *Foundation for Programmimg Languages*. MIT Press, Boston, 1996.

[46] P. M. Neergaard. Theoretical pearls: A bargain for intersection types: a simple strong normalization proof. *Journal of Functional Programming*, 15(5):669–677, 2005.

[47] M. Pagani and S. Ronchi Della Rocca. Solvability in resource lambda-calculus. In C.-H. L. Ong, editor, *13th International Conference on Foundations of Software Science and Computational Structures, FOSSACS 2010*, volume 6014 of *Lecture Notes in Computer Science*, pages 358–373. Springer, 2010.

[48] M. Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *3rd International Conference on Logic Programming and Automated Reasoning, LPAR '92*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[49] Luis Pinto and Roy Dyckhoff. Sequent calculi for the normal terms of the $\lambda\pi$ and $\lambda\pi\sigma$ calculi. *Electronic Notes in Theoretical Computer Science*, 17:1–14, 1998.

[50] G. Pottinger. A type assignment for the strongly normalizable $\lambda$-terms. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 561–577. Academic Press, London, 1980.

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

[52] K H. Rose. CRSX - Combinatory Reduction Systems with Extensions. In Manfred Schmidt-Schauß, editor, *22nd International Conference on Rewriting Techniques and Applications, RTA'11*, volume 10 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 81–90. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2011.

[53] K. H. Rose. Implementation Tricks That Make CRSX Tick. Talk at IFIP 1.6 workshop, RDP'2011, 2011.

[54] Kristoffer Rose, Roel Bloo, and Frédéric Lang. On explicit substitution with names. *Journal of Automated Reasoning*, pages 1–26, 2011.

[55] P. Sallé. Une extension de la théorie des types en lambda-calcul. In G. Ausiello and C. Böhm, editors, *5th International Conference on Automata, Languages and Programming, ICALP '78*, volume 62 of *Lecture Notes in Computer Science*, pages 398–410. Springer, 1978.

[56] P. Schroeder-Heister and K. Došen. *Substructural Logics*. Oxford University Press, UK, 1993.

[57] R. Statman. Logical relations and the typed λ-calculus. *Information and Control*, 65:85–97, 1985.

[58] W. W. Tait. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic*, 32:198–212, 1967.

[59] W. W. Tait. A realizability interpretation of the theory of species. In R. Parikh, editor, *Logic Colloquium*, volume 453 of *Lecture Notes in Mathematics*, pages 240–251. Springer, 1975.

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

[61] V. van Oostrom. Net-calculus. Course notes, Utrecht University, 2001.

[62] D. Žunić. *Computing with sequents and diagrams in classical logic - calculi ${}^*\mathcal{X}$, ${}^d\mathcal{X}$ and ${}^{\copyright}\mathcal{X}$*. Phd thesis, École Normale Supérieure de Lyon, 2007.