

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

S.Ghilezan^a, J.Ivetić^a, P.Lescanne^b, S.Likavec^c

^a*University of Novi Sad, Faculty of Technical Sciences, Serbia*

^b*University of Lyon, École Normal Supérieure de Lyon, France*

^c*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 λ -calculus captures the computational content of intuitionistic natural deduction through Curry-Howard correspondence [35]. This connection between logic and computation can be extended to other calculi and logical systems [30]: Herbelin's $\bar{\lambda}$ -calculus [34], Pinto and Dyckhoff's $\lambda\pi\sigma$ -calculus [51] and Espírito Santo's λ^{Gtz} -calculus [21] correspond to intuitionistic sequent calculus. In the realm of classical logic, Parigot's $\lambda\mu$ -calculus [50] cor-

Email addresses: gsilvia@uns.ac.rs (S.Ghilezan), jelenaivetic@uns.ac.rs (J.Ivetić), pierre.lescanne@ens-lyon.fr (P.Lescanne), likavec@di.unito.it (S.Likavec)

responds to classical natural deduction, whereas Barbanera and Berardi's symmetric calculus [4] and Curien and Herbelin's $\bar{\lambda}\mu\tilde{\mu}$ -calculus [15] correspond to its sequent version. Extending first, the λx calculus of explicit substitution and later λ -calculus and λ^{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 [25], as investigated in [36, 37, 28].

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 [13, 14, 52, 57]. The newly obtained intersection type assignment systems enabled complete characterisation of termination of term calculi [3, 24, 26]. The extension of Curry-Howard correspondence to other formalisms brought the need for intersection types into many different settings [19, 40, 44, 47].

Our work is inspired by and extends Kesner and Lengrand's [36] work on resource operators for λ -calculus with explicit substitution. Their linear λlr -calculus introduces operators for linear substitution, erasure and duplication, preserving at the same time confluence and full composition of explicit substitutions of its predecessor λx [9, 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 λ -calculus was proposed by Ghilezan et al. in [28], whereas resource control both in λ -calculus and λx -calculus was further developed in [37, 38].

In order to control all resources, in the spirit of λI -calculus (see e.g. [5]), 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 $x <_{x_2}^{x_1} M$, 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 re-

ally 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, 8]. In practice, for instance in the description of compilers by rules with binders [55, 56], the implementation of substitutions of linear variables by inlining¹ 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 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: $\lambda_{\mathbb{R}}$ (the extension of the λ -calculus), $\lambda_{\mathbb{R}}^{\text{Gtz}}$ (the extension of the sequent lambda calculus λ^{Gtz}) and $\lambda_{\mathbb{R}}^{\times}$ (the extension of the λx -calculus with explicit substitution). We then introduce intersection types into all three calculi $\lambda_{\mathbb{R}}$, $\lambda_{\mathbb{R}}^{\text{Gtz}}$ and $\lambda_{\mathbb{R}}^{\times}$. Our intersection type assignment systems $\lambda_{\mathbb{R}} \cap$, $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ and $\lambda_{\mathbb{R}}^{\times} \cap$ 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 [27], 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 λ -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 $\lambda_{\mathbb{R}}$ -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 $\lambda_{\mathbb{R}}$ -calculus by using typeability of normal forms and head subject expansion.

Further, we prove strong normalisation for $\lambda_{\mathbb{R}}^{\text{Gtz}}$ and $\lambda_{\mathbb{R}}^{\times}$, in Section 2 and

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.

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

Related work. The idea to control the use of variables can be traced back to Church's λI -calculus [5]. Currently there are several different lines of research in resource aware term calculi. Van Oostrom [48] and later Kesner and Lengrand [36], applying ideas from linear logic [32], proposed to extend the λ -calculus and the λx -calculus, with operators to control the use of variables (resources). Generalising this approach, Kesner and Renaud [37, 38] 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 λ -calculus by Boudol [10] initialised investigations in resource aware non-deterministic λ -calculus with multiplicities and a generalised notion of application [11]. The theory was connected to linear logic via differential λ -calculus in [20] and typed with non-idempotent intersection types in [49]. In this paper we follow the notation of [62] and [27], which is related to [48].

This paper is an extended and revised version of [27]. In addition to $\lambda_{\mathbb{R}}$ -calculus and $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus presented in [27], this extended version adds the treatment of the $\lambda_{\mathbb{R}}^{\times}$ -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 $\lambda_{\mathbb{R}}$ -calculus is improved.

Outline of the paper. In Section 1 we first give the syntax and reduction rules for $\lambda_{\mathbb{R}}$ -calculus, followed by the intersection type assignment system and the characterisation of strong normalisation. Section 2 deals with $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus, its syntax, reduction rules, intersection type assignment system and the characterisation of strong normalisation. Section 3 introduces $\lambda_{\mathbb{R}}^{\times}$ -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 $\lambda_{\mathbb{R}}$	5
1.1	Resource control lambda calculus $\lambda_{\mathbb{R}}$	5
1.2	Intersection types for $\lambda_{\mathbb{R}}$	12
1.3	Typeability \Rightarrow SN in $\lambda_{\mathbb{R}} \cap$	17
1.4	SN \Rightarrow Typeability in $\lambda_{\mathbb{R}} \cap$	24
2	Intersection types for the sequent resource control lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$	26
2.1	Resource control sequent lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$	26
2.2	Intersection types for $\lambda_{\mathbb{R}}^{\text{Gtz}}$	28
2.3	Typeability \Rightarrow SN in $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$	31
2.4	SN \Rightarrow Typeability in $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$	37
3	Intersection types for the resource control lambda calculus with explicit substitution $\lambda_{\mathbb{R}}^x$	38
3.1	Resource control lambda calculus with explicit substitution $\lambda_{\mathbb{R}}^x$	38
3.2	Intersection types for $\lambda_{\mathbb{R}}^x$	39
3.3	Typeability \Rightarrow SN in $\lambda_{\mathbb{R}}^x \cap$	41
3.4	SN \Rightarrow Typeability in $\lambda_{\mathbb{R}}^x \cap$	46
4	Conclusions	46

1. Intersection types for the resource control lambda calculus $\lambda_{\mathbb{R}}$

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

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

The *resource control* lambda calculus, $\lambda_{\mathbb{R}}$, is an extension of the λ -calculus with explicit operators for weakening and contraction. It corresponds to the λ_{cw} -calculus of Kesner and Renaud, proposed in [37] as a vertex of “the prismoid of resources”, where substitution is implicit. We use a notation along the lines of [62] and close to [48]. It is slightly modified w.r.t. [37] 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:

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

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 $x <_{x_2}^{x_1} f$ is a *contraction*. The contraction operator is assumed to be insensitive to the order of the arguments x_1 and x_2 , i.e. $x <_{x_2}^{x_1} f = x <_{x_1}^{x_2} f$.

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

$$\begin{aligned} Fv(x) &= x; & Fv(\lambda x.f) &= Fv(f) \setminus \{x\}; & Fv(ff) &= Fv(f) \cup Fv(g); \\ Fv(x \odot f) &= \{x\} \cup Fv(f); & Fv(x <_{x_2}^{x_1} f) &= \{x\} \cup Fv(f) \setminus \{x_1, x_2\}. \end{aligned}$$

In $x <_{x_2}^{x_1} f$, 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, \dots is a subset of the set of pre-terms, defined in Figure 1.

$\frac{}{x \in \Lambda_{\mathbb{R}}} \qquad \frac{f \in \Lambda_{\mathbb{R}} \quad x \in Fv(f)}{\lambda x.f \in \Lambda_{\mathbb{R}}}$
$\frac{f \in \Lambda_{\mathbb{R}} \quad g \in \Lambda_{\mathbb{R}} \quad Fv(f) \cap Fv(g) = \emptyset}{fg \in \Lambda_{\mathbb{R}}}$
$\frac{f \in \Lambda_{\mathbb{R}} \quad x \notin Fv(f)}{x \odot f \in \Lambda_{\mathbb{R}}} \qquad \frac{f \in \Lambda_{\mathbb{R}} \quad x_1 \neq x_2 \quad x_1, x_2 \in Fv(f) \quad x \notin Fv(f) \setminus \{x_1, x_2\}}{x <_{x_2}^{x_1} f \in \Lambda_{\mathbb{R}}}$

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 [36]. 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 $\lambda_{\mathbb{R}}$ -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 $\lambda_{\mathbb{R}}$ -terms.*

In the sequel, we use the notation $X \odot M$ for $x_1 \odot \dots x_n \odot M$ and $X <_Z^Y M$ for $x_1 <_{z_1}^{y_1} \dots x_n <_{z_n}^{y_n} M$, where X, Y and Z are lists of size n , consisting of all distinct variables $x_1, \dots, x_n, y_1, \dots, y_n, z_1, \dots, z_n$. 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 [5] 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 \mathbb{R} of reduction rules $\rightarrow_{\lambda_{\mathbb{R}}}$ of the $\lambda_{\mathbb{R}}$ -calculus is presented in Figure 2.

(β)	$(\lambda x.M)N \rightarrow M[N/x]$
(γ_1)	$x <_{x_2}^{x_1} (\lambda y.M) \rightarrow \lambda y.x <_{x_2}^{x_1} M$
(γ_2)	$x <_{x_2}^{x_1} (MN) \rightarrow (x <_{x_2}^{x_1} M)N$, if $x_1, x_2 \notin Fv(N)$
(γ_3)	$x <_{x_2}^{x_1} (MN) \rightarrow M(x <_{x_2}^{x_1} N)$, if $x_1, x_2 \notin Fv(M)$
(ω_1)	$\lambda x.(y \odot M) \rightarrow y \odot (\lambda x.M)$, $x \neq y$
(ω_2)	$(x \odot M)N \rightarrow x \odot (MN)$
(ω_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 \mathbb{R} of reduction rules of the $\lambda_{\mathbb{R}}$ -calculus

The reduction rules are divided into four groups. The main computational step is β reduction. The group of (γ) reductions perform propagation of contraction into the expression. Similarly, (ω) 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 β -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.

$x[N/x]$	\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	$Fv(N) \odot M$
$(y <_{y_2}^{y_1} M)[N/x]$	\triangleq	$y <_{y_2}^{y_1} M[N/x], \ x \neq y$
$(x <_{x_2}^{x_1} M)[N/x]$	\triangleq	$Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M[N_1/x_1, N_2/x_2]$

Figure 3: Substitution in $\lambda_{\mathbb{R}}$ -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 $\lambda_{\mathbb{R}}$ -calculus, one works modulo equivalencies given in Figure 4.

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

Proposition 3. *If $M \rightarrow M'$ 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.

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

(ϵ_1)	$x \odot (y \odot M) \equiv_{\lambda_{\mathbb{R}}} y \odot (x \odot M)$
(ϵ_2)	$x <_{x_2}^{x_1} M \equiv_{\lambda_{\mathbb{R}}} x <_{x_1}^{x_2} M$
(ϵ_3)	$x <_z^y (y <_v^u M) \equiv_{\lambda_{\mathbb{R}}} x <_u^y (y <_v^z M)$
(ϵ_4)	$x <_{x_2}^{x_1} (y <_{y_2}^{y_1} M) \equiv_{\lambda_{\mathbb{R}}} y <_{y_2}^{y_1} (x <_{x_2}^{x_1} M), \quad x \neq y_1, y_2, y \neq x_1, x_2$

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

- 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 $x[N[P/z]/x] \triangleq N[P/z]$
- $((\lambda y.M)[N/x])[P/z] \triangleq (\lambda y.M[N/x])[P/z] \triangleq \lambda y.(M[N/x])[P/z] =_{IH} \lambda y.M[N[P/z]/x] \triangleq (\lambda y.M)[N[P/z]/x], \quad x, z \neq y$
- $x \notin FV(Q)$ (the case $x \notin FV(M)$ is analogous)
 $((MQ)[N/x])[P/z] \triangleq (M[N/x]Q)[P/z] =_{z \in FV(N)} (M[N/x])[P/z]Q =_{IH} M[N[P/z]/x]Q = (MQ)[N[P/z]/x]$
- $((y \odot M)[N/x])[P/z] \triangleq (y \odot M[N/x])[P/z] \triangleq y \odot M[N/x][P/z] =_{IH} y \odot M[N[P/z]/x] \triangleq (y \odot M)[N[P/z]/x], \quad y \neq x, z$
- $((x \odot M)[N/x])[P/z] \triangleq (FV(N) \odot M)[P/z] =_{z \in FV(N)} (z \odot \{FV(N) \setminus \{z\}\} \odot M)[P/z] = \{FV(P) \cup FV(N) \setminus z\} \odot M = FV(N[P/z]) \odot M \triangleq (x \odot M)[N[P/z]/x]$
- $((y <_{y_2}^{y_1} M)[N/x])[P/z] \triangleq (y <_{y_2}^{y_1} M[N/x])[P/z] \triangleq y <_{y_2}^{y_1} M[N/x][P/z] =_{IH} y <_{y_2}^{y_1} M[[P/z]N/x] =_{IH} (y <_{y_2}^{y_1} M)[[P/z]N/x], \quad x \neq y$
- $((x <_{x_2}^{x_1} M)[N/x])[P/z] \triangleq (FV(N) <_{FV(N_2)}^{FV(N_1)} M[N_1/x_1, N_2/x_2])[P/z] \triangleq FV(N) <_{FV(N_2)}^{FV(N_1)} M[N_1/x_1][N_2/x_2][P/z] = z <_{z_2}^{z_1} FV(N) \setminus \{z\} <_{FV(N_2) \setminus \{z_2\}}^{FV(N_1) \setminus \{z_1\}} M[N_1/x_1][N_2/x_2][P/z] \triangleq FV(P) <_{FV(P_2)}^{FV(P_1)} FV(N) \setminus \{z\} <_{FV(N_2) \setminus \{z_2\}}^{FV(N_1) \setminus \{z_1\}} M[N_1/x_1][N_2/x_2][P_1/z_1][P_2/z_2] =_{IH}$

$$Fv(P) \cup Fv(N) \setminus \{z\} <_{Fv(P_1) \cup Fv(N_1) \setminus \{z_1\}}^{Fv(P_1) \cup Fv(N_1) \setminus \{z_1\}} M[N_1[P_1/z_1]/x_1, N_2[P_2/z_2]/x_2] \triangleq (x <_{x_2}^{x_1} M)[N[P/z]/x].$$

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 $\lambda_{\mathbb{R}}$ -calculus, i.e., $\rightarrow^* \triangleq (\rightarrow_{\lambda_{\mathbb{R}}} \cup \equiv_{\lambda_{\mathbb{R}}})^*$.

Lemma 5.

- (i) $M[y \odot N/x] \rightarrow^* y \odot M[N/x]$
- (ii) $y <_{y_2}^{y_1} M[N/x] \rightarrow^* M[y <_{y_2}^{y_1} N/x]$, 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 $x \notin Fv(P)$.
- $M = z <_{z_2}^{z_1} P$. Then $M[y \odot N/x] = (z <_{z_2}^{z_1} P)[y \odot N/x] \triangleq z <_{z_2}^{z_1} P[y \odot N/x] \rightarrow_{IH} z <_{z_2}^{z_1} (y \odot P[N/x]) \rightarrow_{\gamma\omega_1} y \odot (z <_{z_2}^{z_1} P[N/x]) = y \odot M[N/x]$.
- $M = x <_{x_2}^{x_1} P$. Then $M[y \odot N/x] = (x <_{x_2}^{x_1} P)[y \odot N/x] \triangleq Fv(y \odot N) <_{Fv(y_2 \odot N_2)}^{Fv(y_1 \odot N_1)} P[y_1 \odot N_1/x_1, y_2 \odot N_2/x_2] \rightarrow_{IH} Fv(y \odot N) <_{Fv(y_2 \odot N_2)}^{Fv(y_1 \odot N_1)} y_1 \odot y_2 \odot P[N_1/x_1, N_2/x_2] = Fv(N) <_{Fv(N_2)}^{Fv(N_1)} y <_{y_2}^{y_1} y_1 \odot y_2 \odot P[N_1/x_1, N_2/x_2] \rightarrow_{\gamma\omega_2} Fv(N) <_{Fv(N_2)}^{Fv(N_1)} y \odot P[N_1/x_1, N_2/x_2] \rightarrow_{\gamma\omega_1} y \odot Fv(N) <_{Fv(N_2)}^{Fv(N_1)} P[N_1/x_1, N_2/x_2] \triangleq y \odot (x <_{x_2}^{x_1} P[N/x]) = y \odot M[N/x]$.

- (ii) – $M = x$. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} x[N/x] \triangleq y <_{y_2}^{y_1} N \triangleq x[y <_{y_2}^{y_1} N/x] = M[y <_{y_2}^{y_1} N/x]$.
- $M = \lambda z.P$. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (\lambda z.P)[N/x] \triangleq y <_{y_2}^{y_1} \lambda z.P[N/x] \rightarrow_{\gamma_1} \lambda z.y <_{y_2}^{y_1} P[N/x] \rightarrow_{IH} \lambda z.P[y <_{y_2}^{y_1} N/x] \triangleq (\lambda z.P)[y <_{y_2}^{y_1} N/x] = M[y <_{y_2}^{y_1} N/x]$.
- $M = PQ$, $x \notin Fv(Q)$. The case when $x \notin Fv(P)$ is analogous. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (PQ)[N/x] \triangleq y <_{y_2}^{y_1} P[N/x]Q \rightarrow_{\gamma_2} (y <_{y_2}^{y_1} P[N/x])Q \rightarrow_{IH} P[y <_{y_2}^{y_1} N/x]Q \triangleq (PQ)[y <_{y_2}^{y_1} N/x] = M[y <_{y_2}^{y_1} N/x]$.
- $M = z \odot P$, where $z \neq x, y_1, y_2$. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (z \odot P)[N/x] \triangleq y <_{y_2}^{y_1} z \odot P[N/x] \rightarrow_{\gamma\omega_1} z \odot y <_{y_2}^{y_1} P[N/x] \rightarrow_{IH} z \odot P[y <_{y_2}^{y_1} N/x] \triangleq (z \odot P)[y <_{y_2}^{y_1} N/x] = M[y <_{y_2}^{y_1} N/x]$.
- $M = x \odot P$. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (x \odot P)[N/x] \triangleq y <_{y_2}^{y_1} Fv(N) \odot P$. Since $y_1, y_2 \in Fv(N)$ we have that $y <_{y_2}^{y_1} y_1 \odot y_2 \odot Fv(N) \setminus \{y_1, y_2\} \odot P \rightarrow_{\gamma\omega_2} y \odot Fv(N) \setminus \{y_1, y_2\} \odot P$. On the other hand, $M[y <_{y_2}^{y_1} N/x] = (x \odot P)[y <_{y_2}^{y_1} N/x] \triangleq Fv(y <_{y_2}^{y_1} N) \odot P = y \odot Fv(N) \setminus \{y_1, y_2\} \odot P$, so the proposition is proved.
- $M = z <_{z_2}^{z_1} P$. Then $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (z <_{z_2}^{z_1} P)[N/x] \triangleq y <_{y_2}^{y_1} z <_{z_2}^{z_1} P[N/x] \equiv_{\lambda_{\mathbb{R}}} z <_{z_2}^{z_1} y <_{y_2}^{y_1} P[N/x] \rightarrow_{IH} z <_{z_2}^{z_1} P[y <_{y_2}^{y_1} N/x] \triangleq (z <_{z_2}^{z_1} P)[y <_{y_2}^{y_1} N/x] = M[y <_{y_2}^{y_1} N/x]$.
- $y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} (x <_{x_2}^{x_1} P)[N/x] \triangleq y <_{y_2}^{y_1} Fv(N) <_{Fv(N_2)}^{Fv(N_1)} P[N_1/x_1, N_2/x_2] = y <_{y_2}^{y_1} y_1 <_{y_2'}^{y_1'} y_2 <_{y_2''}^{y_1''} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\mathbb{R}}} y <_{y_1'}^{y_1} y_1 <_{y_2'}^{y_2} y_2 <_{y_2''}^{y_1''} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\mathbb{R}}} y <_{y_1'}^{y_1} y_1 <_{y_1''}^{y_2} y_2 <_{y_2''}^{y_2'} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\mathbb{R}}} y <_{y_2}^{y_1} y_1 <_{y_1''}^{y_1'} y_2 <_{y_2''}^{y_2'} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} P[N_1/x_1, N_2/x_2] \equiv_{\lambda_{\mathbb{R}}} y <_{y_2}^{y_1} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} y_2 <_{y_2''}^{y_2'} y_1 <_{y_1''}^{y_1'} P[N_1/x_1, N_2/x_2] \rightarrow_{IHx2} y <_{y_2}^{y_1} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y_2', y_2''\}}^{Fv(N_1) \setminus \{y_1', y_1''\}} P[(y_1 <_{y_1''}^{y_1'} N_1)/x_1, (y_2 <_{y_2''}^{y_2'} N_2)/x_2]$
- On the other hand, rewriting the right hand side yields:
 $M[y <_{y_2}^{y_1} N/x] = (x <_{x_2}^{x_1} P)[y <_{y_2}^{y_1} N/x] \triangleq$

$$Fv(y <_{y_2}^{y_1} N) <_{Fv(y_4 <_{y_2}^{y_1} N'_2)}^{Fv(y_3 <_{y_2}^{y_1} N'_1)} P[(y_3 <_{y_2}^{y_1} N'_1)/x_1, (y_4 <_{y_2}^{y_1} N'_2)/x_2]$$

By renaming $y_1 \rightarrow y'_1$ and $y_2 \rightarrow y''_1$ in $y_3 <_{y_2}^{y_1} N'_1$ and $y_1 \rightarrow y'_2$ and $y_2 \rightarrow y''_2$ in $y_4 <_{y_2}^{y_1} N'_2$ we get

$$Fv(y <_{y_2}^{y_1} N) <_{Fv(y_4 <_{y_2}^{y'_2} N_2)}^{Fv(y_3 <_{y_2}^{y'_1} N_1)} P[(y_3 <_{y_1}^{y'_1} N_1)/x_1, (y_4 <_{y_2}^{y'_2} N_2)/x_2]$$

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 $y_3 \rightarrow y_1$ and $y_4 \rightarrow y_2$ we get

$$Fv(y <_{y_2}^{y_1} N) <_{Fv(y_2 <_{y_2}^{y'_2} N_2)}^{Fv(y_1 <_{y_1}^{y'_1} N_1)} P[(y_1 <_{y_1}^{y'_1} N_1)/x_1, (y_2 <_{y_2}^{y'_2} N_2)/x_2] =$$

$$y <_{y_2}^{y_1} Fv(N) \setminus \{y_1, y_2\} <_{Fv(N_2) \setminus \{y'_2, y''_2\}}^{Fv(N_1) \setminus \{y'_1, y''_1\}} P[(y_1 <_{y_1}^{y'_1} N_1)/x_1, (y_2 <_{y_2}^{y'_2} N_2)/x_2]$$

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 $M = x <_{x_2}^{x_1} x_1 x_2$ and $N = y_1 y_2$. Then

$$y <_{y_2}^{y_1} M[N/x] = y <_{y_2}^{y_1} x <_{x_2}^{x_1} x_1 x_2 [(y_1 y_2)/x] \triangleq$$

$$y <_{y_2}^{y_1} Fv(y_1 y_2) <_{Fv(w_1 w_2)}^{Fv(z_1 z_2)} x_1 x_2 [(z_1 z_2)/x_1, (w_1 w_2)/x_2] =$$

$$y <_{y_2}^{y_1} y_1 <_{w_1}^{z_1} y_2 <_{w_2}^{z_2} x_1 x_2 [(z_1 z_2)/x_1, (w_1 w_2)/x_2] \equiv_{\lambda_{\mathbb{R}}, (3 \times \varepsilon_3)}$$

$$y <_{y_2}^{y_1} y_1 <_{z_2}^{z_1} y_2 <_{w_2}^{w_1} (z_1 z_2) (w_1 w_2) = M_1.$$

On the other hand:

$$x <_{x_2}^{x_1} x_1 x_2 [y <_{y_2}^{y_1} y_1 y_2 / x] \triangleq$$

$$Fv(y <_{y_2}^{y_1} y_1 y_2) <_{Fv(y_4 <_{y_2}^{y_1} y_1 y_2)}^{Fv(y_3 <_{y_2}^{y_1} y_1 y_2)} (x_1 x_2) [(y_3 <_{y_2}^{y_1} y_1 y_2)/x_1, (y_4 <_{y_2}^{y_1} y_1 y_2)/x_2] =$$

$$y <_{y_4}^{y_3} (y_3 <_{y_2}^{y_1} y_1 y_2) (y_4 <_{y_2}^{y_1} y_1 y_2).$$

By renaming $y_1 \rightarrow z_1$, $y_2 \rightarrow z_2$ in the first bracket, and $y_1 \rightarrow w_1$, $y_2 \rightarrow w_2$ in the second one we obtain: $y <_{y_4}^{y_3} (y_3 <_{z_2}^{z_1} z_1 z_2) (y_4 <_{w_2}^{w_1} w_1 w_2)$.

By renaming $y_3 \rightarrow y_1$, $y_4 \rightarrow y_2$ we get $y <_{y_2}^{y_1} (y_1 <_{z_2}^{z_1} z_1 z_2) (y_2 <_{w_2}^{w_1} w_1 w_2) = M_2$.

Finally, $M_1 \rightarrow_{\gamma_2, \gamma_3} M_2$.

1.2. Intersection types for $\lambda_{\mathbb{R}}$

In this subsection we introduce an intersection type assignment system which assigns *strict types* to $\lambda_{\mathbb{R}}$ -terms. Strict types were proposed in [3] and used in [23]

for characterisation of strong normalisation in λ^{Gtz} -calculus.

The syntax of types is defined as follows:

$$\begin{array}{ll} \text{Strict types } \sigma & ::= p \mid \alpha \rightarrow \sigma \\ \text{Types } \alpha & ::= \bigcap_i^n \sigma_i \end{array}$$

where p ranges over a denumerable set of type atoms, and $\bigcap_i^n \sigma_i$ stands for $\sigma_1 \cap \dots \cap \sigma_n$, $n \geq 0$. Particularly, if $n = 0$, then $\bigcap_i^0 \sigma_i$ represents the *neutral element* for the intersection operator, denoted by \top .

We denote types with $\alpha, \beta, \gamma, \dots$, strict types with $\sigma, \tau, \rho, \upsilon, \dots$ 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 $(\bigcap_i^n \tau_i) \rightarrow \sigma$.

Definition 7.

- (i) A *basic type assignment* is an expression of the form $x : \alpha$, where x is a term variable and α is a type.
- (ii) A *basis* Γ is a set $\{x_1 : \alpha_1, \dots, x_n : \alpha_n\}$ of basic type assignments, where all term variables are different. $\text{Dom}(\Gamma) = \{x_1, \dots, x_n\}$. A *basis extension* $\Gamma, x : \alpha$ denotes the set $\Gamma \cup \{x : \alpha\}$, where $x \notin \text{Dom}(\Gamma)$.
- (iii) A *bases intersection* is defined as:

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

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

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

The type assignment system $\lambda_{\text{R}} \cap$ is given in Figure 5. Notice that in the syntax of λ_{R} 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*)

$$\boxed{
\begin{array}{c}
\frac{}{x : \sigma \vdash x : \sigma} (Ax) \\
\\
\frac{\Gamma, x : \alpha \vdash M : \sigma}{\Gamma \vdash \lambda x. M : \alpha \rightarrow \sigma} (\rightarrow_I) \quad \frac{\Gamma \vdash M : \cap_i^n \tau_i \rightarrow \sigma \quad \Delta_0 \vdash N : \tau_0 \dots \Delta_n \vdash N : \tau_n}{\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash MN : \sigma} (\rightarrow_E) \\
\\
\frac{\Gamma, x : \alpha, y : \beta \vdash M : \sigma}{\Gamma, z : \alpha \cap \beta \vdash z <_y^x M : \sigma} (Cont) \quad \frac{\Gamma \vdash M : \sigma}{\Gamma, x : \top \vdash x \odot M : \sigma} (Weak)
\end{array}
}$$

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

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 (\rightarrow_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 Δ_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 Δ_0 can be any of the other environments and the type of N the associated type. Since Δ^\top is a neutral element for \sqcap , then Δ^\top disappears in the bottom of the rule. The case for $n = 0$ resembles the rules $(drop)$ and/or $(K-cut)$ in [43] 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 (\rightarrow_I) are traditional, i.e. they are the same as in the simply typed λ -calculus. Notice however that the type of the variable in (Ax) is a strict type.

Lemma 8 (Domain Correspondence for $\lambda_{\mathbb{R}}\cap$). *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, namely (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*, namely (\rightarrow_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*; namely (\rightarrow_E) .

Notice that $(Cont)$ introduces and removes variables. \square

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

Lemma 9 (Generation lemma for $\lambda_{\mathbb{R}}\cap$).

- (i) $\Gamma \vdash \lambda x.M : \tau$ iff there exist α and σ such that $\tau \equiv \alpha \rightarrow \sigma$ and $\Gamma, x : \alpha \vdash M : \sigma$.
- (ii) $\Gamma \vdash MN : \sigma$ iff there exist Δ_j and τ_j , $j = 0, \dots, n$ such that $\Delta_j \vdash N : \tau_j$ and $\Gamma' \vdash M : \cap_i^n \tau_i \rightarrow \sigma$, moreover $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$.
- (iii) $\Gamma \vdash z <_y^x M : \sigma$ iff there exist Γ', α, β 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. \square

The proposed system satisfies the following properties.

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

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 $\Gamma, x : \cap_i^n \tau_i \vdash x \odot P : \sigma$ and $\Delta_i \vdash N : \tau_i$ for all $i \in \{0, \dots, n\}$, 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 $\Gamma, \Delta_0^\top \vdash Fv(N) \odot P : \sigma$ 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 $\Delta_j \vdash N : \tau_j$ for all $j \in \{0, \dots, n\}$, by renaming the variables in Δ_j (i.e. the free variables of N) we get two different sets of sequents: $\Delta'_j \vdash N_1 : \tau_j$ and $\Delta''_j \vdash N_2 : \tau_j$. By applying IH twice, we get $\Gamma, \Delta'_0 \sqcap \Delta'_1 \sqcap \dots \sqcap \Delta'_m, \Delta''_0 \sqcap \Delta''_{m+1} \sqcap \dots \sqcap \Delta''_n \vdash (P[N_1/x_1])[N_2/x_2] : \sigma$. 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 Δ'_l and Δ''_k by introducing again the original names of the free variables of N from Δ_j and finally get what we need:

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

□

Proposition 11 (Subject reduction and equivalence). *For every $\lambda_{\mathbb{R}}$ -term M : if $\Gamma \vdash M : \sigma$ and $M \rightarrow M'$ 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 loss of 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 (β) : 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 $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$, and that there is a type $\cap_{j=0}^n \tau_j$ such that for all $j = 0, \dots, n$, $\Delta_j \vdash N : \tau_j$, and $\Gamma' \vdash \lambda x.M : \cap_i^n \tau_i \rightarrow \sigma$. 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 $\Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash M[N/x] : \sigma$ which is exactly what we need, since $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$.
- Case $(\gamma\omega_2)$: Let $\Gamma \vdash x <_{x_2}^{x_1} x_1 \odot M : \sigma$. We show now 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 $x : \tau_j \vdash x : \tau_j$, $j = 0, \dots, n$ 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 \Rightarrow SN in $\lambda_{\mathbb{R}} \cap$

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 [60] for proving the strong normalisation of simply typed λ -calculus, and developed further to prove *strong normalisation* of various calculi in [61, 31, 42, 26, 29], *confluence* (the Church-Rosser property) of $\beta\eta$ -reduction in [41, 59, 45, 46, 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 [24, 18, 16, 17].

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

$$\frac{M' \in \mathcal{SN} \quad M \rightarrow M'}{M \in \mathcal{SN}}$$

Definition 12. For $\mathcal{M}, \mathcal{N} \subseteq \Lambda_{\mathbb{R}}$, we define $\mathcal{M} \rightarrow \mathcal{N} \subseteq \Lambda_{\mathbb{R}}$ as

$$\mathcal{M} \rightarrow \mathcal{N} = \{N \in \Lambda_{\mathbb{R}} \mid \forall M \in \mathcal{M} \quad NM \in \mathcal{N}\}.$$

Definition 13. The type interpretation $\llbracket - \rrbracket : \text{Types} \rightarrow 2^{\Lambda_{\mathbb{R}}}$ is defined by:

- (I1) $\llbracket p \rrbracket = \mathcal{SN}$, where p is a type atom;
- (I2) $\llbracket \alpha \rightarrow \sigma \rrbracket = \llbracket \alpha \rrbracket \rightarrow \llbracket \sigma \rrbracket$;
- (I3) $\llbracket \cap_i^n \sigma_i \rrbracket = \cap_i^n \llbracket \sigma_i \rrbracket$ and $\llbracket \cap_i^0 \sigma_i \rrbracket = \mathcal{SN}$.

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 [6], whereas reduction property corresponds to the property CR 2 in Chapter 6 of [33]. To this aim we will use the following notation: recall that \mathbb{R} denotes the set of reductions given in Figure 2. If $\mu \in \mathbb{R}$, then $\text{redex}_\mu^\varsigma$ denotes a redex, that is a term which is an instance by the meta-substitution ς of the left hand side of the reduction μ . Whereas $\text{contr}_\mu^\varsigma$ denotes the instance of the right hand side of the same reduction μ by the same meta-substitution ς .³

Definition 14.

- A set $\mathcal{X} \subseteq \Lambda_{\mathbb{R}}$ satisfies the *variable property*, notation $\text{VAR}(\mathcal{X})$, if \mathcal{X} contains all the terms of the form $xM_1 \dots M_n$ for $M_i \in \mathcal{S}\mathcal{N}$.
- A set $\mathcal{X} \subseteq \Lambda_{\mathbb{R}}$ satisfies the *reduction property*, notation $\text{RED}(\mathcal{X})$, if \mathcal{X} is stable by reduction, in other words $M \in \mathcal{X}$ and $M \rightarrow M'$ imply $M' \in \mathcal{X}$.
- A set $\mathcal{X} \subseteq \Lambda_{\mathbb{R}}$ satisfies the *expansion property*, notation $\text{EXP}_\mu(\mathcal{X})$ where μ is a rule in \mathbb{R} , if⁴:

$$\frac{M_1 \in \mathcal{S}\mathcal{N} \dots M_n \in \mathcal{S}\mathcal{N} \quad \text{contr}_\mu^\varsigma M_1 \dots M_n \in \mathcal{X}}{\text{redex}_\mu^\varsigma M_1 \dots M_n \in \mathcal{X}.} \text{EXP}_\mu(\mathcal{X})$$

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

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

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

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

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

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

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

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

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

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

- (i) \mathcal{SN} is \mathbb{R} -saturated.
- (ii) If \mathcal{M} and \mathcal{N} are \mathbb{R} -saturated, then $\mathcal{M} \rightarrow \mathcal{N}$ is \mathbb{R} -saturated.
- (iii) If \mathcal{M} and \mathcal{N} are \mathbb{R} -saturated, then $\mathcal{M} \cap \mathcal{N}$ is \mathbb{R} -saturated.
- (iv) For all types $\phi \in \text{Types}$, $[[\phi]]$ is \mathbb{R} -saturated.

Proof. (i)

- $\mathcal{SN} \subseteq \mathcal{SN}$, $\text{VAR}(\mathcal{SN})$ and $\text{RED}(\mathcal{SN})$ trivially hold.
- $\text{EXP}_{\beta}(\mathcal{SN})$. Suppose that $M[N/x]M_1 \dots M_n \in \mathcal{SN}$ and $M_1, \dots, M_n \in \mathcal{SN}$. 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, $M_1, \dots, M_n \in \mathcal{SN}$, so the reductions inside of these terms terminate. After finitely many reduction steps, we obtain $(\lambda x.M)NM_1 \dots M_n \rightarrow \dots \rightarrow (\lambda x.M')N'M'_1 \dots M'_n$ where $M \rightarrow M'$, $N \rightarrow N'$, $M_1 \rightarrow M'_1, \dots, M_n \rightarrow M'_n$. After contracting $(\lambda x.M')N'M'_1 \dots M'_n$ to $M'[N'/x]M'_1 \dots M'_n$, we obtain a reduct of $M[N/x]M_1 \dots M_n \in \mathcal{SN}$. Hence, $(\lambda x.M)NM_1 \dots M_n \in \mathcal{SN}$.
- $\text{EXP}_{\mu}(\mathcal{SN})$. Analogous to $\text{EXP}_{\beta}(\mathcal{SN})$.
- $\text{WEAK}(\mathcal{SN})$. 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.
- $\text{CONT}(\mathcal{SN})$. Suppose that $M \in \mathcal{SN}$, $y \neq z$, $y, z \in Fv(M)$, $x \notin Fv(M) \setminus \{y, z\}$. We prove $x <_z^y M \in \mathcal{SN}$ by induction on the structure of M .
 - $M = \lambda w.N$. Then $N \in \mathcal{SN}$ and $x <_z^y M = x <_z^y (\lambda w.N) \rightarrow_{\gamma_1} \lambda w.x <_z^y N \in \mathcal{SN}$, since $x <_z^y N \in \mathcal{SN}$ by IH.

- $M = PQ$. Then $P, Q \in \mathcal{SN}$ and if $y, z \notin Fv(Q)$, $x <_z^y M = x <_z^y (PQ) \rightarrow_{\gamma_2} (x <_z^y P)Q \in \mathcal{SN}$, since by IH $x <_z^y P \in \mathcal{SN}$.
The case of \rightarrow_{γ_3} reduction is analogous.
- $M = w \odot N$. Then $x <_z^y M = x <_z^y (w \odot N) \rightarrow_{\gamma_{\omega_1}} w \odot (x <_z^y N)$. By IH $x <_z^y N \in \mathcal{SN}$ and $w \odot (x <_z^y N)$ does not introduce any new redexes.
- $M = y \odot N$. Then $x <_z^y M = x <_z^y (y \odot N) \rightarrow_{\gamma_{\omega_2}} N[x/z] \in \mathcal{SN}$, since $N \in \mathcal{SN}$ by IH.

(ii)

- $\mathcal{M} \rightarrow \mathcal{N} \subseteq \mathcal{SN}$. Suppose that $M \in \mathcal{M} \rightarrow \mathcal{N}$. Then, for all $N \in \mathcal{M}$, $MN \in \mathcal{N}$. Since \mathcal{M} is \mathbb{R} -saturated, $\text{VAR}(\mathcal{M})$ holds so $x \in \mathcal{M}$ and $Mx \in \mathcal{N} \subseteq \mathcal{SN}$. From here we can deduce that $M \in \mathcal{SN}$.
- $\text{VAR}(\mathcal{M} \rightarrow \mathcal{N})$. Suppose that $x \in \text{var}$, and $M_1, \dots, M_n \in \mathcal{SN}, n \geq 0$, such that $x \cap Fv(M_1) \cap \dots \cap Fv(M_n) = \emptyset$. We need to show that $xM_1 \dots M_n \in \mathcal{M} \rightarrow \mathcal{N}$, i.e. $\forall N \in \mathcal{M}, xM_1 \dots M_n N \in \mathcal{N}$. This holds since by IH $\mathcal{M} \subseteq \mathcal{SN}$ and \mathcal{N} is \mathbb{R} -saturated, i.e. $\text{VAR}(\mathcal{N})$ holds.
- $\text{RED}(\mathcal{M} \rightarrow \mathcal{N})$. Let $M \in \mathcal{M} \rightarrow \mathcal{N}$ and M' be such that $M \rightarrow M'$ and let $N \in \mathcal{M}$. We know that $MN \in \mathcal{N}$ and $MN \rightarrow M'N$. By IH, $M'N \in \mathcal{N}$ hence $M' \in \mathcal{M} \rightarrow \mathcal{N}$.
- $\text{EXP}_\beta(\mathcal{M} \rightarrow \mathcal{N})$. Suppose that $M[N/x]M_1 \dots M_n \in \mathcal{M} \rightarrow \mathcal{N}$ and $M_1, \dots, M_n \in \mathcal{SN}$. This means that for all $P \in \mathcal{M}$, $M[N/x]M_1 \dots M_n P \in \mathcal{N}$. But \mathcal{N} is \mathbb{R} -saturated, so $\text{EXP}_\beta(\mathcal{N})$ holds and we have that for all $P \in \mathcal{N}$, $(\lambda x.M)NM_1 \dots M_n P \in \mathcal{N}$. This means that $(\lambda x.M)NM_1 \dots M_n \in \mathcal{M} \rightarrow \mathcal{N}$.
- $\text{EXP}_\mu(\mathcal{M} \rightarrow \mathcal{N})$. Analogous to $\text{EXP}_\beta(\mathcal{M} \rightarrow \mathcal{N})$.
- $\text{WEAK}(\mathcal{M} \rightarrow \mathcal{N})$. Suppose that $M \in \mathcal{M} \rightarrow \mathcal{N}$ and $x \notin Fv(M)$. This means that for all $N \in \mathcal{M}, MN \in \mathcal{N}$. But \mathcal{N} is \mathbb{R} -saturated, i.e. $\text{WEAK}(\mathcal{N})$ holds, hence $x \odot (MN) \in \mathcal{N}$. Also $\text{EXP}_{\omega_2}(\mathcal{N})$ holds so we obtain for all $N \in \mathcal{M}, (x \odot M)N \in \mathcal{N}$, i.e. $x \odot M \in \mathcal{M} \rightarrow \mathcal{N}$.
- $\text{CONT}(\mathcal{M} \rightarrow \mathcal{N})$. Let $M \in \mathcal{M} \rightarrow \mathcal{N}$. We want to prove that $x <_z^y M \in \mathcal{M} \rightarrow \mathcal{N}$ 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 $(x <_z^y M)P \in \mathcal{N}$. Since $M \in \mathcal{M} \rightarrow \mathcal{N}$, we know that $MP \in \mathcal{N}$. By IH $x <_z^y (MP) \in \mathcal{N}$. By reduction γ_2 and hence by $\text{RED}(\mathcal{N})$ we have $(x <_z^y M)P \in \mathcal{N}$. Therefore $x <_z^y M \in \mathcal{M} \rightarrow \mathcal{N}$.

(iii)

- $\mathcal{M} \cap \mathcal{N} \subseteq \mathcal{SN}$ is straightforward, since $M, N \subseteq \mathcal{SN}$ by IH.
- $\text{VAR}(\mathcal{M} \cap \mathcal{N})$. Since $\text{VAR}(\mathcal{M})$ and $\text{VAR}(\mathcal{N})$ hold, we have that $\forall M_1, \dots, M_n \in \mathcal{SN}, n \geq 0: xM_1 \dots M_n \in \mathcal{M}$ and $xM_1 \dots M_n \in \mathcal{N}$. We deduce that $\forall M_1, \dots, M_n \in \mathcal{SN}, n \geq 0: xM_1 \dots M_n \in \mathcal{M} \cap \mathcal{N}$, i.e. $\text{VAR}(\mathcal{M} \cap \mathcal{N})$ holds.
- $\text{RED}(\mathcal{M} \cap \mathcal{N})$ is straightforward.
- $\text{EXP}_\beta(\mathcal{M} \cap \mathcal{N})$ and $\text{EXP}_\mu(\mathcal{M} \cap \mathcal{N})$ are straightforward.
- $\text{WEAK}(\mathcal{M} \cap \mathcal{N})$. 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 \mathbb{R} -saturated $\text{WEAK}(\mathcal{M})$ and $\text{WEAK}(\mathcal{N})$ hold, hence by IH $x \odot M \in \mathcal{M}$ and $x \odot M \in \mathcal{N}$, i.e. $x \odot M \in \mathcal{M} \cap \mathcal{N}$.
- $\text{CONT}(\mathcal{M} \cap \mathcal{N})$. 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 \mathbb{R} -saturated $\text{CONT}(\mathcal{M})$ and $\text{CONT}(\mathcal{N})$ hold, hence by IH $x <_z^y M \in \mathcal{M}$ and $x <_z^y M \in \mathcal{N}$, i.e. $x <_z^y M \in \mathcal{M} \cap \mathcal{N}$.

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

- If $\varphi \equiv p$, p a type atom, then $\llbracket \varphi \rrbracket = \mathcal{SN}$, so it is \mathbb{R} -saturated using (i).
- If $\varphi \equiv \alpha \rightarrow \sigma$, then $\llbracket \varphi \rrbracket = \llbracket \alpha \rrbracket \rightarrow \llbracket \sigma \rrbracket$. Since $\llbracket \alpha \rrbracket$ and $\llbracket \sigma \rrbracket$ are \mathbb{R} -saturated by IH, we can use (ii).
- If $\varphi \equiv \cap_i^n \sigma_i$, then $\llbracket \varphi \rrbracket = \llbracket \cap_i^n \sigma_i \rrbracket = \cap_i^n \llbracket \sigma_i \rrbracket$ and for all $i = 1, \dots, n$, $\llbracket \sigma_i \rrbracket$ are \mathbb{R} -saturated by IH, so we can use (iii). If $\varphi \equiv \cap_i^0 \sigma_i$, then $\llbracket \varphi \rrbracket = \mathcal{SN}$

□

We further define a *valuation of terms* $\llbracket - \rrbracket_\rho : \Lambda_{\mathbb{R}} \rightarrow \Lambda_{\mathbb{R}}$ and the *semantic satisfiability relation* \models connecting the type interpretation with the term valuation.

Definition 17. Let $\rho : \text{var} \rightarrow \Lambda_{\mathbb{R}}$ be a valuation of term variables in $\Lambda_{\mathbb{R}}$. For $M \in \Lambda_{\mathbb{R}}$, with $Fv(M) = \{x_1, \dots, x_n\}$ the *term valuation* $\llbracket - \rrbracket_\rho : \Lambda_{\mathbb{R}} \rightarrow \Lambda_{\mathbb{R}}$ is defined as follows:

$$\llbracket M \rrbracket_\rho = M[\rho(x_1)/x_1, \dots, \rho(x_n)/x_n].$$

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) $\llbracket MN \rrbracket_\rho = \llbracket M \rrbracket_\rho \llbracket N \rrbracket_\rho$
- (ii) $\llbracket \lambda x.M \rrbracket_\rho N \rightarrow \llbracket M \rrbracket_{\rho(N/x)}$.
- (iii) $\llbracket x \odot M \rrbracket_\rho = Fv(\rho(x)) \odot \llbracket M \rrbracket_\rho$.
- (iv) $\llbracket z <_y^x M \rrbracket_\rho = Fv(N) <_{Fv(N_2)}^{Fv(N_1)} \llbracket M \rrbracket_{\rho(N_1/x, N_2/y)}$
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 $Fv(\lambda x.M) = \{x_1, \dots, x_n\}$, then
 $\llbracket \lambda x.M \rrbracket_\rho N = (\lambda x.M)[\rho(x_1)/x_1, \dots, \rho(x_n)/x_n]N \rightarrow$
 $(M[\rho(x_1)/x_1, \dots, \rho(x_n)/x_n])[N/x] = M[\rho(x_1)/x_1, \dots, \rho(x_n)/x_n, N/x] =$
 $\llbracket M \rrbracket_{\rho(N/x)},$
- (iii) If $Fv(M) = \{x_1, \dots, x_n\}$, then
 $\llbracket x \odot M \rrbracket_\rho = (x \odot M)[\rho(x)/x, \rho(x_1)/x_1, \dots, \rho(x_n)/x_n] =$
 $Fv(\rho(x)) \odot M[\rho(x_1)/x_1, \dots, \rho(x_n)/x_n] = Fv(\rho(x)) \odot \llbracket M \rrbracket_\rho.$
- (iv) If $Fv(M) = \{x_1, \dots, x_n\}$, then
 $\llbracket z <_y^x M \rrbracket_\rho = (z <_y^x M)[N/z, \rho(x_1)/x_1, \dots, \rho(x_n)/x_n] =$
 $Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M[N_1/x, N_2/y, \rho(x_1)/x_1, \dots, \rho(x_n)/x_n] =$
 $= Fv(N) <_{Fv(N_2)}^{Fv(N_1)} \llbracket M \rrbracket_{\rho(N_1/x, N_2/y)}.$

□

Definition 19.

- (i) $\rho \models M : \sigma \iff \llbracket M \rrbracket_\rho \in \llbracket \sigma \rrbracket;$
- (ii) $\rho \models \Gamma \iff (\forall (x : \alpha) \in \Gamma) \rho(x) \in \llbracket \alpha \rrbracket;$
- (iii) $\Gamma \models M : \sigma \iff (\forall \rho, \rho \models \Gamma \Rightarrow \rho \models M : \sigma).$

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

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

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

Proposition 21 (Soundness of $\lambda_{\mathbb{R}} \sqcap$). *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 \mathbb{R} -saturated, we obtain $[[\lambda x.M]]_{\rho} N \in [[\sigma]]$.

- The last rule applied is (\rightarrow_E) , i.e.

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

Let ρ be any valuation.

Assuming that $\Gamma \vdash M : \cap_i^n \tau_i \rightarrow \sigma, \Delta_1 \vdash N : \tau_1, \dots, \Delta_n \vdash N : \tau_n$, we have to prove that if $\rho \models \Gamma, \Delta_0^{\top} \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$, then $\rho \models MN : \sigma$.

By IH, $\Gamma \models M : \cap_i^n \tau_i \rightarrow \sigma$ and $\Delta_0 \models N : \tau_0, \dots, \Delta_n \models N : \tau_n$. Assume that $\rho \models \Gamma, \Delta_0^{\top} \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$. This means that $\rho \models \Gamma$ and $\rho \models \Delta_0^{\top} \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$. 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 $[[M]]_{\rho} \in \cap_i^n [[\tau_i]] \longrightarrow [[\sigma]]$. Using Lemma 20 $\rho \models \Delta_0^{\top} \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$ implies $(\rho \models \Delta_0^{\top}) \wedge (\bigwedge_{i=1}^n \rho \models \Delta_i)$, hence by Definition 19 (i) and (iii) we get $([[N]]_{\rho} \in [[\tau]]) \wedge \bigwedge_{i=1}^n ([N]_{\rho} \in [[\tau_i]])$, 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) $\rho \models MN : \sigma$.

- 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 $\llbracket M \rrbracket_\rho \in \llbracket \sigma \rrbracket$. Using multiple times the weakening property WEAK and Lemma 18(iii) we obtain $Fv(\rho(x)) \odot \llbracket M \rrbracket_\rho = \llbracket x \odot M \rrbracket_\rho \in \llbracket \sigma \rrbracket$, since $Fv(\rho(x)) \cap Fv(\llbracket M \rrbracket_\rho) = \emptyset$.

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

$$\Gamma, x : \alpha, y : \beta \vdash M : \sigma \Rightarrow \Gamma, z : \alpha \cap \beta \vdash z <_y^x M : \sigma.$$

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 $\rho \models z : \alpha \cap \beta \Leftrightarrow \rho(z) \in \llbracket \alpha \rrbracket$ and $\rho(z) \in \llbracket \beta \rrbracket$. For the sake of simplicity let $\rho(z) \equiv N$. We define a new valuation ρ' 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 $x, y \notin \text{Dom}(\Gamma)$, $N_1 \in \llbracket \alpha \rrbracket$ and $N_2 \in \llbracket \beta \rrbracket$. By the IH $\llbracket M \rrbracket_{\rho'} = \llbracket M \rrbracket_{\rho(N_1/x, N_2/y)} \in \llbracket \sigma \rrbracket$. Using the contraction property CONT we have that $Fv(N) <_{Fv(N_2)}^{Fv(N_1)} \llbracket M \rrbracket_{\rho(N_1/x, N_2/y)} = \llbracket z <_y^x M \rrbracket_\rho \in \llbracket \sigma \rrbracket$.

□

Theorem 22 (\mathcal{SN} for $\lambda_{\mathbb{R}} \cap$). *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 $(\forall \rho \models \Gamma) \rho \models M : \sigma$. We can choose a particular $\rho_0(x) = x$ for all $x \in \text{var}$. By Proposition 16(iv), $\llbracket \beta \rrbracket$ is \mathbb{R} -saturated for each type β , hence $x = \llbracket x \rrbracket_{\rho_0} \in \llbracket \beta \rrbracket$ (variable condition for $n = 0$). Therefore, $\rho_0 \models \Gamma$ and we can conclude that $\llbracket M \rrbracket_{\rho_0} \in \llbracket \sigma \rrbracket$. On the other hand, $M = \llbracket M \rrbracket_{\rho_0}$ and $\llbracket \sigma \rrbracket \subseteq \mathcal{SN}$ (Proposition 16), hence $M \in \mathcal{SN}$. □

1.4. $SN \Rightarrow$ Typeability in $\lambda_{\mathbb{R}} \cap$

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

$$\begin{aligned} M_{nf} &::= x \mid \lambda x. M_{nf} \mid \lambda x. x \odot M_{nf} \mid x M_{nf}^1 \dots M_{nf}^n \mid \\ &\quad x <_{x_2}^{x_1} M_{nf} N_{nf}, \text{ with } x_1 \in Fv(M_{nf}), x_2 \in Fv(N_{nf}) \\ W_{nf} &::= x \odot M_{nf} \mid x \odot W_{nf} \end{aligned}$$

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 $\lambda x.y \odot M_{nf} \rightarrow_{\omega_1} y \odot \lambda x.M_{nf}$.

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

Proof. By induction on the structure of M_{nf} and W_{nf} . \square

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

Proof. By induction on the structure of M . \square

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

Proof. By case study according to the applied reduction. \square

Theorem 26 (SN \Rightarrow typeability). *All strongly normalising $\lambda_{\mathbb{R}}$ -terms are typeable in the $\lambda_{\mathbb{R}} \cap$ 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 $M \equiv (\lambda x.N)P \rightarrow_\beta N[P/x] \equiv M'$, 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 $x <_{x_2}^{x_1} NP$, $x_1 \in Fv(N)$, $x_2 \in Fv(P)$ (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 $x <_{x_2}^{x_1} NP$ with $x_1 \in Fv(N)$, $x_2 \in Fv(P)$. By induction NP is typeable, hence N is typeable with say $\Gamma, x_1 : \beta \vdash N : \cap_i^n \tau_i \rightarrow \sigma$ 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 $(E \rightarrow)$ 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 $\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n, x : \beta \cap (\cap_i^n \gamma_i) \vdash x <_{x_2}^{x_1} NP : \sigma$.

□

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

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

Proof. Immediate consequence of Theorems 22 and 26. □

2. Intersection types for the sequent resource control lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$

In this section we focus on the sequent resource control lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$. 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 $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions.

2.1. Resource control sequent lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$

The *resource control lambda Gentzen* calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$ is derived from the $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus (more precisely its confluent sub-calculus $\lambda_{\mathbb{V}}^{\text{Gtz}}$) by adding the explicit operators for weakening and contraction. It is proposed in [28]. The abstract syntax of $\lambda_{\mathbb{R}}^{\text{Gtz}}$ pre-expressions is the following:

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

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 $\lambda_{\mathbb{R}}$ -calculus with the following additions:

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

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

$$\begin{array}{c}
\frac{}{x \in \Lambda_{\mathbb{R}}^{\text{Gtz}}} \quad \frac{f \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \quad x \in Fv(f)}{\lambda x. f \in \Lambda_{\mathbb{R}}^{\text{Gtz}}} \\
\\
\frac{f \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \quad c \in \Lambda_{\mathbb{R},C}^{\text{Gtz}} \quad Fv(f) \cap Fv(c) = \emptyset}{fc \in \Lambda_{\mathbb{R}}^{\text{Gtz}}} \\
\\
\frac{f \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \quad x \in Fv(f)}{\hat{x}.f \in \Lambda_{\mathbb{R},C}^{\text{Gtz}}} \quad \frac{f \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \quad c \in \Lambda_{\mathbb{R},C}^{\text{Gtz}} \quad Fv(f) \cap Fv(c) = \emptyset}{f :: c \in \Lambda_{\mathbb{R},C}^{\text{Gtz}}} \\
\\
\frac{E \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}} \quad x \notin Fv(E)}{x \odot E \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}}} \\
\\
\frac{E \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}} \quad x_1 \neq x_2 \quad x_1, x_2 \in Fv(E) \quad x \notin Fv(E) \setminus \{x_1, x_2\}}{x <_{x_2}^{x_1} E \in \Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}}}
\end{array}$$

Figure 6: $\Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}}$: $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions

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

The first group consists of β_g , π , σ and μ reductions from the λ^{Gtz} . New reductions are added to deal with explicit contraction (γ reductions) and weakening (ω reductions). The groups of γ and ω 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 $\lambda_{\mathbb{R}}^{\text{Gtz}}$ are the ones given in Figure 4, except for the fact that they refer to $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions.

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

(β_g)	$(\lambda x.t)(u :: k) \rightarrow u(\hat{x}.tk)$
(σ)	$T(\hat{x}.v) \rightarrow v[T/x]$
(π)	$(tk)k' \rightarrow t(k@k')$
(μ)	$\hat{x}.xk \rightarrow k$
(γ_1)	$x <_{x_2}^{x_1} (\lambda y.t) \rightarrow \lambda y.x <_{x_2}^{x_1} t$
(γ_2)	$x <_{x_2}^{x_1} (tk) \rightarrow (x <_{x_2}^{x_1} t)k, \text{ if } x_1, x_2 \notin Fv(k)$
(γ_3)	$x <_{x_2}^{x_1} (tk) \rightarrow t(x <_{x_2}^{x_1} k), \text{ if } x_1, x_2 \notin Fv(t)$
(γ_4)	$x <_{x_2}^{x_1} (\hat{y}.t) \rightarrow \hat{y}.(x <_{x_2}^{x_1} t)$
(γ_5)	$x <_{x_2}^{x_1} (t :: k) \rightarrow (x <_{x_2}^{x_1} t) :: k, \text{ if } x_1, x_2 \notin Fv(k)$
(γ_6)	$x <_{x_2}^{x_1} (t :: k) \rightarrow t :: (x <_{x_2}^{x_1} k), \text{ if } x_1, x_2 \notin Fv(t)$
(ω_1)	$\lambda x.(y \odot t) \rightarrow y \odot (\lambda x.t), \quad x \neq y$
(ω_2)	$(x \odot t)k \rightarrow x \odot (tk)$
(ω_3)	$t(x \odot k) \rightarrow x \odot (tk)$
(ω_4)	$\hat{x}.(y \odot t) \rightarrow y \odot (\hat{x}.t), \quad x \neq y$
(ω_5)	$(x \odot t) :: k \rightarrow x \odot (t :: k)$
(ω_6)	$t :: (x \odot k) \rightarrow x \odot (t :: k)$
$(\gamma\omega_1)$	$x <_{x_2}^{x_1} (y \odot e) \rightarrow y \odot (x <_{x_2}^{x_1} e) \quad x_1 \neq y \neq x_2$
$(\gamma\omega_2)$	$x <_{x_2}^{x_1} (x_1 \odot e) \rightarrow e[x/x_2]$

Figure 7: Reduction rules of $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus

tions:

$$\begin{aligned}
(tk)[u/x] &= t[u/x]k, \quad x \notin Fv(k) & (tk)[u/x] &= tk[u/x], \quad x \notin Fv(t) \\
(\hat{y}.t)[u/x] &= \hat{y}.t[u/x] \\
(t :: k)[u/x] &= t[u/x] :: k, \quad x \notin Fv(k) & (t :: k)[u/x] &= t :: k[u/x], \quad x \notin Fv(t)
\end{aligned}$$

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

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

2.2. Intersection types for $\lambda_{\mathbb{R}}^{\text{Gtz}}$

The type assignment system $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ that assigns strict types to $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions is given in Figure 8. Due to the sequent flavour of the $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -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 $\lambda_{\mathbb{R}} \cap$. The $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ 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 (\rightarrow_L). In these two rules $Dom(\Gamma_1) = \dots = Dom(\Gamma_n)$. The role of Γ_0^\top has been already explained in subsection 1.2.

$$\begin{array}{c}
\frac{}{x : \sigma \vdash x : \sigma} (Ax) \\
\\
\frac{\Gamma, x : \alpha \vdash t : \sigma}{\Gamma \vdash \lambda x. t : \alpha \rightarrow \sigma} (\rightarrow_R) \quad \frac{\Gamma, x : \alpha \vdash t : \sigma}{\Gamma; \alpha \vdash \hat{x}. t : \sigma} (Sel) \\
\\
\frac{\Gamma_0 \vdash t : \sigma_0 \quad \dots \quad \Gamma_n \vdash t : \sigma_n \quad \Delta; \cap_j^m \tau_j \vdash k : \rho}{\Gamma_0^\top \cap \Gamma_1 \cap \dots \cap \Gamma_n, \Delta; \cap_j^m (\cap_i^n \sigma_i \rightarrow \tau_j) \vdash t :: k : \rho} (\rightarrow_L) \\
\\
\frac{\Gamma_0 \vdash t : \sigma_0 \quad \dots \quad \Gamma_n \vdash t : \sigma_n \quad \Delta; \cap_i^n \sigma_i \vdash k : \tau}{\Gamma_0^\top \cap \Gamma_1 \cap \dots \cap \Gamma_n, \Delta \vdash tk : \tau} (Cut) \\
\\
\frac{\Gamma, x : \alpha, y : \beta \vdash t : \sigma}{\Gamma, z : \alpha \cap \beta \vdash z <_y^x t : \sigma} (Cont_t) \quad \frac{\Gamma \vdash t : \sigma}{\Gamma, x : \top \vdash x \odot t : \sigma} (Weak_t) \\
\\
\frac{\Gamma, x : \alpha, y : \beta; \gamma \vdash k : \sigma}{\Gamma, z : \alpha \cap \beta; \gamma \vdash z <_y^x k : \sigma} (Cont_k) \quad \frac{\Gamma; \gamma \vdash k : \sigma}{\Gamma, x : \top; \gamma \vdash x \odot k : \sigma} (Weak_k)
\end{array}$$

Figure 8: $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$: $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus with intersection types

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

Lemma 28 (Generation lemma for $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$).

- (i) $\Gamma \vdash \lambda x.t : \tau$ iff there exist α and σ such that $\tau \equiv \alpha \rightarrow \sigma$ and $\Gamma, x : \alpha \vdash t : \sigma$.
- (ii) $\Gamma; \gamma \vdash t :: k : \rho$ iff $\Gamma = \Gamma'_0 \top \Gamma'_1 \sqcap \dots \sqcap \Gamma'_n, \Delta$, $\gamma \equiv \cap_j^m (\cap_i^n \sigma_i \rightarrow \tau_j)$, $\Delta; \cap_j^m \tau_j \vdash k : \rho$ and $\Gamma'_l \vdash t : \sigma_l$ for all $l \in \{0, \dots, n\}$.
- (iii) $\Gamma \vdash tk : \sigma$ iff $\Gamma = \Gamma'_0 \top \Gamma'_1 \sqcap \dots \sqcap \Gamma'_n, \Delta$, and there exist $\tau_j, j = 0, \dots, n$ such that for all $j \in \{0, \dots, n\}$ the following holds: $\Gamma'_j \vdash t : \tau_j$, and $\Delta; \cap_i^n \tau_i \vdash k : \sigma$.
- (iv) $\Gamma; \alpha \vdash \hat{x}.t : \sigma$ iff $\Gamma, x : \alpha \vdash t : \sigma$.
- (v) $\Gamma \vdash z <_y^x t : \sigma$ iff there exist Γ', α, β 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) $\Gamma; \gamma \vdash z <_y^x k : \sigma$ iff there exist Γ', α, β 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 $\text{Dom}(\Gamma) = \text{Fv}(t)$.
- (ii) If $\Gamma; \alpha \vdash k : \sigma$, then $\text{Dom}(\Gamma) = \text{Fv}(k)$.

Proof. Similar to the proof of Lemma 8. □

Lemma 30 (Substitution lemma for $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$).

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

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 $j = 0, \dots, n$, and $\Delta; \cap_i^n \tau_i \vdash k' : \sigma$, then $\Gamma_0 \top \Gamma_1 \sqcap \dots \sqcap \Gamma_n, \Delta; \alpha \vdash k @ k' : \sigma$.

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

Proposition 32 (Subject equivalence for $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$).

- (i) For every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -term t : if $\Gamma \vdash t : \sigma$ and $t \equiv_{\lambda_{\mathbb{R}}^{\text{Gtz}}} t'$, then $\Gamma \vdash t' : \sigma$.
- (ii) For every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -context k : if $\Gamma; \alpha \vdash k : \sigma$ and $k \equiv_{\lambda_{\mathbb{R}}^{\text{Gtz}}} k'$, then $\Gamma; \alpha \vdash k' : \sigma$.

Proof. By case analysis on the applied equivalence. \square

Proposition 33 (Subject reduction for $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$).

- (i) For every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -term t : if $\Gamma \vdash t : \sigma$ and $t \rightarrow t'$, then $\Gamma \vdash t' : \sigma$.
- (ii) For every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -context k : if $\Gamma; \alpha \vdash k : \sigma$ and $k \rightarrow k'$, then $\Gamma; \alpha \vdash k' : \sigma$.

Proof. By case analysis on the applied reduction, using Lemmas 30 and 31 for the cases of (σ) and (π) rule, respectively. \square

2.3. Typeability \Rightarrow SN in $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$

In this section, we prove the strong normalisation of the $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus with intersection types. The termination is proved by showing that the reduction on the set $\Lambda_{\mathbb{R}}^{\text{Gtz}} \cup \Lambda_{\mathbb{R},C}^{\text{Gtz}}$ of the typeable $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -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 $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions into $\lambda_{\mathbb{R}}$ -terms. We show that this mapping preserves types and that every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -reduction can be simulated either by a $\lambda_{\mathbb{R}}$ -reduction or by an equality and each $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -equivalence can be simulated by an $\lambda_{\mathbb{R}}$ -equivalence. The other two well-founded orders are based on the introduction of quantities designed to decrease a global measure associated with specific $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions during the computation.

Definition 34. The mapping $\lfloor \cdot \rfloor : \Lambda_{\mathbb{R}}^{\text{Gtz}} \rightarrow \Lambda_{\mathbb{R}}$ is defined together with the auxiliary mapping $\lfloor \cdot \rfloor_k : \Lambda_{\mathbb{R},C}^{\text{Gtz}} \rightarrow (\Lambda_{\mathbb{R}} \rightarrow \Lambda_{\mathbb{R}})$ in the following way:

$$\begin{array}{ll}
\lfloor x \rfloor &= x & \lfloor \widehat{x}.t \rfloor_k(M) &= (\lambda x. \lfloor t \rfloor)M \\
\lfloor \lambda x. t \rfloor &= \lambda x. \lfloor t \rfloor & \lfloor t :: k \rfloor_k(M) &= \lfloor k \rfloor_k(M \lfloor t \rfloor) \\
\lfloor x \odot t \rfloor &= x \odot \lfloor t \rfloor & \lfloor x \odot k \rfloor_k(M) &= x \odot \lfloor k \rfloor_k(M) \\
\lfloor x <_z^y t \rfloor &= x <_z^y \lfloor t \rfloor & \lfloor x <_z^y k \rfloor_k(M) &= x <_z^y \lfloor k \rfloor_k(M) \\
\lfloor tk \rfloor &= \lfloor k \rfloor_k(\lfloor t \rfloor)
\end{array}$$

Lemma 35.

- (i) $Fv(t) = Fv(\lfloor t \rfloor)$, for $t \in \Lambda_{\mathbb{R}}^{\text{Gtz}}$.

(ii) $\lfloor v[t/x] \rfloor = \lfloor v \rfloor [\lfloor t \rfloor / x]$, for $v, t \in \Lambda_{\mathbb{R}}^{\text{Gtz}}$.

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

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

- (i) If $\Gamma \vdash t : \sigma$, then $\Gamma \vdash_{\lambda_{\mathbb{R}}} \lfloor t \rfloor : \sigma$.
- (ii) If $\Gamma; \cap_i^n \tau_i \vdash k : \sigma$, then $\lfloor k \rfloor_k : \Lambda_{\mathbb{R}}(\Delta_j \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma, \Delta \vdash_{\lambda_{\mathbb{R}}} \sigma)$, for all $j \in \{0, \dots, n\}$ and for some $\Delta = \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$.

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

- Cases (Ax) , (\rightarrow_R) , $(Weak_t)$ and $(Cont_t)$ are easy, because the intersection type assignment system of $\lambda_{\mathbb{R}}$ 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 $\Gamma, x : \alpha \vdash_{\lambda_{\mathbb{R}}} \lfloor t \rfloor : \sigma$, where $\alpha = \cap_i^n \tau_i$. For any $M \in \Lambda_{\mathbb{R}}$ such that $\Delta_j \vdash_{\lambda_{\mathbb{R}}} M : \tau_j$, for all $j \in \{0, \dots, n\}$, we have

$$\frac{\frac{\Gamma, x : \cap_i^n \tau_i \vdash_{\lambda_{\mathbb{R}}} \lfloor t \rfloor : \sigma}{\Gamma \vdash_{\lambda_{\mathbb{R}}} \lambda x. \lfloor t \rfloor : \cap_i^n \tau_i \rightarrow \sigma} (\rightarrow_I) \quad \Delta_0 \vdash_{\lambda_{\mathbb{R}}} M : \tau_0 \dots \Delta_n \vdash_{\lambda_{\mathbb{R}}} M : \tau_n}{\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash_{\lambda_{\mathbb{R}}} (\lambda x. \lfloor t \rfloor) M : \sigma} (\rightarrow_E)$$

Since $(\lambda x. \lfloor t \rfloor) M = \lfloor \widehat{x}.t \rfloor_k(M)$, we conclude that $\lfloor \widehat{x}.t \rfloor_k : \Lambda_{\mathbb{R}}(\Delta_j \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash_{\lambda_{\mathbb{R}}} \sigma)$.

- Case (\rightarrow_L) : the derivation ends with the rule

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

for $\Gamma = \Gamma_0^\top \sqcap \Gamma_1 \sqcap \dots \sqcap \Gamma_n$. By IH we have that $\Gamma_l \vdash_{\lambda_{\mathbb{R}}} [t] : \sigma_l$, for $l \in \{0, \dots, n\}$. For any $M \in \Lambda_{\mathbb{R}}$ such that $\Gamma'_j \vdash_{\Lambda_{\mathbb{R}}} M : \cap_i^n \sigma_i \rightarrow \tau_j$, $j = 1, \dots, m$ we have

$$\frac{\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} M : \cap_i^n \sigma_i \rightarrow \tau_j \quad \Gamma_0 \vdash_{\lambda_{\mathbb{R}}} [t] : \sigma_0 \dots \Gamma_n \vdash_{\lambda_{\mathbb{R}}} [t] : \sigma_n}{\Gamma_0^\top \sqcap \Gamma_1 \sqcap \dots \sqcap \Gamma_n, \Gamma'_j \vdash_{\lambda_{\mathbb{R}}} M[t] : \tau_j} (\rightarrow_E)$$

From the right-hand side premise in the (\rightarrow_L) rule, by IH, we get that $[k]_k$ is the function with the scope $[k]_k : \Lambda_{\mathbb{R}}(\Gamma_j''' \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma''', \Gamma'' \vdash_{\lambda_{\mathbb{R}}} \rho)$, for some $\Gamma''' = \Gamma_0'''^\top \sqcap \Gamma_1''' \sqcap \dots \sqcap \Gamma_n'''$. For $\Gamma''' \equiv \Gamma, \Gamma'$ and by taking $M[t]$ as the argument of the function $[k]_k$, we get $\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\mathbb{R}}} [k]_k(M[t]) : \rho$. Since $[k]_k(M[t]) = [t :: k]_k(M)$, we have that $\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\mathbb{R}}} [t :: k]_k(M) : \rho$. This holds for any M of the appropriate type, yielding

$[t :: k]_k : \Lambda_{\mathbb{R}}(\Gamma' \vdash_{\lambda_{\mathbb{R}}} \cap_i^n \sigma_i \rightarrow \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma, \Delta, \Gamma' \vdash_{\lambda_{\mathbb{R}}} \rho)$, which is exactly what we need.
Case (*Cut*): the derivation ends with the rule

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

By IH we have that $\Gamma_j \vdash_{\lambda_{\mathbb{R}}} [t] : \tau_j$ and $[k]_k : \Lambda_{\mathbb{R}}(\Gamma_j' \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma', \Delta \vdash_{\lambda_{\mathbb{R}}} \sigma)$ for all $j = 0, \dots, n$ and for $\Gamma' = \Gamma_0^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n'$. Hence, for any $M \in \Lambda_{\mathbb{R}}$ such that $\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} M : \tau_j$, $\Gamma', \Delta \vdash_{\lambda_{\mathbb{R}}} [k]_k(M) : \sigma$ holds. By taking $M \equiv [t]$ and $\Gamma' \equiv \Gamma$, we get $\Gamma, \Delta \vdash_{\lambda_{\mathbb{R}}} [k]_k([t]) : \sigma$. But $[k]_k([t]) = [tk]$, 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 $[k]_k$ is the function with the scope $[k]_k : \Lambda_{\mathbb{R}}(\Gamma_j' \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma, \Gamma_0'^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n' \vdash_{\lambda_{\mathbb{R}}} \sigma)$, meaning that for each $M \in \Lambda_{\mathbb{R}}$ such that $\Gamma'_j \vdash_{\lambda_{\mathbb{R}}} M : \tau_j$ for all $j \in \{0, \dots, n\}$ holds $\Gamma_0'^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n', \Gamma \vdash_{\lambda_{\mathbb{R}}} [k]_k(M) : \sigma$. Now, we can apply (*Weak*) rule:

$$\frac{\Gamma, \Gamma_0'^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n' \vdash [k]_k(M) : \sigma}{\Gamma, \Gamma_0'^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n', x : \top \vdash x \odot [k]_k(M) : \sigma} (Weak)$$

Since $x \odot [k]_k(M) = [x \odot k]_k(M)$, this means that $[x \odot k]_k : \Lambda_{\mathbb{R}}(\Gamma_j' \vdash_{\lambda_{\mathbb{R}}} \tau_j) \rightarrow \Lambda_{\mathbb{R}}(\Gamma, \Gamma_0'^\top \sqcap \Gamma_1' \sqcap \dots \sqcap \Gamma_n', x : \top \vdash_{\lambda_{\mathbb{R}}} \sigma)$, 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 $\lfloor \cdot \rfloor$, we show that each $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -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 [22], use Regnier's σ reductions, investigated in [53].

$$\begin{aligned} ((\lambda x.M)N)P &\rightarrow (\lambda x.(MN))P \quad x \notin P \\ (\lambda xy.M)N &\rightarrow \lambda y.((\lambda x.M)N) \quad y \notin N \\ M((\lambda x.P)N) &\rightarrow (\lambda x.MP)N \quad x \notin M \end{aligned}$$

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

Lemma 38. $\lfloor k \rfloor_k((\lambda x.P)N) \rightarrow_{\lambda_{\mathbb{R}}} (\lambda x.\lfloor k \rfloor_k(P))N$.

Lemma 39. *If $M \in \Lambda^{\mathbb{R}}$ and $k, k' \in \Lambda_{\mathbb{R},C}^{\text{Gtz}}$, then $\lfloor k' \rfloor_k \circ \lfloor k \rfloor_k(M) \rightarrow_{\lambda_{\mathbb{R}}} \lfloor k@k' \rfloor_k(M)$.*

Lemma 40.

- (i) *If $x \notin Fv(k)$, then $(\lfloor k \rfloor_k(M))[N/x] = \lfloor k \rfloor_k(M[N/x])$.*
- (ii) *If $x, y \notin Fv(k)$, then $z <_y^x (\lfloor k \rfloor_k(M)) \rightarrow_{\lambda_{\mathbb{R}}} \lfloor k \rfloor_k(z <_y^x M)$.*
- (iii) $\lfloor k \rfloor_k(x \odot M) \rightarrow_{\lambda_{\mathbb{R}}} x \odot \lfloor k \rfloor_k(M)$.

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

Theorem 41 (Simulation of $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -reduction by $\lambda_{\mathbb{R}}$ -reduction).

- (i) *If a term $t \rightarrow_{\lambda_{\mathbb{R}}^{\text{Gtz}}} t'$, then $\lfloor t \rfloor \rightarrow_{\lambda_{\mathbb{R}}} \lfloor t' \rfloor$.*
- (ii) *If a context $k \rightarrow_{\lambda_{\mathbb{R}}^{\text{Gtz}}} k'$ by γ_6 or ω_6 reduction, then $\lfloor k \rfloor_k(M) \equiv \lfloor k' \rfloor_k(M)$, for any $M \in \Lambda^{\mathbb{R}}$.*
- (iii) *If a context $k \rightarrow_{\lambda_{\mathbb{R}}^{\text{Gtz}}} k'$ by some other reduction, then $\lfloor k \rfloor_k(M) \rightarrow_{\lambda_{\mathbb{R}}} \lfloor k' \rfloor_k(M)$, for any $M \in \Lambda^{\mathbb{R}}$.*

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

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

The previous proposition shows that $\beta_g, \pi, \sigma, \mu, \gamma_1 - \gamma_5, \omega_1 - \omega_5, \gamma\omega_1$ and $\gamma\omega_2$ $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -reductions are interpreted by $\lambda_{\mathbb{R}}$ -reductions and that γ_6 and ω_6 $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -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 $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -reductions one has to prove that there cannot exist an infinite sequence of $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -reductions which are all interpreted as equalities. To prove this, one shows that if a term is reduced with such a $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -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 $\mathcal{S}(), \| \cdot \|_c, \| \cdot \|_w : \Lambda_{\mathbb{R}}^{\text{Gtz}} \rightarrow \mathbb{N}$ are defined as follows:

$$\begin{aligned} \mathcal{S}(x) &= 1 & \mathcal{S}(tk) &= \mathcal{S}(t) + \mathcal{S}(k) \\ \mathcal{S}(\lambda x.t) &= 1 + \mathcal{S}(t) & \mathcal{S}(\hat{x}.t) &= 1 + \mathcal{S}(t) \\ \mathcal{S}(x \odot e) &= 1 + \mathcal{S}(e) & \mathcal{S}(t :: k) &= \mathcal{S}(t) + \mathcal{S}(k) \\ \mathcal{S}(x <_z^y e) &= 1 + \mathcal{S}(e) \end{aligned}$$

$$\begin{aligned} \|x\|_c &= 0 & \|x\|_w &= 1 \\ \|\lambda x.t\|_c &= \|t\|_c & \|\lambda x.t\|_w &= 1 + \|t\|_w \\ \|x \odot e\|_c &= \|e\|_c & \|x \odot e\|_w &= 0 \\ \|x <_z^y e\|_c &= \|e\|_c + \mathcal{S}(e) & \|x <_z^y e\|_w &= 1 + \|e\|_w \\ \|tk\|_c &= \|t\|_c + \|k\|_c & \|tk\|_w &= 1 + \|t\|_w + \|k\|_w \\ \|\hat{x}.t\|_c &= \|t\|_c & \|\hat{x}.t\|_w &= 1 + \|t\|_w \\ \|t :: k\|_c &= \|t\|_c + \|k\|_c & \|t :: k\|_w &= 1 + \|t\|_w + \|k\|_w \end{aligned}$$

Lemma 43. For all $e, e' \in \Lambda_{\mathbb{R}}^{\text{Gtz}}$:

- (i) If $e \rightarrow_{\gamma_6} e'$, then $\|e\|_c > \|e'\|_c$.
- (ii) If $e \rightarrow_{\omega_6} e'$, then $\|e\|_c = \|e'\|_c$.
- (iii) If $e \equiv_{\lambda_{\mathbb{R}}^{\text{Gtz}}} e'$, then $\|e\|_c = \|e'\|_c$.

Lemma 44.

- (i) For all $e, e' \in \Lambda_{\mathbb{R}}^{\text{Gtz}}$: If $e \rightarrow_{\omega_6} e'$, then $\|e\|_w > \|e'\|_w$.
- (ii) If $e \equiv_{\lambda_{\mathbb{R}}^{\text{Gtz}}} e'$, then $\|e\|_w = \|e'\|_w$.

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 $\Lambda_{\mathbb{R}}^{\text{Gtz}}$:

- (i) $t >_{\lambda_{\mathbb{R}}} t'$ iff $\lfloor t \rfloor \rightarrow_{\lambda_{\mathbb{R}}}^+ \lfloor t' \rfloor$; $t =_{\lambda_{\mathbb{R}}} t'$ iff $\lfloor t \rfloor \equiv_{\lambda_{\mathbb{R}}} \lfloor t' \rfloor$
 $k >_{\lambda_{\mathbb{R}}} k'$ iff $\lfloor k \rfloor_k(M) \rightarrow_{\lambda_{\mathbb{R}}}^+ \lfloor k' \rfloor(M)$ for every $\lambda_{\mathbb{R}}$ term M ;
 $k =_{\lambda_{\mathbb{R}}} k'$ iff $\lfloor k \rfloor_k(M) \equiv_{\lambda_{\mathbb{R}}} \lfloor k' \rfloor_k(M)$ or $\lfloor k \rfloor_k(M) \equiv \lfloor k' \rfloor(M)$ for every $\lambda_{\mathbb{R}}$ term M ;
- (ii) $e >_c e'$ iff $\|e\|_c > \|e'\|_c$; $e =_c e'$ iff $\|e\|_c = \|e'\|_c$;
- (iii) $e >_w e'$ iff $\|e\|_w > \|e'\|_w$; $e =_w e'$ iff $\|e\|_w = \|e'\|_w$;

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 $\Lambda_{\mathbb{R}}^{\text{Gtz}}$ as the lexicographic product:

$$\gg = >_{\lambda_{\mathbb{R}}} \times_{lex} >_c \times_{lex} >_w.$$

The following propositions proves that the reduction relation on the set of typed $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expressions is included in the given lexicographic product \gg .

Proposition 47. For each $e \in \Lambda_{\mathbb{R}}^{\text{Gtz}}$: if $e \rightarrow e'$, then $e \gg e'$.

Proof. By case analysis on the kind of reduction and the structure of \gg .

If $e \rightarrow e'$ by $\beta_g, \sigma, \pi, \mu, \gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma\omega_1, \gamma\omega_2, \omega_1, \omega_2, \omega_3, \omega_4$ or ω_5 reduction, then $e >_{\lambda_{\mathbb{R}}} e'$ by Proposition 41.

If $e \rightarrow e'$ by γ_6 , then $e =_{\lambda_{\mathbb{R}}} e'$ by Proposition 41, and $e >_c e'$ by Lemma 43.

Finally, if $e \rightarrow e'$ by ω_6 , then $e =_{\lambda_{\mathbb{R}}} e'$ by Proposition 41, $e =_c e'$ by Lemma 43 and $e >_w e'$ by Lemma 44. \square

Strong normalisation of \rightarrow is another terminology for the well-foundedness of the relation \rightarrow 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 $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$). *Each expression in $\Lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ is strongly normalising.*

Proof. The reduction \rightarrow is well-founded on $\Lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ as it is included (Proposition 47) in the relation \gg which is well-founded as the lexicographic product of the well-founded relations $>_{\lambda_{\mathbb{R}}}$, $>_c$ and $>_w$. Relation $>_{\lambda_{\mathbb{R}}}$ is based on the interpretation $\lfloor \cdot \rfloor : \Lambda_{\mathbb{R}}^{\text{Gtz}} \rightarrow \Lambda_{\mathbb{R}}$. By Proposition 36 typeability is preserved by the interpretation $\lfloor \cdot \rfloor$ and $\rightarrow_{\lambda_{\mathbb{R}}}$ is strongly normalising (i.e., well-founded) on $\Lambda_{\mathbb{R}} \cap$ (Section 1.3), hence $>_{\lambda_{\mathbb{R}}}$ is well-founded on $\Lambda_{\mathbb{R}}^{\text{Gtz}} \cap$. 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. \square

2.4. $\text{SN} \Rightarrow \text{Typeability in } \lambda_{\mathbb{R}}^{\text{Gtz}} \cap$

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

The abstract syntax of $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -normal forms is the following:

$$\begin{aligned} t_{nf} &::= x \mid \lambda x. t_{nf} \mid \lambda x. x \odot t_{nf} \mid x(t_{nf} :: k_{nf}) \mid x <_z^y y(t_{nf} :: k_{nf}) \\ k_{nf} &::= \hat{x}. t_{nf} \mid \hat{x}. x \odot t_{nf} \mid t_{nf} :: k_{nf} \mid x <_z^y (t_{nf} :: k_{nf}), y \in Fv(t_{nf}), z \in Fv(k_{nf}) \\ w_{nf} &::= x \odot e_{nf} \mid x \odot w_{nf} \end{aligned}$$

We use e_{nf} for any $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -expression in the normal form.

Proposition 49. $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -normal forms are typeable in the system $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$.

Proof. By mutual induction on the structure of t_{nf} , k_{nf} and w_{nf} . \square

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 Δ_j and τ_j , $j = 0, \dots, n$ such that $\Delta_j \vdash u : \tau_j$ and $\Gamma', x : \cap_i^n \tau_i \vdash t : \sigma$, where $\Gamma = \Gamma', \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n$.

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

Proof. By mutual induction on the structure of terms and contexts. \square

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

Proof. By induction on the structure of the context k . \square

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

Proposition 52 (Head subject expansion). *For every $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -term t : if $t \rightarrow t'$, t is contracted redex and $\Gamma \vdash t' : \sigma$, then $\Gamma \vdash t : \sigma$.*

Proof. By case study according to the applied reduction. \square

Theorem 53 (SN \Rightarrow typeability). *All strongly normalising $\lambda_{\mathbb{R}}^{\text{Gtz}}$ terms are typeable in the $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$ system.*

Proof. Analogous to the proof of Theorem 26. \square

Now we give a characterisation of strong normalisation in $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus.

Theorem 54. *In $\lambda_{\mathbb{R}}^{\text{Gtz}}$ -calculus, the term t is strongly normalising if and only if it is typeable in $\lambda_{\mathbb{R}}^{\text{Gtz}} \cap$.*

Proof. Immediate consequence of Theorems 48 and 53. \square

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

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

The *resource control* lambda calculus with explicit substitution $\lambda_{\mathbb{R}}^x$, is an extension of the λ_x -calculus with explicit operators for weakening and contraction. It corresponds to the λ_{lxr} -calculus of Kesner and Lengrand, proposed in [36], and also represents a vertex of “the prismoid of resources”.

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

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

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:

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

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

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

$\frac{}{x \in \Lambda_{\mathbb{R}}^x} \qquad \frac{f \in \Lambda_{\mathbb{R}}^x \quad x \in Fv(f)}{\lambda x.f \in \Lambda_{\mathbb{R}}^x}$
$\frac{f \in \Lambda_{\mathbb{R}}^x \quad g \in \Lambda_{\mathbb{R}}^x \quad Fv(f) \cap Fv(g) = \emptyset}{fg \in \Lambda_{\mathbb{R}}^x}$
$\frac{f \in \Lambda_{\mathbb{R}}^x \quad g \in \Lambda_{\mathbb{R}}^x \quad x \in Fv(f) \quad (Fv(f) \setminus \{x\}) \cap Fv(g) = \emptyset}{f\langle x := g \rangle \in \Lambda_{\mathbb{R}}^x}$
$\frac{f \in \Lambda_{\mathbb{R}}^x \quad x \notin Fv(f)}{x \odot f \in \Lambda_{\mathbb{R}}^x} \qquad \frac{f \in \Lambda_{\mathbb{R}}^x \quad x_1 \neq x_2, \quad x_1, x_2 \in Fv(f) \quad x \notin Fv(f) \setminus \{x_1, x_2\}}{x <_{x_2}^{x_1} f \in \Lambda_{\mathbb{R}}^x}$

Figure 9: $\Lambda_{\mathbb{R}}^x$: $\lambda_{\mathbb{R}}^x$ -terms

The notion of terms corresponds to the notion of linear terms in [36].

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

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

3.2. Intersection types for $\lambda_{\mathbb{R}}^x$

In this subsection we introduce intersection type assignment system which assigns *strict types* to $\lambda_{\mathbb{R}}^x$ -terms. The system is syntax-directed, hence significantly different from the one proposed in [43].

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 $\lambda_{\mathbb{R}}^x \cap$ 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.

(β_x)	$(\lambda x.M)N \rightarrow M\langle x := N \rangle$
(σ_1)	$x\langle x := N \rangle \rightarrow N$
(σ_2)	$(\lambda y.M)\langle x := N \rangle \rightarrow \lambda y.M\langle x := N \rangle$
(σ_3)	$(MP)\langle x := N \rangle \rightarrow M\langle x := N \rangle P$, if $x \notin Fv(P)$
(σ_4)	$(MP)\langle x := N \rangle \rightarrow MP\langle x := N \rangle$, if $x \notin Fv(M)$
(σ_5)	$(x \odot M)\langle x := N \rangle \rightarrow Fv(N) \odot M$
(σ_6)	$(y \odot M)\langle x := N \rangle \rightarrow y \odot M\langle x := N \rangle$, if $x \neq y$
(σ_7)	$(x <_{x_2}^{x_1} M)\langle x := N \rangle \rightarrow Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M\langle x_1 := N_1 \rangle \langle x_2 := N_2 \rangle$
(σ_8)	$(M\langle x := N \rangle)\langle y := P \rangle \rightarrow M\langle x := N \langle y := P \rangle \rangle$, if $y \notin Fv(M) \setminus \{x\}$
(γ_1)	$x <_{x_2}^{x_1} (\lambda y.M) \rightarrow \lambda y.x <_{x_2}^{x_1} M$
(γ_2)	$x <_{x_2}^{x_1} (MN) \rightarrow (x <_{x_2}^{x_1} M)N$, if $x_1, x_2 \notin Fv(N)$
(γ_3)	$x <_{x_2}^{x_1} (MN) \rightarrow M(x <_{x_2}^{x_1} N)$, if $x_1, x_2 \notin Fv(M)$
(γ_4)	$x <_{x_2}^{x_1} (M\langle y := N \rangle) \rightarrow M\langle y := x <_{x_2}^{x_1} N \rangle$, if $x_1, x_2 \notin Fv(M) \setminus \{y\}$
(ω_1)	$\lambda x.(y \odot M) \rightarrow y \odot (\lambda x.M)$, $x \neq y$
(ω_2)	$(x \odot M)N \rightarrow x \odot (MN)$
(ω_3)	$M(x \odot N) \rightarrow x \odot (MN)$
(ω_4)	$M\langle y := x \odot N \rangle \rightarrow x \odot (M\langle y := N \rangle)$
$(\gamma\omega_1)$	$x <_{x_2}^{x_1} (y \odot M) \rightarrow y \odot (x <_{x_2}^{x_1} M)$, $y \neq x_1, x_2$
$(\gamma\omega_2)$	$x <_{x_2}^{x_1} (x_1 \odot M) \rightarrow M\langle x_2 := x \rangle$

Figure 10: Reduction rules of $\lambda_{\mathbb{R}}^x$ -calculus

(ε_1)	$x \odot (y \odot M) \equiv_{\lambda_{\mathbb{R}}^x} y \odot (x \odot M)$
(ε_2)	$x <_{x_2}^{x_1} M \equiv_{\lambda_{\mathbb{R}}^x} x <_{x_1}^{x_2} M$
(ε_3)	$x <_z^y (y <_v^u M) \equiv_{\lambda_{\mathbb{R}}^x} x <_u^y (y <_v^z M)$
(ε_4)	$x <_{x_2}^{x_1} (y <_{y_2}^{y_1} M) \equiv_{\lambda_{\mathbb{R}}^x} y <_{y_2}^{y_1} (x <_{x_2}^{x_1} M)$, $x \neq y_1, y_2$, $y \neq x_1, x_2$
(ε_5)	$M\langle x := N \rangle \langle y := P \rangle \equiv_{\lambda_{\mathbb{R}}^x} M\langle y := P \rangle \langle x := N \rangle$, $x \notin Fv(P)$, $y \notin Fv(M)$
(ε_6)	$(y <_{y_2}^{y_1} M)\langle x := N \rangle \equiv_{\lambda_{\mathbb{R}}^x} y <_{y_2}^{y_1} M\langle x := N \rangle$, $x \neq y$, $y_1, y_2 \notin Fv(N)$

Figure 11: Equivalences in $\lambda_{\mathbb{R}}^x$ -calculus

The rules (\rightarrow_E) and $(Subst)$ are constructed in the same manner, as explained in subsection 1.2.

$$\begin{array}{c}
\frac{}{x : \sigma \vdash x : \sigma} (Ax) \\
\\
\frac{\Gamma, x : \alpha \vdash M : \sigma}{\Gamma \vdash \lambda x. M : \alpha \rightarrow \sigma} (\rightarrow_I) \quad \frac{\Gamma \vdash M : \cap_i^n \tau_i \rightarrow \sigma \quad \Delta_0 \vdash N : \tau_0 \quad \dots \quad \Delta_n \vdash N : \tau_n}{\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash MN : \sigma} (\rightarrow_E) \\
\\
\frac{\Gamma, x : \cap_i^n \tau_i \vdash M : \sigma \quad \Delta_0 \vdash N : \tau_0 \quad \dots \quad \Delta_n \vdash N : \tau_n}{\Gamma, \Delta_0^\top \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash M\langle x := N \rangle : \sigma} (Subst) \\
\\
\frac{\Gamma, x : \alpha, y : \beta \vdash M : \sigma}{\Gamma, z : \alpha \cap \beta \vdash z <_y^x M : \sigma} (Cont) \quad \frac{\Gamma \vdash M : \sigma}{\Gamma, x : \top \vdash x \odot M : \sigma} (Weak)
\end{array}$$

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

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

- (i) $\Gamma \vdash \lambda x. M : \tau$ iff there exist α and σ such that $\tau \equiv \alpha \rightarrow \sigma$ and $\Gamma, x : \alpha \vdash M : \sigma$.
- (ii) $\Gamma \vdash MN : \sigma$ iff there exist Δ_j and τ_j , $j = 0, \dots, n$ such that $\Delta_j \vdash N : \tau_j$ and $\Gamma' \vdash M : \cap_i^n \tau_i \rightarrow \sigma$, 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) $\Gamma \vdash z <_y^x M : \sigma$ iff there exist Γ', α, β 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.

3.3. Typeability $\Rightarrow SN$ in $\lambda_{\mathbb{R}}^x \cap$

Now we prove the strong normalisation of the $\lambda_{\mathbb{R}}^x$ -calculus with intersection types. We follow the procedure established for proving the strong normalisation of the system $\lambda_{\mathbb{R}}^{Gtz} \cap$. In the sequel, we denote by $\Lambda_{\mathbb{R}}^x$ the set of $\lambda_{\mathbb{R}}^x$ -terms, and by $\Lambda_{\mathbb{R}}^x \cap$ the set of $\lambda_{\mathbb{R}}^x$ -terms typeable in the system $\lambda_{\mathbb{R}}^x \cap$.

Definition 56. The mapping $\lceil \cdot \rceil : \Lambda_{\mathbb{R}}^{\times} \rightarrow \Lambda_{\mathbb{R}}$ is defined in the following way:

$$\begin{array}{llll} \lceil x \rceil & = & x & \lceil M \langle x := N \rangle \rceil & = & \lceil M \rceil [\lceil N \rceil / x] \\ \lceil \lambda x. M \rceil & = & \lambda x. \lceil M \rceil & \lceil x \odot M \rceil & = & x \odot \lceil M \rceil \\ \lceil MN \rceil & = & \lceil M \rceil \lceil N \rceil & \lceil x <_z^y M \rceil & = & x <_z^y \lceil M \rceil \end{array}$$

Lemma 57. $Fv(M) = Fv(\lceil M \rceil)$, for $M \in \Lambda_{\mathbb{R}}^{\times}$.⁵

Proof. The proof is an easy induction on the structure of the term M . All the cases are straightforward, we only present the case of explicit substitution.

$$\begin{aligned} Fv(M \langle x := N \rangle) &= (Fv(M) \setminus \{x\}) \cup Fv(N) = Fv(\lceil M \rceil) \setminus \{x\} \cup Fv(\lceil N \rceil) = \\ Fv((\lambda x. \lceil M \rceil) \lceil N \rceil) &= Fv(\lceil M \rceil [\lceil N \rceil / x]) = Fv(\lceil M \langle x := N \rangle \rceil) \end{aligned} \quad \square$$

We prove that the mapping $\lceil \cdot \rceil$ preserves types.

Proposition 58 (Type preservation by $\lceil \cdot \rceil$). *If $\Gamma \vdash_{\lambda_{\mathbb{R}}^{\times}} M : \sigma$, then $\Gamma \vdash_{\lambda_{\mathbb{R}}} \lceil M \rceil : \sigma$.*

Proof. The proposition is proved by induction on derivations. We distinguish cases according to the last typing rule used. Cases (Ax) , (\rightarrow_I) , (\rightarrow_E) , $(Weak)$ and $(Cont)$ are trivial, because the intersection type assignment system of $\lambda_{\mathbb{R}}$ has exactly the same rules. The only interesting case is the rule $(Subst)$. In that case, the derivation ends with the rule

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

By IH we have that $\Gamma, x : \cap_i^n \tau_i \vdash_{\lambda_{\mathbb{R}}} \lceil M \rceil : \sigma$ and for all $j \in \{0, \dots, n\}$, $\Delta_j \vdash_{\lambda_{\mathbb{R}}} \lceil N \rceil : \tau_j$. Now we can apply Lemma 10 yielding $\Gamma, \Delta_0^{\top} \sqcap \Delta_1 \sqcap \dots \sqcap \Delta_n \vdash_{\lambda_{\mathbb{R}}} \lceil M \rceil [\lceil N \rceil / x] : \sigma$. Since $\lceil M \rceil [\lceil N \rceil / x] = \lceil M \langle x := N \rangle \rceil$, the proof is done. \square

We now show that each $\lambda_{\mathbb{R}}^{\times}$ -reduction step can be simulated through the encoding $\lceil \cdot \rceil$, by a $\lambda_{\mathbb{R}}$ -reduction or an equality.

Theorem 59 (Simulation of $\lambda_{\mathbb{R}}^{\times}$ -reduction and equivalence by $\lambda_{\mathbb{R}}$ -reduction and equivalence).

(i) *If a $\lambda_{\mathbb{R}}^{\times}$ -term $M \rightarrow M'$, then $\lceil M \rceil \rightarrow_{\lambda_{\mathbb{R}}} \lceil M' \rceil$ or $\lceil M \rceil = \lceil M' \rceil$.*

⁵Notice that in [36] $Fv(\mathcal{B}(t)) \subseteq Fv(t)$ holds, where \mathcal{B} is the mapping from $\lambda_{\mathbb{R}}^{\times}$ -calculus to ordinary λ -calculus without resources and that is why free variables disappear there. Instead, our mapping $\lceil \cdot \rceil$ maps $\lambda_{\mathbb{R}}^{\times}$ -terms to $\lambda_{\mathbb{R}}$ -terms.

(ii) If a $\lambda_{\mathbb{R}}^{\times}$ -term $M \equiv_{\lambda_{\mathbb{R}}^{\times}} M'$, then $\lceil M \rceil \equiv_{\lambda_{\mathbb{R}}} \lceil M' \rceil$ or $\lceil M \rceil = \lceil M' \rceil$.

Proof. (i) Without losing generality, we prove the statement only for the outermost reductions. The rules $\gamma_1, \gamma_2, \gamma_3, \omega_1, \omega_2, \omega_3, \gamma\omega_1, \gamma\omega_2$ are simulated by the corresponding rules of $\lambda_{\mathbb{R}}$. For the remaining rules we have:

$$(\beta_x) (\lambda x.M)N \rightarrow M\langle x := N \rangle.$$

$$\lceil (\lambda x.M)N \rceil = \lceil \lambda x.M \rceil \lceil N \rceil = (\lambda x.\lceil M \rceil) \lceil N \rceil \rightarrow_{\lambda_{\mathbb{R}}} \lceil M \rceil \lceil \lceil N \rceil / x \rceil = \lceil M\langle x := N \rangle \rceil.$$

$$(\sigma_1) x\langle x := N \rangle \rightarrow N.$$

$$\lceil x\langle x := N \rangle \rceil = \lceil x \rceil \lceil \lceil N \rceil / x \rceil = x \lceil \lceil N \rceil / x \rceil \triangleq \lceil N \rceil.$$

$$(\sigma_2) (\lambda y.M)\langle x := N \rangle \rightarrow \lambda y.M\langle x := N \rangle.$$

$$\lceil (\lambda y.M)\langle x := N \rangle \rceil = \lceil \lambda y.M \rceil \lceil \lceil N \rceil / x \rceil = (\lambda y.\lceil M \rceil) \lceil \lceil N \rceil / x \rceil \triangleq \lambda y.\lceil M \rceil \lceil \lceil N \rceil / x \rceil = \lambda y.\lceil M\langle x := N \rangle \rceil = \lceil \lambda y.M\langle x := N \rangle \rceil.$$

$$(\sigma_3), (\sigma_4) (MP)\langle x := N \rangle \rightarrow M\langle x := N \rangle P, \text{ if } x \notin Fv(P).$$

$$(MP)\langle x := N \rangle \rightarrow MP\langle x := N \rangle, \text{ if } x \notin Fv(M).$$

$$\lceil MP\langle x := N \rangle \rceil = (\lceil M \rceil \lceil P \rceil) \lceil \lceil N \rceil / x \rceil.$$

$$\text{If } x \notin Fv(P) \text{ then } (\lceil M \rceil \lceil P \rceil) \lceil \lceil N \rceil / x \rceil \triangleq \lceil M \rceil \lceil \lceil N \rceil / x \rceil \lceil P \rceil = \lceil M\langle x := N \rangle \rceil \lceil P \rceil = \lceil M\langle x := N \rangle P \rceil.$$

$$\text{If } x \notin Fv(M) \text{ then } (\lceil M \rceil \lceil P \rceil) \lceil \lceil N \rceil / x \rceil \triangleq \lceil M \rceil \lceil P \rceil \lceil \lceil N \rceil / x \rceil = \lceil M \rceil \lceil P\langle x := N \rangle \rceil = \lceil MP\langle x := N \rangle \rceil.$$

$$(\sigma_5) (x \odot M)\langle x := N \rangle \rightarrow Fv(N) \odot M.$$

$$\lceil (x \odot M)\langle x := N \rangle \rceil = \lceil x \odot M \rceil \lceil \lceil N \rceil / x \rceil = (x \odot \lceil M \rceil) \lceil \lceil N \rceil / x \rceil \triangleq Fv(\lceil N \rceil) \odot \lceil M \rceil = Fv(N) \odot \lceil M \rceil = \lceil Fv(N) \odot M \rceil, \text{ since } Fv(N) = Fv(\lceil N \rceil).$$

$$(\sigma_6) (y \odot M)\langle x := N \rangle \rightarrow y \odot M\langle x := N \rangle, \text{ if } x \neq y.$$

$$\lceil (y \odot M)\langle x := N \rangle \rceil = \lceil y \odot M \rceil \lceil \lceil N \rceil / x \rceil = (y \odot \lceil M \rceil) \lceil \lceil N \rceil / x \rceil \triangleq y \odot \lceil M \rceil \lceil \lceil N \rceil / x \rceil = y \odot \lceil M\langle x := N \rangle \rceil = \lceil y \odot M\langle x := N \rangle \rceil.$$

$$(\sigma_7) (x <_{x_2}^{x_1} M)\langle x := N \rangle \rightarrow Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M\langle x_1 := N_1 \rangle \langle x_2 := N_2 \rangle.$$

$$\lceil (x <_{x_2}^{x_1} M)\langle x := N \rangle \rceil = \lceil x <_{x_2}^{x_1} M \rceil \lceil \lceil N \rceil / x \rceil = x <_{x_2}^{x_1} \lceil M \rceil \lceil \lceil N \rceil / x \rceil \triangleq$$

$$Fv(\lceil N \rceil) <_{Fv(\lceil N_2 \rceil)}^{Fv(\lceil N_1 \rceil)} \lceil M \rceil \lceil \lceil N_1 \rceil / x_1, \lceil N_2 \rceil / x_2 \rceil =$$

$$\lceil Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M\langle x_1 := N_1 \rangle \langle x_2 := N_2 \rangle \rceil.$$

$$(\sigma_8) (M\langle x := N \rangle)\langle y := P \rangle \rightarrow M\langle x := N\langle y := P \rangle \rangle, \text{ if } y \notin Fv(M) \setminus \{x\}.$$

$$\lceil (M\langle x := N \rangle)\langle y := P \rangle \rceil = \lceil M\langle x := N \rangle \rceil \lceil \lceil P \rceil / y \rceil = (\lceil M \rceil \lceil \lceil N \rceil / x \rceil) \lceil \lceil P \rceil / y \rceil \stackrel{\text{Lemma 4}}{=} \lceil M \rceil \lceil \lceil N \rceil \lceil \lceil P \rceil / y \rceil / x \rceil = \lceil M \rceil \lceil \lceil N\langle y := P \rangle \rceil / x \rceil = \lceil M\langle x := N\langle y := P \rangle \rangle \rceil.$$

$$\begin{aligned}
(\gamma_4) \quad & x <_{x_2}^{x_1} (M\langle y := N \rangle) \rightarrow M\langle y := x <_{x_2}^{x_1} N \rangle, \text{ if } x_1, x_2 \notin Fv(M) \setminus \{y\}. \\
& [x <_{x_2}^{x_1} (M\langle y := N \rangle)] = x <_{x_2}^{x_1} [M\langle y := [N] \rangle] = x <_{x_2}^{x_1} [M][[N]/y] \xrightarrow{*}_{\text{Lemma 5(ii)}} \\
& [M][x <_{x_2}^{x_1} [N]/y] = x <_{x_2}^{x_1} ([M][N/y]) = [M\langle y := x <_{x_2}^{x_1} N \rangle]. \\
(\omega_4) \quad & M\langle y := x \odot N \rangle \rightarrow x \odot (M\langle y := N \rangle). \\
& [M\langle y := x \odot N \rangle] = [M][x \odot [N]/y] \xrightarrow{*}_{\text{Lemma 5(i)}} x \odot [M][[N]/y] = \\
& x \odot [M\langle y := N \rangle] = [x \odot (M\langle y := N \rangle)].
\end{aligned}$$

(ii) The equivalences $\varepsilon_1 - \varepsilon_4$ coincide in both calculi, hence we prove the statement only for ε_5 and ε_6 .

$$\begin{aligned}
(\varepsilon_5) \quad & M\langle x := N \rangle\langle y := P \rangle \equiv M\langle y := P \rangle\langle x := N \rangle, \quad x \notin Fv(P), y \notin Fv(M). \\
& [M\langle x := N \rangle\langle y := P \rangle] = [M\langle x := N \rangle][[P]/y] = ([M][[N]/x])[[P]/y] = \\
& [M][[N]/x, [P]/y] = [M][[P]/y, [N]/x] = ([M][[P]/y])[[N]/x] = \\
& [M\langle y := P \rangle][[N]/x] = [M\langle y := P \rangle\langle x := N \rangle]. \\
(\varepsilon_6) \quad & (y <_{y_2}^{y_1} M)\langle x := N \rangle \equiv y <_{y_2}^{y_1} M\langle x := N \rangle, \quad x \neq y, y_1, y_2 \notin Fv(N). \\
& [(y <_{y_2}^{y_1} M)\langle x := N \rangle] = [y <_{y_2}^{y_1} M][[N]/x] = (y <_{y_2}^{y_1} [M])[[N]/x] \triangleq \\
& y <_{y_2}^{y_1} [M][[N]/x] = y <_{y_2}^{y_1} [M\langle x := N \rangle] = [y <_{y_2}^{y_1} M\langle x := N \rangle].
\end{aligned}$$

□

The previous proposition shows that each $\lambda_{\mathbb{R}}^x$ -reduction step is interpreted either by a $\lambda_{\mathbb{R}}$ -reduction or by an equality. More precisely: $\beta_x, \gamma_1, \gamma_2, \gamma_3, \omega_1, \omega_2, \omega_3, \gamma\omega_1$ and $\gamma\omega_2$ reductions are interpreted by $\lambda_{\mathbb{R}}$ -reductions, $\sigma_1 - \sigma_8$ reductions are interpreted by identities, while γ_4 and ω_4 reductions are interpreted either by $\lambda_{\mathbb{R}}$ -reductions or by $\lambda_{\mathbb{R}}$ -equivalencies, depending on the structure of the term. In order to define a lexicographic product of orders that forbid infinite decreasing chains of $\lambda_{\mathbb{R}}^x$ -reductions, we use two measures defined on the set $\Lambda_{\mathbb{R}}^x$, namely term complexity $\mathcal{TC}(M)$ and interpretation $I(M)$ (based on multiplicity $\mathcal{M}(M)$), proposed in [36] and already used for proving the strong normalisation of the simply typed λlxr -calculus. We give their definitions here.

Definition 60 (Multiplicity [36]). Given a free variable x in a $\lambda_{\mathbb{R}}^x$ -term M , the multiplicity of x in M , written $\mathcal{M}_x(M)$, is defined by induction on terms as follows, supposing that $x \neq y, x \neq z, x \neq w$:

$$\begin{array}{ll}
\mathcal{M}_x(x) &= 1 & \mathcal{M}_x(x \odot M) &= 1 \\
\mathcal{M}_x(\lambda y.M) &= \mathcal{M}_x(M) & \mathcal{M}_x(y \odot M) &= \mathcal{M}_x(M) \\
\mathcal{M}_x(MN) &= \mathcal{M}_x(M) \text{ if } x \in Fv(M) & \mathcal{M}_x(x <_z^y M) &= \mathcal{M}_y(M) + \mathcal{M}_z(M) + 1 \\
\mathcal{M}_x(MN) &= \mathcal{M}_x(N) \text{ if } x \in Fv(N) & \mathcal{M}_x(w <_z^y M) &= \mathcal{M}_x(M)
\end{array}$$

$$\begin{aligned}\mathcal{M}_x(M\langle y := N \rangle) &= \mathcal{M}_x(M) \text{ if } x \in Fv(M) \setminus \{y\} \\ \mathcal{M}_x(M\langle y := N \rangle) &= \mathcal{M}_y(M) \cdot (\mathcal{M}_x(N) + 1) \text{ if } x \in Fv(N)\end{aligned}$$

Definition 61 (Term complexity [36]). The notion of term complexity $\mathcal{TC}(-)$ is defined by induction on terms as follows:

$$\begin{aligned}\mathcal{TC}(x) &= 1 & \mathcal{TC}(M\langle x := N \rangle) &= \mathcal{TC}(M) + \mathcal{M}_x(M) \cdot \mathcal{TC}(N) \\ \mathcal{TC}(\lambda x.M) &= \mathcal{TC}(M) & \mathcal{TC}(x \odot M) &= \mathcal{TC}(M) \\ \mathcal{TC}(MN) &= \mathcal{TC}(M) + \mathcal{TC}(N) & \mathcal{TC}(x <_z^y M) &= \mathcal{TC}(M)\end{aligned}$$

Definition 62 (Interpretation [36]). The notion of interpretation $I(-) : \Lambda_{\mathbb{R}}^x \rightarrow \mathbb{N}$ is defined as follows:

$$\begin{aligned}I(x) &= 2 & I(M\langle x := N \rangle) &= I(M) \cdot (I(N) + 1) \\ I(\lambda x.M) &= 2I(M) + 2 & I(x \odot M) &= I(M) + 1 \\ I(MN) &= 2(I(M) + I(N)) + 2 & I(x <_z^y M) &= 2I(M)\end{aligned}$$

Definition 63. We define the following strict orders and equivalencies on $\Lambda_{\mathbb{R}}^x \cap$:

- (i) $M >_{\lambda_{\mathbb{R}}^x} M'$ iff $[M] \rightarrow_{\lambda_{\mathbb{R}}^x}^+ [M']$; $M =_{\lambda_{\mathbb{R}}^x} M'$ iff $[M] \equiv_{\lambda_{\mathbb{R}}^x} [M']$ or $[M] = [M']$;
- (ii) $M >_{\mathcal{TC}} M'$ iff $\mathcal{TC}(M) > \mathcal{TC}(M')$; $M =_{\mathcal{TC}} M'$ iff $\mathcal{TC}(M) = \mathcal{TC}(M')$;
- (iii) $M >_I M'$ iff $I(M) > I(M')$; $M =_I M'$ iff $I(M) = I(M')$;

Definition 64. We define the relation \gg_x on $\Lambda_{\mathbb{R}}^x$ as the lexicographic product:

$$\gg_x = >_{\lambda_{\mathbb{R}}^x} \times_{lex} >_{\mathcal{TC}} \times_{lex} >_I.$$

The following proposition proves that the reduction relation on the set of typed $\lambda_{\mathbb{R}}^x$ -terms, $\Lambda_{\mathbb{R}}^x \cap$, is included in the given lexicographic product \gg_x .

Proposition 65. For each $M \in \Lambda_{\mathbb{R}}^x$: if $M \rightarrow M'$, then $M \gg_x M'$.

Proof. The proof is by case analysis on the kind of reduction and the structure of \gg_x .

If $M \rightarrow M'$ by $\beta_x, \gamma_1, \gamma_2, \gamma_3, \omega_1, \omega_2, \omega_3, \gamma\omega_1$ or $\gamma\omega_2$, reduction, then $M >_{\lambda_{\mathbb{R}}^x} M'$ by Theorem 59.

If $M \rightarrow M'$ by $\sigma_1, \sigma_5, \sigma_7$ or σ_8 then $M =_{\lambda_{\mathbb{R}}^x} M'$ by Theorem 59, and $M >_{\mathcal{TC}} M'$ by ([36], Lemma 3).

Finally, if $M \rightarrow M'$ by $\sigma_2, \sigma_3, \sigma_4, \sigma_6, \gamma_4$ or ω_4 , then $M =_{\lambda_{\mathbb{R}}^x} M'$ by Theorem 59, $M =_{\mathcal{TC}} M'$ by ([36], Lemma 3) and $M >_I M'$ by ([36], Lemma 4). \square

Theorem 66 (Strong normalization of $\lambda_{\mathbb{R}}^x \cap$). Each term in $\Lambda_{\mathbb{R}}^x \cap$ is SN.

Proof. The proof is analogous to the proof of Theorem 48. \square

3.4. $SN \Rightarrow$ Typeability in $\lambda_{\mathbb{R}}^x \cap$

We finally prove that if a $\lambda_{\mathbb{R}}^x$ -term is SN, then it is typeable in the system $\lambda_{\mathbb{R}}^x \cap$. Due to the definition of the $\lambda_{\mathbb{R}}^x$ -terms given in Figure 9, particularly $M\langle x := N \rangle$ where $x \in Fv(M)$ is required, as well as to the reductions $(\sigma_1 - \sigma_8)$, the set of $\lambda_{\mathbb{R}}^x$ -normal forms coincide with the set of $\lambda_{\mathbb{R}}$ -normal forms. This, combined with the fact that all type assignment rules from Figure 5 exist also in the $\lambda_{\mathbb{R}}^x \cap$ system, has as a consequence the following proposition.

Proposition 67. *$\lambda_{\mathbb{R}}^x$ -normal forms are typeable in the system $\lambda_{\mathbb{R}}^x \cap$.*

Proposition 68 (Head subject expansion). *For every $\lambda_{\mathbb{R}}^x$ -term M : if $M \rightarrow M'$, M is contracted redex and $\Gamma \vdash M' : \sigma$, then $\Gamma \vdash M : \sigma$, provided that if $M \equiv (\lambda x.N)P \rightarrow_{\beta_x} N\langle x := P \rangle \equiv M'$, P is typeable.*

Proof. By the case study according to the applied reduction. \square

Theorem 69 (SN \Rightarrow typeability). *All strongly normalising $\lambda_{\mathbb{R}}^x$ -terms are typeable in the $\lambda_{\mathbb{R}}^x \cap$ system.*

Now we can give a characterisation of strong normalisation in $\lambda_{\mathbb{R}}^x$ -calculus.

Theorem 70. *In $\lambda_{\mathbb{R}}^x$ -calculus, the term M is strongly normalising if and only if it is typeable in $\lambda_{\mathbb{R}}^x \cap$.*

Proof. Immediate consequence of Theorems 66 and 69. \square

4. Conclusions

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

- resource control lambda calculus $\lambda_{\mathbb{R}}$, which corresponds to λ_{CW} of [37];
- resource control sequent lambda calculus $\lambda_{\mathbb{R}}^{\text{Gtz}}$ of [28] and
- resource control calculus with explicit substitution $\lambda_{\mathbb{R}}^x$ of [36].

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 [49], 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 $\lambda_{\mathbb{R}}$, $\lambda_{\mathbb{R}}^{\text{Gtz}}$ and $\lambda_{\mathbb{R}}^{\times}$ are good candidates to investigate the computational content of substructural logics [58], 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 [7, 17], 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 [12, 39].

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] Abramsky, S., 1993. Computational interpretations of linear logic. *Theoretical Computer Science* 111 (1&2), 3–57.
- [2] Baader, F., Nipkow, T., 1998. *Term Rewriting and All That*. Cambridge University Press, UK.

- [3] van Bakel, S., 1992. Complete restrictions of the intersection type discipline. *Theoretical Computer Science* 102 (1), 135–163.
- [4] Barbanera, F., Berardi, S., 1996. A symmetric lambda calculus for classical program extraction. *Information and Computation* 125 (2), 103–117.
- [5] Barendregt, H. P., 1984. *The Lambda Calculus: its Syntax and Semantics*, revised Edition. North-Holland, Amsterdam.
- [6] Barendregt, H. P., 1992. Lambda calculi with types. In: Abramsky, S., Gabbay, D. M., Maibaum, T. S. E. (Eds.), *Handbook of Logic in Computer Science*. Oxford University Press, UK, pp. 117–309.
- [7] Barendregt, H. P., Coppo, M., Dezani-Ciancaglini, M., 1983. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic* 48 (4), 931–940 (1984).
- [8] Benton, N., Bierman, G., de Paiva, V., Hyland, M., 1993. A term calculus for intuitionistic linear logic. In: Bezem, M., Groote, J. F. (Eds.), *1st International Conference on Typed Lambda Calculus, TLCA '93*. Vol. 664 of *Lecture Notes in Computer Science*. Springer, pp. 75–90.
- [9] Bloo, R., Rose, K. H., 1995. Preservation of strong normalisation in named lambda calculi with explicit substitution and garbage collection. In: *Computer Science in the Netherlands, CSN '95*. pp. 62–72.
URL <ftp://ftp.diku.dk/diku/semantics/papers/D-246.ps>
- [10] Boudol, G., 1993. The lambda-calculus with multiplicities (abstract). In: Best, E. (Ed.), *4th International Conference on Concurrency Theory, CONCUR '93*. Vol. 715 of *Lecture Notes in Computer Science*. Springer, pp. 1–6.
- [11] Boudol, G., Curien, P.-L., Lavatelli, C., 1999. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science* 9 (4), 437–482.
- [12] Boudol, G., Zimmer, P., 2005. On type inference in the intersection type discipline. *Electronic Notes in Theoretical Computer Science* 136, 23–42.
- [13] Coppo, M., Dezani-Ciancaglini, M., 1978. A new type-assignment for lambda terms. *Archiv für Mathematische Logik* 19, 139–156.

- [14] Coppo, M., Dezani-Ciancaglini, M., 1980. An extension of the basic functionality theory for the λ -calculus. *Notre Dame Journal of Formal Logic* 21 (4), 685–693.
- [15] Curien, P.-L., Herbelin, H., 2000. The duality of computation. In: 5th International Conference on Functional Programming, ICFP'00. ACM Press, pp. 233–243.
- [16] Dezani-Ciancaglini, M., Ghilezan, S., 2003. Two behavioural lambda models. In: Geuvers, H., Wiedijk, F. (Eds.), *Types for Proofs and Programs*. Vol. 2646 of *Lecture Notes in Computer Science*. Springer, pp. 127–147.
- [17] Dezani-Ciancaglini, M., Ghilezan, S., Likavec, S., 2004. Behavioural Inverse Limit Models. *Theoretical Computer Science* 316 (1–3), 49–74.
- [18] Dezani-Ciancaglini, M., Honsell, F., Motohama, Y., 2000. Compositional characterization of λ -terms using intersection types. In: 25th International Symposium on Mathematical Foundations of Computer Science, MFCS '00. Vol. 1893 of *Lecture Notes in Computer Science*. Springer, pp. 304–314.
- [19] Dougherty, D. J., Ghilezan, S., Lescanne, P., 2008. Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: extending the Coppo-Dezani heritage. *Theoretical Computer Science* 398, 114–128.
- [20] Ehrhard, T., Regnier, L., 2003. The differential lambda-calculus. *Theoretical Computer Science* 309 (1-3), 1–41.
- [21] Espírito Santo, J., 2007. Completing Herbelin's programme. In: Rocca, S. R. D. (Ed.), 9th International Conference on Typed Lambda Calculi and Applications, TLCA '07. Vol. 4583 of *Lecture Notes in Computer Science*. Springer, pp. 118–132.
- [22] Espírito Santo, J., Ghilezan, S., Ivetić, J., 2008. Characterising strongly normalising intuitionistic sequent terms. In: International Workshop TYPES'07 (Selected Papers). Vol. 4941 of *Lecture Notes in Computer Science*. Springer, pp. 85–99.
- [23] Espírito Santo, J., Ivetić, J., Likavec, S., 2011. Characterising strongly normalising intuitionistic terms. *Fundamenta Informaticae* To appear.

- [24] Gallier, J., 1998. Typing untyped λ -terms, or reducibility strikes again! *Annals of Pure and Applied Logic* 91, 231–270.
- [25] Gentzen, G., 1935. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift* 39, 176–210, 405–431.
- [26] Ghilezan, S., 1996. Strong normalization and typability with intersection types. *Notre Dame Journal of Formal Logic* 37 (1), 44–52.
- [27] Ghilezan, S., Ivetić, J., Lescanne, P., Likavec, S., 2011. Intersection types for the resource control lambda calculi. In: Cerone, A., Pihlajasaari, P. (Eds.), 8th International Colloquium on Theoretical Aspects of Computing, ICTAC '11. Vol. 6916 of Lecture Notes in Computer Science. Springer, pp. 116–134.
- [28] Ghilezan, S., Ivetić, J., Lescanne, P., Žunić, D., 2011. Intuitionistic sequent-style calculus with explicit structural rules. In: 8th International Tbilisi Symposium on Language, Logic and Computation. Vol. 6618 of LNAI. pp. 101–124.
- [29] Ghilezan, S., Likavec, S., 2002. Reducibility: A Ubiquitous Method in Lambda Calculus with Intersection Types. In: van Bakel, S. (Ed.), ITRS '02. Vol. 70 of Electronic Notes in Theoretical Computer Science. pp. 106–123.
- [30] Ghilezan, S., Likavec, S., 2009. Computational interpretations of logics. In: Ognjanović, Z. (Ed.), Collection of Papers, special issue Logic in Computer Science 20(12). Mathematical Institute of Serbian Academy of Sciences and Arts, pp. 159–215.
- [31] Girard, J.-Y., 1971. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types. In: Fenstad, J. E. (Ed.), 2nd Scandinavian Logic Symposium. North-Holland, pp. 63–92.
- [32] Girard, J.-Y., 1987. Linear logic. *Theoretical Computer Science* 50, 1–102.
- [33] Girard, J.-Y., Lafont, Y., Taylor, P., 1989. Proofs and Types. Vol. 7 of Cambridge Tracts in Theoret Computer Science. Cambridge University Press.
- [34] Herbelin, H., 1995. A lambda calculus structure isomorphic to Gentzen-style sequent calculus structure. In: Pacholski, L., Tiuryn, J. (Eds.), Computer

Science Logic, CSL '94. Vol. 933 of Lecture Notes in Computer Science. Springer, pp. 61–75.

- [35] Howard, W. A., 1980. The formulas-as-types notion of construction. In: Seldin, J. P., Hindley, J. R. (Eds.), *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*. Academic Press, London, pp. 479–490.
- [36] Kesner, D., Lengrand, S., 2007. Resource operators for lambda-calculus. *Information and Computation* 205 (4), 419–473.
- [37] Kesner, D., Renaud, F., 2009. The prismoid of resources. In: c, R. K., Niwiński, D. (Eds.), *34th International Symposium on Mathematical Foundations of Computer Science, MFCS '09*. Vol. 5734 of Lecture Notes in Computer Science. Springer, pp. 464–476.
- [38] Kesner, D., Renaud, F., 2011. A prismoid framework for languages with resources. *Theoretical Computer Science* 412 (37), 4867–4892.
- [39] Kfoury, A. J., Wells, J. B., 2004. Principality and type inference for intersection types using expansion variables. *Theoretical Computer Science* 311 (1–3), 1–70.
- [40] Kikuchi, K., 2007. Simple proofs of characterizing strong normalisation for explicit substitution calculi. In: Baader, F. (Ed.), *18th International Conference on Term Rewriting and Applications, RTA'07*. Vol. 4533 of Lecture Notes in Computer Science. Springer, pp. 257–272.
- [41] Koletsos, G., 1985. Church-Rosser theorem for typed functionals. *Journal of Symbolic Logic* 50, 782–790.
- [42] Krivine, J.-L., 1990. *Lambda-calcul types et modèles*. Masson, Paris.
- [43] Lengrand, S., Lescanne, P., Dougherty, D., Dezani-Ciancaglini, M., van Bakel, S., 2004. Intersection types for explicit substitutions. *Information and Computation* 189 (1), 17–42.
- [44] Matthes, R., 2000. Characterizing strongly normalizing terms of a calculus with generalized applications via intersection types. In: *ICALP Satellite Workshops*. pp. 339–354.

- [45] Mitchell, J. C., 1990. Type systems for programming languages. In: van Leeuwen, J. (Ed.), Handbook of Theoretical Computer Science. Volume B. Elsevier, Amsterdam, pp. 415–431.
- [46] Mitchell, J. C., 1996. Foundation for Programming Languages. MIT Press, Boston.
- [47] Neergaard, P. M., 2005. Theoretical pearls: A bargain for intersection types: a simple strong normalization proof. *Journal of Functional Programming* 15 (5), 669–677.
- [48] van Oostrom, V., 2001. Net-calculus. Course notes, Utrecht University.
URL <http://www.phil.uu.nl/~oostrom/oudonderwijs/cmitten/00-01/net.ps>
- [49] Pagani, M., Rocca, S. R. D., 2010. Solvability in resource lambda-calculus. In: Ong, C.-H. L. (Ed.), 13th International Conference on Foundations of Software Science and Computational Structures, FOSSACS 2010. Vol. 6014 of Lecture Notes in Computer Science. Springer, pp. 358–373.
- [50] Parigot, M., 1992. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In: Voronkov, A. (Ed.), 3rd International Conference on Logic Programming and Automated Reasoning, LPAR '92. Vol. 624 of Lecture Notes in Computer Science. Springer, pp. 190–201.
- [51] Pinto, L., Dyckhoff, R., 1998. Sequent calculi for the normal terms of the $\lambda\pi$ and $\lambda\pi\sigma$ calculi. *Electronic Notes in Theoretical Computer Science* 17, 1–14.
- [52] Pottinger, G., 1980. A type assignment for the strongly normalizable λ -terms. In: Seldin, J. P., Hindley, J. R. (Eds.), To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism. Academic Press, London, pp. 561–577.
- [53] Regnier, L., 1994. Une équivalence sur les lambda-termes. *Theoretical Computer Science* 126 (2), 281–292.
- [54] Rose, K., Bloo, R., Lang, F., 2011. On explicit substitution with names. *Journal of Automated Reasoning*, 1–26.
- [55] Rose, K. H., 2011. CRSX - Combinatory Reduction Systems with Extensions. In: Schmidt-Schauß, M. (Ed.), 22nd International Conference on

Rewriting Techniques and Applications, RTA'11. Vol. 10 of Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 81–90.

- [56] Rose, K. H., 2011. Implementation Tricks That Make CRSX Tick. Talk at IFIP 1.6 workshop, RDP'2011.
- [57] Sallé, P., 1978. Une extension de la théorie des types en lambda-calcul. In: Ausiello, G., Böhm, C. (Eds.), 5th International Conference on Automata, Languages and Programming, ICALP '78. Vol. 62 of Lecture Notes in Computer Science. Springer, pp. 398–410.
- [58] Schroeder-Heister, P., Došen, K., 1993. Substructural Logics. Oxford University Press, UK.
- [59] Statman, R., 1985. Logical relations and the typed λ -calculus. *Information and Control* 65, 85–97.
- [60] Tait, W. W., 1967. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic* 32, 198–212.
- [61] Tait, W. W., 1975. A realizability interpretation of the theory of species. In: Parikh, R. (Ed.), *Logic Colloquium*. Vol. 453 of Lecture Notes in Mathematics. Springer, pp. 240–251.
- [62] Žunić, D., 2007. Computing with sequents and diagrams in classical logic - calculi $*\mathcal{X}$, $^d\mathcal{X}$ and $^\circ\mathcal{X}$. Phd thesis, École Normale Supérieure de Lyon.
URL <http://tel.archives-ouvertes.fr/tel-00265549>