Resource control and strong normalisation

Silvia Ghilezan, Jelena Ivetic, Pierre Lescanne, Silvia Likavec

Resource control and strong normalisation

S. Ghilezan
University of Novi Sad, Faculty of Technical Sciences, Serbia
gsilvia@uns.ac.rs

J. Ivetić
University of Novi Sad, Faculty of Technical Sciences, Serbia
jelenaivetic@uns.ac.rs

P. Lescanne
University of Lyon, École Normal Supérieure de Lyon, France
pierre.lescanne@ens-lyon.fr

S. Likavec
Dipartimento di Informatica, Università di Torino, Italy
likavec@di.unito.it

Abstract

We introduce the *resource control cube*, a system consisting of eight intuitionistic lambda calculi with either implicit or explicit control of resources and with either natural deduction or sequent calculus. The four calculi of the cube that correspond to natural deduction have been proposed by Kesner and Renaud and the four calculi that correspond to sequent lambda calculi are introduced in this paper. The presentation is paramatrized with the set of resources (weakening or contraction), which enables a uniform treatment of the eight calculi of the cube. The simply typed resource control cube, on the one hand, expands the Curry-Howard correspondence to intuitionistic natural deduction and intuitionistic sequent logic with implicit or explicit structural rules and, on the other hand, is related to substructural logics.

We propose a general intersection type system for the resource control cube calculi. Our main contribution is a characterisation of strong normalisation of reductions in this cube. First, we prove that typeability implies strong normalisation in the "natural deduction base" of the cube by adapting the reducibility method. We then prove that typeability implies strong normalisation in the "sequent base" of the cube by using a combination of well-orders and a suitable embedding in the "natural deduction base". Finally, we prove that strong normalisation implies typeability in the cube using head subject expansion. All proofs are general and can be made specific to each calculus of the cube by instantiating the set of resources.

Keywords: lambda calculus, sequent calculus, resource control, intersection types, strong normalisation

ACM subject classification: F.4.1 [Mathematical Logic]: Lambda calculus and related systems, F.3.1 [Specifying and Verifying and Reasoning about Programs]: Logics of programs, F.3.2 [Semantics of Programming Languages]: Operational semantics, F.3.3 [Studies of Program Constructs]: Type structure

Introduction

Curry–Howard correspondence or formulae-as-types and proofs-as-programs paradigm [How80], establishes a fundamental connection between various logical and computational systems. Simply typed λ-calculus provides the compu-

tational interpretation of intuitionistic natural deduction where simplifying a proof corresponds to a program execution. The correspondence between sequent calculus derivations and natural deduction derivations is not a one-to-one map: several cut-free derivations correspond to one normal derivation [BG00]. Starting from Griffin's extension of the Curry–Howard correspondence to classical logic [Gri90], this connection has been extended to other calculi and logical systems. For instance, Parigot's $\lambda\mu$-calculus [Par92] corresponds to classical natural deduction and as such inspired investigation into the relationship between classical logic and theories of control in programming languages [Par97, dG94, OS97, Bie98, AH03, HG08]. In the realm of sequent calculus, Herbelin's $\bar{\lambda}$-calculus [Her95] and Espírito Santo's λ^{Gtz}-calculus [Esp07a] correspond to intuitionistic sequent calculus, whereas Barbanera and Berardi's symmetric calculus [BB96] and Curien and Herbelin's $\bar{\lambda}\tilde{\mu}\tilde{\mu}$-calculus [CH00] correspond to its classical variants. An extensive overview of this subject can be found in [SU06, GL09].

Our intention is to bring this correspondence to the calculi with control operators, namely erasure and duplication, which correspond to weakening and contraction on the logical side. The wish to control the use of variables in a λ-term can be traced back to Church [Chu41] who introduced the λI-calculus. According to Church, the variables bound by λ abstraction should occur in the term at least once. More recently, following the ideas of linear logic [Gir87], van Oostrom [vO01] proposed to extend the λ-calculus, and Kesner and Lengrand [KL07] proposed to extend the λx-calculus with explicit substitution, with operators to tightly control the use of variables (resources). Resource control in sequent calculus corresponding to classical logic was proposed in [Ž07]. Resource control both in λ-calculus and λx-calculus is proposed in [KR09, KR11], whereas resource control for sequent λ-calculus is proposed in [GILŽ11, GILL11]. Like in the λI-calculus, bound variables must still occur in the term, but if a variable x is not used in a term M this can be expressed by using the expression $x \odot M$ where the operator $\odot$ is called *erasure (aka weakening)*. In this way, the term M does not contain the variable x, but the term $x \odot M$ does. Similarly, a variable should not occur twice. If nevertheless, we want to have two positions for the same variable, we have to duplicate it explicitly, using fresh names. This is done by using the operator $x <^{x_1}_{x_2}$, called *duplication (aka contraction)* which creates two fresh variables x_1 and x_2. Extending the classical λ-calculus and the sequent λ-calculus λ^{Gtz} with explicit erasure and duplication provides the Curry–Howard correspondence for intuitionistic natural deduction and intuitionistic sequent calculus with explicit structural rules, as investigated in [KL07, KR09, KR11, GILŽ11]. The notation used in this paper is along the lines of [Ž07] and close to [vO01].

A different approach to the resource aware lambda calculus, motivated mostly by the development of the process calculi, was investigated by Boudol in [Bou93]. Instead of extending the syntax of λ-calculus with explicit resource operators, Boudol proposed a non-deterministic calculus with a generalised notion of application. In his work, a function is applied to a structure called a bag, having the form $(N_1^{m_1}|...|N_k^{m_k})$ in which N_i, $i = 1,...,k$ are resources and $m_i \in \mathbb{N} \cup \{\infty\}$, $i = 1,...,k$ are multiplicities, representing the maximum possible number of the resource usage. In this framework, the usual application is written as MN^{∞}. The theory was further developed in [BCL99], connected to linear logic via differential λ-calculus in [ER03] and even typed with non-idempotent intersection types in [PR10].

Our contribution is inspired by Kesner and Lengrand's [KL07] and Kesner and Renaud's [KR09, KR11] work on resource operators for λ-calculus. The linear λlxr calculus of Kesner and Lengrand introduces operators for substitution, erasure and duplication, preserving at the same time strong normalisation, confluence and subject reduction property of its predecessor λx [BR95]. This approach is then generalised in Kesner and Renaud's *Prismoid of Resources* [KR09, KR11] by considering various combinations in which these three operators are made explicit or implicit. In this paper we introduce the notion of *resource control cube*, consisting of two systems $\lambda_{\mathcal{R}}$ and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$, which can be seen as two opposite sides of the cube (see Figure 1). In $\lambda_{\mathcal{R}}$ we consider four calculi, each of which is obtained by making erasure (aka weakening) or duplication (aka contraction) explicit or implicit. In $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ we consider the four sequent style counterparts of the $\lambda_{\mathcal{R}}$-calculi. We will call the $\lambda_{\mathcal{R}}$ system *the natural deduction ND-base of the cube*, and the $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ system *the sequent LJ-base of the cube*[1]. Explicit control of erasure and duplication leads to decomposing of reduction steps into more atomic ones, thus revealing details of computation which are usually left implicit. Since erasing and duplicating of (sub)terms essentially changes the structure of a program, it is important to see how this mechanism really works and to be able to control this part of computation.

Let us turn our attention to type assignment systems for various term calculi. The basic type assignment system is the one with *simple types* in which the only forming operator is an arrow $\rightarrow$. Among many extensions of this type

[1] We are aware that our notation is not symmetric, since Gtz is specified explicitly for the LJ-base but ND is not for the ND-base. This was a conscious decision because ND-base contains ordinary λ-calculus where correspondence with natural deduction is not explicitly written.

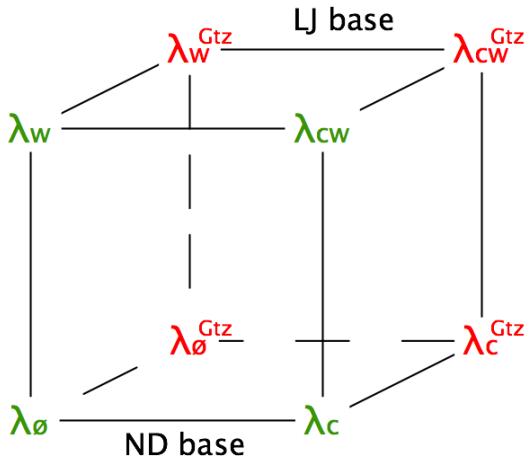

Figure 1: Resource control cube

assignment system, the one that characterises the largest class of terms, is the extension with *intersection types*, where a new type forming operator $\cap$ is introduced. The principal feature of this operator is the possibility to assign two different types to a certain term at the same time. The intersection types have been originally introduced in [CDC78, Sal78, CDC80, Pot80, CDCV80] in order to characterise termination properties of various term calculi [vB92, Gal98, Ghi96]. The extension of Curry–Howard correspondence to other formalisms brought the need for intersection types into many different settings [DGL08, Kik07, Mat00, Nee05].

We introduce the intersection types into all eight calculi of the resource control cube. Our intersection type assignment systems $\lambda_{\mathcal{R}}\cap$ and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$ integrate intersections into logical rules, thus preserving syntax-directedness of the systems. We assign a restricted form of intersection types to terms, namely strict types, therefore eliminating the need for pre-order on types. Using these intersection type assignment systems we manage to completely characterise strong normalisation in the cube by proving that terms in all calculi in both bases enjoy the strong normalisation property if and only if they are typeable. While this characterization is well-known for the pure λ-calculus, it is a new result for the other seven calculi of the cube. The paper proves the characterization in a modular way, presenting one uniform proof for the first four calculi in natural deduction style, and one uniform proof for the four sequent style calculi. Concerning the sequent calculus, to the best of our knowledge, there is no literature on explicit resource control operators for λ^{Gtz}-calculus, nor on the connection to intersection types (except for the initial version of this work presented in [GILL11]).

The rest of the paper is organised as follows. In Section 1 we introduce the untyped resource control cube which consists of eight calculi $\lambda_{\mathcal{R}}$ and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ with different combinations of explicit/implicit control operators of weakening and contraction. Basic type assignment systems with simple types for these calculi are given in Section 2. Intersection type assignment systems with strict types are introduced in Section 3. By adapting the reducibility method to explicit resource control operators, we first prove that typeable terms are strongly normalising in $\lambda_{\mathcal{R}}$-calculi of the ND-base in Section 4. Next, we prove that typeability implies strong normalization in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi of the LJ-base by using a combination of well-orders and a suitable embedding of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-terms into $\lambda_{\mathcal{R}}$-terms which preserves types and enables the simulation of all the reductions in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi by the operational semantics of the $\lambda_{\mathcal{R}}$-calculi. Finally, in Section 5, we prove that strong normalisation implies typeability in both systems using head subject expansion. We conclude in Section 6.

Contents

1 Untyped resource control cube

1.1 Resource control lambda calculi $\lambda_{\mathcal{R}}$

In this section we present four calculi obtained by adding contraction and weakening to the λ-calculus either as explicit or implicit operators. We denote these four calculi by $\lambda_{\mathcal{R}}$, where $\mathcal{R}\subseteq\{\mathsf{c},\mathsf{w}\}$ and c, w denote explicit contraction and weakening, respectively. We revisit a part of the untyped resource control calculi of Kesner and Renaud's prismoid [KR09], the so called implicit base, in which substitution is implicit. We use a notation along the lines of [Ž07] and close to [vO01]. It is slightly modified w.r.t. [KR09] in order to emphasize the correspondence between these calculi and their sequent counterparts.[2] For the convenience of the reader and to avoid repetition, we present all the calculi in a uniform way. This implies that some constructions or features are part of one calculus and not of the others. When a feature occurs in a calculus associated with the operator $\mathsf{r}\in\mathcal{R}$ and is ignored elsewhere, we put this feature between brackets indexed by the symbol r. For instance, if we write $[x\in Fv(f)]_{\mathsf{w}}$, this means that the condition $x\in Fv(f)$ appears only in the calculus which contains explicit weakening, as seen from the index w.

In order to define the terms of the calculus, we introduce a syntactic category called *pre-terms*. A *pre-term* can be a variable, an abstraction, an application, a contraction or a weakening. The abstract syntax of the $\lambda_{\mathcal{R}}$ pre-terms is given by the following:

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

where x ranges over a denumerable set of term variables

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

$$Fv(x)=x;\quad Fv(\lambda x.f)=Fv(f)\setminus\{x\};\quad Fv(fg)=Fv(f)\cup Fv(g);$$
$$[Fv(x\odot f)=\{x\}\cup Fv(f)]_{\mathsf{w}};$$
$$\left[Fv(x<^{x_1}_{x_2} f)=\begin{cases} Fv(f), & \{x_1,x_2\}\cap Fv(f)=\emptyset \\ \{x\}\cup Fv(f)\setminus\{x_1,x_2\}, & \{x_1,x_2\}\cap Fv(f)\neq\emptyset\end{cases}\right]_{\mathsf{c}}$$

In $x<^{x_1}_{x_2} f$, the duplication binds the variables x_1 and x_2 in f and introduces a fresh variable x if at least one of x_1,x_2 is free in f. In $x\odot f$, the variable x is free. In order to avoid parentheses, we let the scope of all binders extend to the right as much as possible.

The sets of $\lambda_{\mathcal{R}}$-terms, denoted by $\Lambda_{\mathcal{R}}$, are subsets of the set of pre-terms and are defined by the inference rules given in Figure 2. In the reminder of the paper, we will use $M,N,P...$ to denote terms. We also use the notation $X\odot M$ for $x_1\odot ...\, x_n\odot M$ and $X<^Y_Z M$ for $x_1<^{y_1}_{z_1} ...\, x_n<^{y_n}_{z_n} M$, where X, Y and Z are lists of size n, consisting of all distinct

[2] Note that in [KR09], instead of considering the sequent counterparts, the authors consider the so called explicit base, i.e. the four corresponding calculi with explicit substitutions.

variables $x_1,...,x_n,y_1,...,y_n,z_1,...,z_n$. If $n=0$, i.e., if X is the empty list, then $X \odot M = X <^Y_Z M = M$. Note that due to the equivalence relation defined in Figure 5, we can use these notations also for sets of variables of the same size.

$$\frac{}{x \in \Lambda_{\mathcal{R}}}$$

$$\frac{f \in \Lambda_{\mathcal{R}} \;\; [x \in Fv(f)]_{\mathsf{w}}}{\lambda x.f \in \Lambda_{\mathcal{R}}} \qquad \frac{f \in \Lambda_{\mathcal{R}} \;\; g \in \Lambda_{\mathcal{R}} \;\; [Fv(f) \cap Fv(g) = \emptyset]_{\mathsf{c}}}{fg \in \Lambda_{\mathcal{R}}}$$

$$\frac{f \in \Lambda_{\mathcal{R}} \;\; x \notin Fv(f)}{x \odot f \in \Lambda_{\mathcal{R}}} \; (\mathsf{w} \in \mathcal{R})$$

$$\frac{f \in \Lambda_{\mathcal{R}} \;\; x_1 \neq x_2 \;\; x \notin Fv(f) \setminus \{x_1,x_2\} \;\; [x_1,x_2 \in Fv(f)]_{\mathsf{w}}}{x <^{x_1}_{x_2} f \in \Lambda_{\mathcal{R}}} \; (\mathsf{c} \in \mathcal{R})$$

Figure 2: $\lambda_{\mathcal{R}}$-terms

Example 1 *Pre-terms $\lambda x.y$ and $y <^{y_1}_{y_2} x$ are $\lambda_{\mathcal{R}}$-terms only if $\mathsf{w} \notin \mathcal{R}$. Similarly, pre-terms $\lambda x.xx$ and $x \odot \lambda y.yy$ are $\lambda_{\mathcal{R}}$-terms only if $\mathsf{c} \notin \mathcal{R}$.*

In what follows we use Barendregt's convention [Bar84] 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, $x <^{x_1}_{x_2} M$ 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.

Implicit substitution $M[N/x]$ is defined in Figure 3. In this definition, in case $\mathsf{c} \in \mathcal{R}$, the following condition must be satisfied:

$$Fv(M) \cap Fv(N) = \emptyset,$$

otherwise the substitution result would not be a well-formed term. In the same definition, terms N_1 and N_2 are obtained from N by renaming all free variables in N by fresh variables, and $M[N_1/x_1,N_2/x_2]$ denotes parallel substitution.[3]

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

Figure 3: Substitution in $\lambda_{\mathcal{R}}$-calculi

The operational semantics for four calculi that form the "natural deduction base" of the resource control cube are given in Figures 4 and 5. Reduction rules of $\lambda_{\mathcal{R}}$-calculi are given in Figure 4, whereas equivalences in are given in Figure 5. Reduction rules specific for each calculus are given in Figure 6.

Which reductions in which system?

Generally speaking the reduction rules can been devided into four groups. The main computational step is β reduction represents. The group of (γ) reductions exists only in the two calculi that contain contraction (i.e., if $\mathsf{c} \in \mathcal{R}$). These

[3]Note that the terms N_1 and N_2 do not have any free variables in common hence, it is not a problem to perform the substitution in parallel.

$$
\begin{array}{rrcl}
(\beta) & (\lambda x.M)N & \to & M[N/x] \\
(\gamma_0) & x <^{x_1}_{x_2} y & \to & y \quad y \neq x_1, x_2 \\
(\gamma_0') & x <^{x_1}_{x_2} x_1 & \to & x \\
(\gamma_1) & x <^{x_1}_{x_2} (\lambda y.M) & \to & \lambda y.x <^{x_1}_{x_2} M \\
(\gamma_2) & x <^{x_1}_{x_2} (MN) & \to & (x <^{x_1}_{x_2} M)N, \text{ if } x_1, x_2 \notin Fv(N) \\
(\gamma_3) & x <^{x_1}_{x_2} (MN) & \to & M(x <^{x_1}_{x_2} N), \text{ if } x_1, x_2 \notin Fv(M) \\
(\omega_1) & \lambda x.(y \odot M) & \to & y \odot (\lambda x.M),\ x \neq y \\
(\omega_2) & (x \odot M)N & \to & \{x\} \setminus Fv(N) \odot (MN) \\
(\omega_3) & M(x \odot N) & \to & \{x\} \setminus Fv(M) \odot (MN) \\
(\gamma\omega_1) & x <^{x_1}_{x_2} (y \odot M) & \to & y \odot (x <^{x_1}_{x_2} M),\ y \neq x_1, x_2 \\
(\gamma\omega_2) & x <^{x_1}_{x_2} (x_1 \odot M) & \to & M[x/x_2]
\end{array}
$$

Figure 4: Reduction rules of $\lambda_{\mathcal{R}}$-calculi

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

Figure 5: Equivalences in $\lambda_{\mathcal{R}}$-calculi

$\lambda_{\mathcal{R}}$-calculi	reduction rules	equivalences
$\lambda_{\emptyset}$	β	
λ_{c}	$\beta, \gamma_0, \gamma_0', \gamma_1, \gamma_2, \gamma_3$	$\varepsilon_2, \varepsilon_3, \varepsilon_4$
λ_{w}	$\beta, \omega_1, \omega_2, \omega_3$	ε_1
λ_{cw}	$\beta, \gamma_1, \gamma_2, \gamma_3, \omega_1, \omega_2, \omega_3, \gamma\omega_1, \gamma\omega_2$	$\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4$

Figure 6: ND base of the resource control cube

rules perform propagation of contraction into the expression. Similarly, (ω) reductions belong only to the two calculi containing weakening (i.e., if $\mathsf{w} \in \mathcal{R}$). Their role is to 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, hence these rules exist only if $\mathcal{R} = \{\mathsf{c}, \mathsf{w}\}$. The rules (γ_0) and (γ_0')[4] exist only if $\mathcal{R} = \{\mathsf{c}\}$ and they erase meaningless contractions.

Remark 1 *Notice the asymmetry between the reduction rules of the calculi λ_{c} and λ_{w}, namely in λ_{w} there is no counterpart of the (γ_0) and (γ_0') reductions of λ_{c}. The reason can be tracked back to the definition of $\lambda_{\mathcal{R}}$-terms in Figure 2, where the definition of the weakening operator $x \odot f$ does not depend on the presence of the explicit contraction. It would be possible to define weakening where the condition $x \notin Fv(f)$ is not required if the contraction is implicit. In that case, terms like $x \odot x \odot M$ would exist, and the reduction $(\omega_0) : x \odot M \to M$, if $x \in Fv(M)$ would erase this redundant weakening. The typing rule for this weakening would require multiset treatment of the bases Γ, which is out of the scope of this paper.*

1.2 Resource control sequent lambda calculi $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$

An attempt of creating the sequent-style λ-calculus (i.e., the system corresponding to the Gentzen's *LJ* system) was proposed by Barendregt and Ghilezan in [BG00] by keeping the original syntax of λ-calculus and changing the type assignment rules in accordance with the inference rules of *LJ*. The obtained simply typed λLJ-calculus, although useful for giving a new simpler proof of the Cut-elimination theorem, was not in one-to-one correspondence with *LJ*.

Herbelin realised that, in order to achieve such a correspondence, significant changes in syntax should be made. In [Her95], he proposed $\bar{\lambda}$-calculus with explicit substitution, a new syntactic construct called a list and new operators for creating and manipulating the lists. The role of the lists was to overcome the key difference between natural deduction and sequent calculi, namely the associativity of applications. While in ordinary λ-calculus applications are left-associated, i.e., $(\lambda x.M)N_1N_2...N_k \equiv ((((\lambda x.M)N_1)N_2)...)N_k$, in $\bar{\lambda}$-calculus the application is right-associated, i.e., $(\lambda x.M)[N_1, N_2, ..., N_k] \equiv (\lambda x.M)(N_1(N_2...(N_{k-1}N_k)))$. The $\bar{\lambda}$-calculus was the first formal calculus whose simply typed version extended the Curry-Howard correspondence to the intuitionistic sequent calculus, more precisely, its cut-free restriction LJT^{cf}.

Relying on Herbelin's work, Espírito Santo and Pinto developed λ^{Jm}-calculus with generalised multiary application [EP03] and later λ^{Gtz}-calculus [Esp07a], the calculus that fully corresponds to Gentzen's *LJ*. In the λ^{Gtz}-calculus one can also distinguish two kinds of expressions, namely terms and contexts, the later being the generalisation of the lists from the $\bar{\lambda}$-calculus.

In this section we present the syntax and the operational semantics of the four sequent calculi with explicit or implicit resource control, denoted by $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$, where $\mathcal{R} \subseteq \{\mathsf{c}, \mathsf{w}\}$ and c, w denote explicit contraction and weakening, respectively. These four calculi are sequent counterparts of the four resource control calculi $\lambda_{\mathcal{R}}$ presented above, and represent extensions of Espírito Santo's λ^{Gtz}-calculus.

The abstract syntax of the $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ pre-expressions is the following:

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

where $x, x_1, \ldots, y, \ldots$ range over a denumerable set of term variables.

A *pre-term* can be a variable, an abstraction, a cut (an application), a contraction or a weakening. Note that the application is of the form fc. A *pre-context* is one of the following features: a selection $\widehat{x}.f$, which turns a term into a context by choosing an active variable; a context constructor $f :: c$ (usually called cons) which expands a context by introducing a term into the left-most position; a weakening on a pre-context or a contraction on a pre-context. Pre-terms and pre-contexts are together referred to as *pre-expressions* and will be ranged over by E. Pre-contexts $x \odot c$ and $x <^{x_1}_{x_2} c$ behave exactly like the corresponding pre-terms $x \odot f$ and $x <^{x_1}_{x_2} f$ in the untyped calculi, so they will not be treated separately.

[4] The rules (γ_0) and (γ_0') correspond to the CGc rule in [KR09].

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

$$Fv(x) = x; \quad Fv(\lambda x.f) = Fv(f) \setminus \{x\}; \quad Fv(fc) = Fv(f) \cup Fv(c);$$
$$Fv(\widehat{x}.f) = Fv(f) \setminus \{x\}; \quad Fv(f :: c) = Fv(f) \cup Fv(c);$$
$$[Fv(x \odot E) = \{x\} \cup Fv(E)]_{\mathsf{w}};$$
$$\left[Fv(x <^{x_1}_{x_2} E) = \begin{cases} Fv(E), & \{x_1,x_2\} \cap Fv(E) = \emptyset \\ \{x\} \cup Fv(E) \setminus \{x_1,x_2\}, & \{x_1,x_2\} \cap Fv(E) \neq \emptyset \end{cases}\right]_{\mathsf{c}}.$$

The sets of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}} = \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \cup \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}$ (where $\mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}$ are the sets of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-terms and $\mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}$ are the sets of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-contexts) are the subsets of the set of pre-expressions, defined by the inference rules given in Figure 7. We denote terms by $t,u,v...$, contexts by $k,k',...$ and expressions by e,e'.

$$\frac{}{x \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}}$$

$$\frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad [x \in Fv(f)]_{\mathsf{w}}}{\lambda x.f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}} \qquad \frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}} \quad [Fv(f) \cap Fv(c) = \emptyset]_{\mathsf{c}}}{fc \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}}$$

$$\frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad [x \in Fv(f)]_{\mathsf{w}}}{\widehat{x}.f \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}} \qquad \frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}} \quad [Fv(f) \cap Fv(c) = \emptyset]_{\mathsf{c}}}{f :: c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}}$$

$$\frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad x \notin Fv(f)}{x \odot f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}} \ (\mathsf{w} \in \mathcal{R}) \qquad \frac{c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}} \quad x \notin Fv(c)}{x \odot c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}} \ (\mathsf{w} \in \mathcal{R})$$

$$\frac{f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \quad x_1 \neq x_2 \quad x \notin Fv(f) \setminus \{x_1,x_2\} \quad [x_1,x_2 \in Fv(f)]_{\mathsf{w}}}{x <^{x_1}_{x_2} f \in \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}}} \ (\mathsf{c} \in \mathcal{R})$$

$$\frac{c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}} \quad x_1 \neq x_2 \quad x \notin Fv(c) \setminus \{x_1,x_2\} \quad [x_1,x_2 \in Fv(c)]_{\mathsf{w}}}{x <^{x_1}_{x_2} c \in \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}}} \ (\mathsf{c} \in \mathcal{R})$$

Figure 7: $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions

Example 2 - $\lambda x.x(y :: \widehat{z}.z)$ *belongs to all four sets* $\Lambda^{\mathsf{Gtz}}_{\emptyset}$, $\Lambda^{\mathsf{Gtz}}_{\mathsf{c}}$, $\Lambda^{\mathsf{Gtz}}_{\mathsf{w}}$ *and* $\Lambda^{\mathsf{Gtz}}_{\mathsf{cw}}$;

- $\lambda x.w(y :: \widehat{z}.z)$ *belongs to* $\Lambda^{\mathsf{Gtz}}_{\emptyset}$ *and* $\Lambda^{\mathsf{Gtz}}_{\mathsf{c}}$;
- $\lambda x.x(x :: \widehat{z}.z)$ *belongs to* $\Lambda^{\mathsf{Gtz}}_{\emptyset}$ *and* $\Lambda^{\mathsf{Gtz}}_{\mathsf{w}}$;
- $\lambda x.y(y :: \widehat{z}.z)$ *belongs only to the* $\Lambda^{\mathsf{Gtz}}_{\emptyset}$;
- $\lambda x.x \odot y(y :: \widehat{z}.z)$ *belongs only to* $\Lambda^{\mathsf{Gtz}}_{\mathsf{w}}$;
- $\lambda x.y <^{y_1}_{y_2} y_1(y_2 :: \widehat{z}.z)$ *belongs only to* $\Lambda^{\mathsf{Gtz}}_{\mathsf{c}}$;
- $\lambda x.x \odot y <^{y_1}_{y_2} y_1(y_2 :: \widehat{z}.z)$ *belongs only to* $\Lambda^{\mathsf{Gtz}}_{\mathsf{cw}}$.

The inductive definition of the meta operator of implicit substitution $e[t/x]$, representing the substitution of free variables, is given in Figure 8. In this definition, in case $\mathsf{c} \in \mathcal{R}$, the following condition must be satisfied:

$$Fv(e) \cap Fv(t) = \emptyset,$$

otherwise the substitution result would not be a well-formed term. In the same definition, terms t_1 and t_2 are obtained from t by renaming all free variables in t by fresh variables.

$$
\begin{array}{rcl@{\qquad}rcl}
x[t/x] & \triangleq & t & y[t/x] & \triangleq & y \\
(\lambda y.v)[t/x] & \triangleq & \lambda y.v[t/x],\ x \neq y & (y \odot e)[t/x] & \triangleq & \{y\} \setminus Fv(t) \odot e[t/x],\ x \neq y \\
(vk)[t/x] & \triangleq & v[t/x]\,k[t/x] & (x \odot e)[t/x] & \triangleq & Fv(t) \setminus Fv(e) \odot e \\
(v :: k)[t/x] & \triangleq & v[t/x] :: k[t/x] & (y <^{y_1}_{y_2} e)[t/x] & \triangleq & y <^{y_1}_{y_2} e[t/x],\ x \neq y \\
(\widehat{y}.v)[t/x] & \triangleq & \widehat{y}.v[t/x] & (x <^{x_1}_{x_2} e)[t/x] & \triangleq & Fv(t) <^{Fv(t_1)}_{Fv(t_2)} e[t_1/x_1][t_2/x_2]
\end{array}
$$

Figure 8: Substitution in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi

The computation over the set of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions reflects the cut-elimination process, and manages the explicit/implicit resource control operators. Four groups of reductions in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi are given in Figure 9, while the equivalences are given in Figure 10. Reduction rules and equivalences specific to each of the four term calculi forming the "LJ base" of the resource control cube are given in Figure 11.

$$
\begin{array}{lrcl}
(\beta) & (\lambda x.t)(u :: k) & \to & u(\widehat{x}.tk) \\
(\sigma) & t(\widehat{x}.v) & \to & v[t/x] \\
(\pi) & (tk)k' & \to & t(k@k') \\
(\mu) & \widehat{x}.xk & \to & k \\
(\gamma_0) & x <^{x_1}_{x_2} y & \to & y \quad y \neq x_1, x_2 \\
(\gamma_0{}') & x <^{x_1}_{x_2} x_1 & \to & x \\
(\gamma_1) & x <^{x_1}_{x_2} (\lambda y.t) & \to & \lambda y.x <^{x_1}_{x_2} t \\
(\gamma_2) & x <^{x_1}_{x_2} (tk) & \to & (x <^{x_1}_{x_2} t)k,\ \text{if } x_1, x_2 \notin Fv(k) \\
(\gamma_3) & x <^{x_1}_{x_2} (tk) & \to & t(x <^{x_1}_{x_2} k),\ \text{if } x_1, x_2 \notin Fv(t) \\
(\gamma_4) & x <^{x_1}_{x_2} (\widehat{y}.t) & \to & \widehat{y}.(x <^{x_1}_{x_2} t) \\
(\gamma_5) & x <^{x_1}_{x_2} (t :: k) & \to & (x <^{x_1}_{x_2} t) :: k,\ \text{if } x_1, x_2 \notin Fv(k) \\
(\gamma_6) & x <^{x_1}_{x_2} (t :: k) & \to & t :: (x <^{x_1}_{x_2} k),\ \text{if } x_1, x_2 \notin Fv(t) \\
(\omega_1) & \lambda x.(y \odot t) & \to & y \odot (\lambda x.t),\ x \neq y \\
(\omega_2) & (x \odot t)k & \to & \{x\} \setminus Fv(k) \odot (tk) \\
(\omega_3) & t(x \odot k) & \to & \{x\} \setminus Fv(t) \odot (tk) \\
(\omega_4) & \widehat{x}.(y \odot t) & \to & y \odot (\widehat{x}.t),\ x \neq y \\
(\omega_5) & (x \odot t) :: k & \to & \{x\} \setminus Fv(k) \odot (t :: k) \\
(\omega_6) & t :: (x \odot k) & \to & \{x\} \setminus Fv(t) \odot (t :: k) \\
(\gamma\omega_1) & x <^{x_1}_{x_2} (y \odot e) & \to & y \odot (x <^{x_1}_{x_2} e) \qquad x_1 \neq y \neq x_2 \\
(\gamma\omega_2) & x <^{x_1}_{x_2} (x_1 \odot e) & \to & e[x/x_2]
\end{array}
$$

Figure 9: Reduction rules of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi

The first group consists of (β), (π), (σ) and (μ) reductions that exist in all four $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi. (β) reduction represents the main computational step. It creates a substitution, but it is the rule (σ) that executes it. In that sense, substitution can be controlled (i.e. delayed) although it is implicit. Note that this feature is not present in $\lambda_{\mathcal{R}}$-calculi since it is a consequence of the existence of contexts. For more details see [Esp07b]. Combination of $(\beta)+(\sigma)$ rules corresponds to the traditional (β) reduction in λ-calculus. The rule (π) simplifies the head of a cut (t is the head of tk).

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

Figure 10: Equivalences in $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi

$\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi	reduction rules	equivalences
$\lambda^{\mathsf{Gtz}}_{\emptyset}$	β, π, σ, μ	
$\lambda^{\mathsf{Gtz}}_{\mathsf{c}}$	β, π, σ, μ, γ_0 - γ_6	$\varepsilon_2, \varepsilon_3, \varepsilon_4$
$\lambda^{\mathsf{Gtz}}_{\mathsf{w}}$	β, π, σ, μ, ω_1 - ω_6	ε_1
$\lambda^{\mathsf{Gtz}}_{\mathsf{cw}}$	β, π, σ, μ, γ_1 - γ_6, ω_1 - ω_6, $\gamma\omega_1$, $\gamma\omega_2$	ε_1 - ε_4

Figure 11: LJ base of the resource control cube

The rule (μ) erases the sequence made of a trivial cut (a cut is trivial if its head is a variable) followed by the selection of the same variable. In that sense, it represents a kind of garbage collection.

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

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

If $\mathsf{c} \in \mathcal{R}$, with respect to the $\lambda_{\mathcal{R}}$-calculi, the group of (γ) reductions has the additional three reductions which manage the interaction of contraction with selection and context construction. Also if $\mathsf{w} \in \mathcal{R}$, the group of (ω) reductions has additional three reductions which manage the interaction of weakening with selection and context construction. Finally, the group of $(\gamma\omega)$ reductions has additional two rules which manage in contexts the interaction between explicit resource operators of different nature.

We have presented the syntax and the reduction rules of the $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$, $\mathcal{R} \subseteq \{\mathsf{c}, \mathsf{w}\}$, the family of intuitionistic sequent term calculi. By instantiating $\mathcal{R}$, we obtain the following particular calculi. $\mathcal{R} = \emptyset$ gives us the well-known *lambda Gentzen calculus* λ^{Gtz}, proposed by Espírito Santo [Esp07a], whose simply typed version corresponds to the intuitionistic sequent calculus with the cut and implicit structural rules, according to the Curry-Howard correspondence. By letting $\mathcal{R} = \{\mathsf{c}, \mathsf{w}\}$, we obtain the *resource control lambda Gentzen calculus*, $\lambda^{\mathsf{Gtz}}_{\circledR}$, whose call-by-value version was proposed and investigated in [GILŽ11]. The simply typed $\lambda^{\mathsf{Gtz}}_{\circledR}$ expands the Curry-Howard correspondence to the intuitionistic sequent calculus with the cut and explicit structural rules of weakening and contraction. Finally, in case that $\mathcal{R} = \{\mathsf{c}\}$ and $\mathcal{R} = \{\mathsf{w}\}$ we get two new calculi, namely $\lambda^{\mathsf{Gtz}}_{\mathsf{c}}$ and $\lambda^{\mathsf{Gtz}}_{\mathsf{w}}$. These calculi could be related to the substructural logics, as will be elaborated in the sequel.

2 Simple types for resource control cube

In this section we summarise the type assignment systems that assign *simple types* to all eight calculi of the resource control cube in a uniform way. Simple types for λ^{Gtz}-calculus were introduced by Espírito Santo in [Esp07a]. As far as resource control calculi are concerned, simple types were introduced to λlxr-calculus by Kesner and Lengrand in [KL07] and to resource control lambda Gentzen calculus $\lambda^{\mathsf{Gtz}}_{\circledR}$ in [GILŽ11]. The syntax of types is defined as follows:

$$\text{Simple Types} \qquad \alpha, \beta \ ::= \ p \mid \alpha \to \beta$$

where p ranges over a denumerable set of type atoms and $\alpha \to \beta$ is an arrow type. We denote types by $\alpha, \beta, \gamma, \ldots$.

Definition 1

(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 $\{x_1:\alpha_1,\ldots,x_n:\alpha_n\}$ of basic type assignments, where all term variables are different. $Dom(\Gamma)=\{x_1,\ldots,x_n\}$.

(iii) A basis extension $\Gamma,x:\alpha$ denotes the set $\Gamma\cup\{x:\alpha\}$, where $x\notin Dom(\Gamma)$. Γ,Δ represents the disjoint union of the two bases.

(iv) $\Gamma\cup_{\mathsf{c}}\Delta$ denotes the standard union of the bases, if $\mathsf{c}\notin\mathcal{R}$, and the disjoint union, if $\mathsf{c}\in\mathcal{R}$.

The type assignment systems $\lambda_{\mathcal{R}}\!\!\rightarrow$ for the natural deduction ND-base base of the resource control cube are given in Figure 12.

$$\frac{\mathsf{w}\notin\mathcal{R}}{\Gamma,x:\alpha\vdash_{\mathcal{R}}x:\alpha}\ (Ax_{iw}) \qquad \frac{\mathsf{w}\in\mathcal{R}}{x:\alpha\vdash_{\mathcal{R}}x:\alpha}\ (Ax_{ew})$$

$$\frac{\Gamma,x:\alpha\vdash_{\mathcal{R}}M:\beta}{\Gamma\vdash_{\mathcal{R}}\lambda x.M:\alpha\to\beta}\ (\to_R) \qquad \frac{\Gamma\vdash_{\mathcal{R}}M:\alpha\to\beta\quad\Delta\vdash_{\mathcal{R}}N:\beta}{\Gamma\cup_{\mathsf{c}}\Delta\vdash_{\mathcal{R}}MN:\beta}\ (\to_E)$$

$$\frac{\Gamma,x:\alpha,y:\alpha\vdash_{\mathcal{R}}M:\beta\quad\mathsf{c}\in\mathcal{R}}{\Gamma,z:\alpha\vdash_{\mathcal{R}}z<^{x}_{y}M:\beta}\ (Cont) \qquad \frac{\Gamma\vdash_{\mathcal{R}}M:\beta\quad\mathsf{w}\in\mathcal{R}}{\Gamma,x:\alpha\vdash_{\mathcal{R}}x\odot M:\beta}\ (Weak)$$

Figure 12: $\lambda_{\mathcal{R}}\!\!\rightarrow$: Simply typed $\lambda_{\mathcal{R}}$-calculi

The type assignment systems $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\!\!\rightarrow$ for the sequent LJ-base base of the resource control cube are given in Figure 13. In this case, we distinguish two sorts of type assignments:

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

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

Simply typed $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi correspond to intuitionistic sequent calculus with implicit/explicit structural rules *à la* Curry-Howard. Particularly, $\lambda^{\mathsf{Gtz}}_{\emptyset}\!\!\rightarrow$ and $\lambda^{\mathsf{Gtz}}_{\mathsf{cw}}\!\!\rightarrow$ calculi correspond to the intuitionistic implicative fragments of Kleene's systems G3 and G1 from [Kle52], respectively, except for the fact that the exchange rule is implicit here. The exchange rule could be made explicit by considering the bases as lists instead of sets. The system $\lambda^{\mathsf{Gtz}}_{\mathsf{c}}\!\!\rightarrow$ corresponds to the intuitionistic sequent calculus with explicit contraction and implicit weakening, whereas the system $\lambda^{\mathsf{Gtz}}_{\mathsf{w}}\!\!\rightarrow$ corresponds to the intuitionistic sequent calculus with explicit weakening and implicit contraction.

Modifications of $\lambda_{\mathcal{R}}\!\!\rightarrow$ and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\!\!\rightarrow$ systems can provide the computational interpretation of substructural logics, different from the usual approach via linear logic. For instance, if one combines (Ax_{ew}) axiom and the other rules in $\mathsf{w}\notin\mathcal{R}$ modality, the resulting system would correspond to the logic without weakening i.e. to the variant of relevance logic. Similarly, if we use $\cup_{\mathsf{c}}$ as disjoint union together with the $\mathsf{c}\notin\mathcal{R}$ modality of the rest of the system, correspondence with the variant of the logic without contraction i.e. affine logic is obtained. The properties of these systems will not be investigated in this paper.

Although the systems $\lambda_{\mathcal{R}}\!\!\rightarrow$ and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\!\!\rightarrow$ enjoy the subject reduction and the strong normalisation, they (as expected) do not assign types to all strongly normalising expressions, e.g. $\lambda x.xx$ and $\lambda x.x :: x(\hat{y}.y)$. This is the motivation for introducing intersection types in the next section.

$$\frac{\mathsf{w} \notin \mathcal{R}}{\Gamma, x:\alpha \vdash_{\mathcal{R}} x:\alpha}\ (Ax_{iw}) \qquad \frac{\mathsf{w} \in \mathcal{R}}{x:\alpha \vdash_{\mathcal{R}} x:\alpha}\ (Ax_{ew})$$

$$\frac{\Gamma, x:\alpha \vdash_{\mathcal{R}} t:\beta}{\Gamma \vdash_{\mathcal{R}} \lambda x.t:\alpha\to\beta}\ (\to_R) \qquad \frac{\Gamma \vdash_{\mathcal{R}} t:\alpha \quad \Delta;\beta \vdash_{\mathcal{R}} k:\gamma}{\Gamma \cup_{\mathsf{c}} \Delta;\alpha\to\beta \vdash_{\mathcal{R}} t :: k:\gamma}\ (\to_L)$$

$$\frac{\Gamma, x:\alpha \vdash_{\mathcal{R}} t:\beta}{\Gamma;\alpha \vdash_{\mathcal{R}} \widehat{x}.t:\beta}\ (Sel) \qquad \frac{\Gamma \vdash_{\mathcal{R}} t:\alpha \quad \Delta;\alpha \vdash_{\mathcal{R}} k:\beta}{\Gamma \cup_{\mathsf{c}} \Delta \vdash_{\mathcal{R}} tk:\beta}\ (Cut)$$

$$\frac{\Gamma, x:\alpha, y:\alpha \vdash_{\mathcal{R}} t:\beta \quad \mathsf{c} \in \mathcal{R}}{\Gamma, z:\alpha \vdash_{\mathcal{R}} z <^{x}_{y} t:\beta}\ (Cont_t) \qquad \frac{\Gamma \vdash_{\mathcal{R}} t:\beta \quad \mathsf{w} \in \mathcal{R}}{\Gamma, x:\alpha \vdash_{\mathcal{R}} x \odot t:\beta}\ (Weak_t)$$

$$\frac{\Gamma, x:\alpha, y:\alpha;\beta \vdash_{\mathcal{R}} k:\gamma \quad \mathsf{c} \in \mathcal{R}}{\Gamma, z:\alpha;\beta \vdash_{\mathcal{R}} z <^{x}_{y} k:\gamma}\ (Cont_k) \qquad \frac{\Gamma;\gamma \vdash_{\mathcal{R}} k:\gamma \quad \mathsf{w} \in \mathcal{R}}{\Gamma, x:\alpha;\beta \vdash_{\mathcal{R}} x \odot k:\gamma}\ (Weak_k)$$

Figure 13: $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\!\!\to$: Simply typed $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi

3 Intersection types for resource control cube

In this section we introduce intersection type assignment systems which assign *strict types* to $\lambda_{\mathcal{R}}$-terms and $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions. Intersection types for the λ^{Gtz}-calculus were introduced in [EGI08]. Strict types were proposed in [vB92] and already used in [EIL11] and [GILL11] for characterisation of strong normalisation in λ^{Gtz}-calculus and in $\lambda_{\circledR}$ and $\lambda^{\mathsf{Gtz}}_{\circledR}$-calculi, respectively.

The syntax of types is defined as follows:

$$\begin{array}{lllll} \text{Strict Types} & \sigma & ::= & p \mid \alpha\to\sigma \\ \text{Types} & \alpha & ::= & \cap^{n}_{i}\sigma_i \end{array}$$

where p ranges over a denumerable set of type atoms and $\cap^{n}_{i}\sigma_i = \sigma_1 \cap \ldots \cap \sigma_n$, $n \geq 1$. We denote types by $\alpha,\beta,\gamma\ldots$, strict types by $\sigma,\tau,\rho,\upsilon\ldots$ and the set of all types by Types. We assume that the intersection operator is idempotent, commutative and associative, and that it has priority over the arrow operator. Hence, we will omit parenthesis in expressions like $(\cap^{n}_{i}\tau_i)\to\sigma$.

The definition of a basic type assignment, a basis and a basis extension is analogous to the one given in Section 2. The type assignments are of the form

$$x_1:\alpha_1,\ldots,x_n:\alpha_n \vdash M:\sigma$$

so that only strict types are assigned to terms.

Definition 2

(i) A union of bases with intersection types is defined in the standard way:

$$\begin{array}{rcl} \Gamma \sqcup \Delta & = & \{x:\alpha \mid x:\alpha \in \Gamma \,\&\, x \notin Dom(\Delta)\} \\ & \cup & \{x:\alpha \mid x:\alpha \in \Delta \,\&\, x \notin Dom(\Gamma)\} \\ & \cup & \{x:\alpha\cap\beta \mid x:\alpha \in \Gamma \,\&\, x:\beta \in \Delta\}. \end{array}$$

(ii) $\Gamma \sqcup_{\mathsf{c}} \Delta$ *represents* $\Gamma \sqcup \Delta$*, if* $\mathsf{c} \notin \mathcal{R}$*, and the disjoint union* Γ,Δ *otherwise.*

3.1 Intersection types for $\lambda_{\mathcal{R}}$

The type assignment systems $\lambda_{\mathcal{R}}\cap$ for the natural deduction ND-base base of the resource control cube are given in Figure 14. The rules that correspond to each of the four $\lambda_{\mathcal{R}}\cap$-systems are given in Figure 15.

All systems are syntax-directed i.e. the intersection operator is incorporated into already existing rules of the simply-typed systems. Intersection elimination is managed by the axioms (Ax_{iw}), (Ax_{ew}) and the contraction rule $(Cont)$, whereas the intersection introduction is performed by the arrow elimination rule $(\to_E)$. Notice that in the $(\to_E)$ rule, $Dom(\Delta_1) = \ldots = Dom(\Delta_n)$. The explicit contraction is naturally connected to intersection, because if some data is used twice, once as data of type α and once as data of type β, that data should be of type $\alpha\cap\beta$.

$$\frac{\mathsf{w}\notin\mathcal{R}}{\Gamma,x:\cap_i^n\sigma_i\vdash_{\mathcal{R}} x:\sigma_i}\ (Ax_{iw}) \qquad \frac{\mathsf{w}\in\mathcal{R}}{x:\cap_i^n\sigma_i\vdash_{\mathcal{R}} x:\sigma_i}\ (Ax_{ew})$$

$$\frac{\Gamma,x:\alpha\vdash_{\mathcal{R}} M:\sigma}{\Gamma\vdash_{\mathcal{R}}\lambda x.M:\alpha\to\sigma}\ (\to_I)$$

$$\frac{\Gamma\vdash_{\mathcal{R}} M:\cap_i^n\tau_i\to\sigma \quad \Delta_1\vdash_{\mathcal{R}} N:\tau_1 \quad \ldots \quad \Delta_n\vdash_{\mathcal{R}} N:\tau_n}{\Gamma\sqcup_{\mathsf{c}}(\Delta_1\sqcup\ldots\sqcup\Delta_n)\vdash_{\mathcal{R}} MN:\sigma}\ (\to_E)$$

$$\frac{\Gamma,x:\alpha,y:\beta\vdash_{\mathcal{R}} M:\sigma \quad \mathsf{c}\in\mathcal{R}}{\Gamma,z:\alpha\cap\beta\vdash_{\mathcal{R}} z<_y^x M:\sigma}\ (Cont) \qquad \frac{\Gamma\vdash_{\mathcal{R}} M:\sigma \quad \mathsf{w}\in\mathcal{R}}{\Gamma,x:\alpha\vdash_{\mathcal{R}} x\odot M:\sigma}\ (Weak)$$

Figure 14: $\lambda_{\mathcal{R}}\cap$: $\lambda_{\mathcal{R}}$-calculi with intersection types

$\lambda_{\mathcal{R}}\cap$-systems	type assignment rules
$\lambda_{\emptyset}\cap$	(Ax_{iw}), $(\to_I)$, $(\to_E)$
$\lambda_{\mathsf{c}}\cap$	(Ax_{iw}), $(\to_I)$, $(\to_E)$, $(Cont)$
$\lambda_{\mathsf{w}}\cap$	(Ax_{ew}), $(\to_I)$, $(\to_E)$, $(Weak)$
$\lambda_{\mathsf{cw}}\cap$	(Ax_{ew}), $(\to_I)$, $(\to_E)$, $(Cont)$, $(Weak)$

Figure 15: Four ND intersection type systems

The proposed systems satisfy the following properties.

Proposition 1 (Generation lemma for $\lambda_{\mathcal{R}}\cap$)

(i) *For* $\mathsf{w}\notin\mathcal{R}$*:* $\Gamma\vdash_{\mathcal{R}} x:\tau$ *iff there exist* $\sigma_i, i=1,\ldots,n$ *such that* $x:\tau\cap(\cap_i^n\sigma_i)\in\Gamma$.

(ii) *For* $\mathsf{w}\in\mathcal{R}$*:* $\Gamma\vdash_{\mathcal{R}} x:\tau$ *iff there exist* $\sigma_i, i=1,\ldots,n$ *such that* $x:\tau\cap(\cap_i^n\sigma_i)=\Gamma$.

(iii) $\Gamma\vdash_{\mathcal{R}}\lambda x.M:\tau$ *iff there exist* α *and* σ *such that* $\tau\equiv\alpha\to\sigma$ *and* $\Gamma,x:\alpha\vdash_{\mathcal{R}} M:\sigma$.

(iv) $\Gamma\vdash_{\mathcal{R}} MN:\sigma$ *iff* $\Gamma=\Gamma'\sqcup_{\mathsf{c}}\Delta$, $\Delta=\Delta_1\sqcup\ldots\sqcup\Delta_n$ *and there exist* τ_i, $i=1,\ldots,n$ *such that* $\Gamma'\vdash_{\mathcal{R}} M:\cap_i^n\tau_i\to\sigma$ *and for all* $i\in\{1,\ldots,n\}$, $\Delta_i\vdash_{\mathcal{R}} N:\tau_i$.

(v) $\Gamma\vdash_{\mathcal{R}} x\odot M:\sigma$ *iff there exist* Γ',β *such that* $\Gamma=\Gamma',x:\beta$ *and* $\Gamma'\vdash_{\mathcal{R}} M:\sigma$.

(vi) $\Gamma\vdash_{\mathcal{R}} z<_y^x M:\sigma$ *iff there exist* Γ',α,β *such that* $\Gamma=\Gamma',z:\alpha\cap\beta$ *and* $\Gamma',x:\alpha,y:\beta\vdash_{\mathcal{R}} M:\sigma$.

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

Proposition 2 (Substitution lemma for $\lambda_{\mathcal{R}}\cap$) *If* $\Gamma, x : \cap_i^n \tau_i \vdash_{\mathcal{R}} M : \sigma$ *and for all* $i \in \{1,\ldots,n\}$, $\Delta_i \vdash_{\mathcal{R}} N : \tau_i$, *then* $\Gamma \sqcup_{\mathsf{c}} (\Delta_1 \sqcup \ldots \sqcup \Delta_n) \vdash_{\mathcal{R}} M[N/x] : \sigma$.

Proposition 3 (Subject equivalence for $\lambda_{\mathcal{R}}\cap$) *For every* $\lambda_{\mathcal{R}}$*-term* M*: if* $\Gamma \vdash_{\mathcal{R}} M : \sigma$ *and* $M \equiv M'$, *then* $\Gamma \vdash_{\mathcal{R}} M' : \sigma$.

Proposition 4 (Subject reduction for $\lambda_{\mathcal{R}}\cap$) *For every* $\lambda_{\mathcal{R}}$*-term* M*: if* $\Gamma \vdash_{\mathcal{R}} M : \sigma$ *and* $M \to M'$, *then* $\Gamma \vdash_{\mathcal{R}} M' : \sigma$.

3.2 Intersection types for $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$

The type assignment systems $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$ for the sequent LJ-base base of the resource control cube are given in Figure 16. The rules that correspond to each of the four $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$-systems are given in Figure 17.

As in $\lambda_{\mathcal{R}}\cap$, no new rules are added compared to $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\to$ in order to manage intersection. In the style of sequent calculus, left intersection introduction is managed by the axioms (Ax_{iw}), (Ax_{ew}) and the contraction rules $(Cont_t)$ and $(Cont_k)$, whereas the right intersection introduction is performed by the cut rule (Cut) and left arrow introduction rule $(\to_L)$. In these two rules $Dom(\Gamma_1) = \ldots = Dom(\Gamma_n)$.

In order to explain the rule $(\to_L)$ in more details let us consider a simple case $n = 2$ and $m = 2$.

$$\frac{\Gamma_1 \vdash_{\mathcal{R}} t : \sigma_1 \quad \Gamma_2 \vdash_{\mathcal{R}} t : \sigma_2 \quad \Delta; \tau_1 \cap \tau_2 \vdash_{\mathcal{R}} k : \rho}{(\Gamma_1 \sqcup \Gamma_2) \sqcup_{\mathsf{c}} \Delta; (\sigma_1 \cap \sigma_2 \to \tau_1) \cap (\sigma_1 \cap \sigma_2 \to \tau_2) \vdash_{\mathcal{R}} t :: k : \rho}\ (\to_L)$$

Although one would expect in the stoup $\sigma_1 \cap \sigma_2 \to \tau_1 \cap \tau_2$, this is not a type according to the definition of intersection types at the beginning of this section. Therefore the corresponding type in the stoup is $(\sigma_1 \cap \sigma_2 \to \tau_1) \cap (\sigma_1 \cap \sigma_2 \to \tau_2)$. This difficulty does not exist in natural deduction, because natural deduction is isomorphic to a fragment of λ^{Gtz}, where the selection rule (Sel) is restricted to $\widehat{x}.x$. In that fragment the $(\to_L)$ rule would be simpler with strict types in the stoup. Hence, the presented (Sel) and $(\to_L)$ rules keep the full power of the sequent calculus.

$$\frac{\mathsf{w} \notin \mathcal{R}}{\Gamma, x : \cap_i^n \sigma_i \vdash_{\mathcal{R}} x : \sigma_i}\ (Ax_{iw}) \qquad \frac{\mathsf{w} \in \mathcal{R}}{x : \cap_i^n \sigma_i \vdash_{\mathcal{R}} x : \sigma_i}\ (Ax_{ew})$$

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

$$\frac{\Gamma_1 \vdash_{\mathcal{R}} t : \sigma_1 \quad \ldots \quad \Gamma_n \vdash_{\mathcal{R}} t : \sigma_n \quad \Delta; \cap_j^m \tau_j \vdash_{\mathcal{R}} k : \rho}{(\Gamma_1 \sqcup \ldots \sqcup \Gamma_n) \sqcup_{\mathsf{c}} \Delta; \cap_j^m (\cap_i^n \sigma_i \to \tau_j) \vdash_{\mathcal{R}} t :: k : \rho}\ (\to_L)$$

$$\frac{\Gamma_1 \vdash_{\mathcal{R}} t : \sigma_1 \quad \ldots \quad \Gamma_n \vdash_{\mathcal{R}} t : \sigma_n \quad \Delta; \cap_i^n \sigma_i \vdash_{\mathcal{R}} k : \tau}{(\Gamma_1 \sqcup \ldots \sqcup \Gamma_n) \sqcup_{\mathsf{c}} \Delta \vdash_{\mathcal{R}} tk : \tau}\ (Cut)$$

$$\frac{\Gamma, x : \alpha, y : \beta \vdash_{\mathcal{R}} t : \sigma \quad \mathsf{c} \in \mathcal{R}}{\Gamma, z : \alpha \cap \beta \vdash_{\mathcal{R}} z <_y^x t : \sigma}\ (Cont_t) \qquad \frac{\Gamma \vdash_{\mathcal{R}} t : \sigma \quad \mathsf{w} \in \mathcal{R}}{\Gamma, x : \alpha \vdash_{\mathcal{R}} x \odot t : \sigma}\ (Weak_t)$$

$$\frac{\Gamma, x : \alpha, y : \beta; \gamma \vdash_{\mathcal{R}} k : \sigma \quad \mathsf{c} \in \mathcal{R}}{\Gamma, z : \alpha \cap \beta; \gamma \vdash_{\mathcal{R}} z <_y^x k : \sigma}\ (Cont_k) \qquad \frac{\Gamma; \gamma \vdash_{\mathcal{R}} k : \sigma \quad \mathsf{w} \in \mathcal{R}}{\Gamma, x : \alpha; \gamma \vdash_{\mathcal{R}} x \odot k : \sigma}\ (Weak_k)$$

Figure 16: $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$: $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi with intersection types

The proposed systems satisfy the following properties.

$\lambda_{\mathcal{R}}\cap$-systems	type assignment rules
$\lambda_{\emptyset}\cap$	(Ax_{iw}), $(\to_R)$, $(\to_L)$, (Sel), (Cut)
$\lambda_{\mathsf{c}}\cap$	(Ax_{iw}), $(\to_R)$, $(\to_L)$, (Sel), (Cut), $(Cont_t)$, $(Cont_k)$
$\lambda_{\mathsf{w}}\cap$	(Ax_{ew}), $(\to_R)$, $(\to_L)$, (Sel), (Cut), $(Weak_t)$, $(Weak_k)$
$\lambda_{\mathsf{cw}}\cap$	(Ax_{ew}), $(\to_R)$, $(\to_L)$, (Sel), (Cut), $(Cont_t)$, $(Cont_k)$, $(Weak_t)$, $(Weak_k)$

Figure 17: Four LJ intersection type systems

Proposition 5 (Generation lemma for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$)

(i) *For* $\mathsf{w}\notin\mathcal{R}$: $\Gamma\vdash_{\mathcal{R}} x:\tau$ *iff there exist* $\sigma_i, i=1,\ldots,n$ *such that* $x:\tau\cap(\cap_i^n\sigma_i)\in\Gamma$.

(ii) *For* $\mathsf{w}\in\mathcal{R}$: $\Gamma\vdash_{\mathcal{R}} x:\tau$ *iff there exist* $\sigma_i, i=1,\ldots,n$ *such that* $x:\tau\cap(\cap_i^n\sigma_i)=\Gamma$.

(iii) $\Gamma\vdash_{\mathcal{R}}\lambda x.t:\tau$ *iff there exist* α *and* σ *such that* $\tau\equiv\alpha\to\sigma$ *and* $\Gamma,x:\alpha\vdash_{\mathcal{R}} t:\sigma$.

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

(v) $\Gamma\vdash_{\mathcal{R}} tk:\sigma$ *iff* $\Gamma=\Gamma'\sqcup_{\mathsf{c}}\Delta$, $\Gamma'=\Gamma'_1\sqcup\ldots\sqcup\Gamma'_n$ *and there exist* $\tau_i, i=1,\ldots,n$ *such that for all* $i\in\{1,\ldots,n\}$, *the following holds* $\Gamma'_i\vdash_{\mathcal{R}} t:\tau_i$, *and* $\Delta;\cap_1^n\tau_i\vdash_{\mathcal{R}} k:\sigma$.

(vi) $\Gamma;\gamma\vdash_{\mathcal{R}} t::k:\rho$ *iff* $\Gamma=\Gamma'\sqcup_{\mathsf{c}}\Delta$, $\Gamma'=\Gamma'_1\sqcup\ldots\sqcup\Gamma'_n$, $\gamma\equiv\cap_j^m(\cap_i^n\sigma_i\to\tau_j)$ *and for all* $i\in\{1,\ldots,n\}$, *the following holds* $\Gamma'_i\vdash_{\mathcal{R}} t:\sigma_i$ *and* $\Delta;\cap_j^m\tau_j\vdash k:\rho$.

(vii) $\Gamma\vdash_{\mathcal{R}} x\odot t:\sigma$ *iff there exist* Γ',β *such that* $\Gamma=\Gamma',x:\beta$ *and* $\Gamma'\vdash_{\mathcal{R}} t:\sigma$.

(viii) $\Gamma;\gamma\vdash x\odot k:\sigma$ *iff there exist* Γ,β *such that* $\Gamma=\Gamma',x:\beta$ *and* $\Gamma;\gamma\vdash_{\mathcal{R}} k:\sigma$.

(ix) $\Gamma\vdash_{\mathcal{R}} z<^x_y t:\sigma$ *iff there exist* Γ',α,β *such that* $\Gamma=\Gamma',z:\alpha\cap\beta$ *and* $\Gamma',x:\alpha,y:\beta\vdash_{\mathcal{R}} t:\sigma$.

(x) $\Gamma;\varepsilon\vdash_{\mathcal{R}} z<^x_y k:\sigma$ *iff there exist* Γ',α,β *such that* $\Gamma=\Gamma',z:\alpha\cap\beta$ *and* $\Gamma,x:\alpha,y:\beta;\varepsilon\vdash_{\mathcal{R}} k:\sigma$.

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

Proposition 6 *If* $e\to e'$, *then* $Fv(e)=Fv(e')$.

Proposition 7

(i) *If* $\mathsf{w}\in\mathcal{R}$ *and* $\Gamma\vdash_{\mathcal{R}} t:\sigma$, *then* $Dom(\Gamma)=Fv(t)$.

(ii) *If* $\mathsf{w}\in\mathcal{R}$ *and* $\Gamma;\alpha\vdash_{\mathcal{R}} k:\sigma$, *then* $Dom(\Gamma)=Fv(k)$.

Proposition 8 (Substitution lemma for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$)

(i) *If* $\Gamma,x:\cap_i^n\tau_i\vdash_{\mathcal{R}} t:\sigma$ *and for all* i, $\Delta_i\vdash_{\mathcal{R}} u:\tau_i$, *then* $\Gamma\sqcup_{\mathsf{c}}(\Delta_1\sqcup\ldots\sqcup\Delta_n)\vdash_{\mathcal{R}} t[u/x]:\sigma$.

(ii) *If* $\Gamma,x:\cap_i^n\tau_i;\alpha\vdash_{\mathcal{R}} k:\sigma$ *and for all* i, $\Delta_i\vdash_{\mathcal{R}} u:\tau_i$, *then* $\Gamma\sqcup_{\mathsf{c}}(\Delta_1\sqcup\ldots\sqcup\Delta_n);\alpha\vdash_{\mathcal{R}} k[u/x]:\sigma$.

Proposition 9 (Append lemma) *If* $\Gamma;\alpha\vdash_{\mathcal{R}} k:\tau_i$ *for all* i, *and* $\Delta;\cap_i^n\tau_i\vdash_{\mathcal{R}} k':\sigma$, *then* $\Gamma\sqcup_{\mathsf{c}}\Delta;\alpha\vdash_{\mathcal{R}} k@k':\sigma$.

Proposition 10 (Subject equivalence for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$)

(i) *For every* $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$*-term* t: *if* $\Gamma\vdash_{\mathcal{R}} t:\sigma$ *and* $t\equiv t'$, *then* $\Gamma\vdash_{\mathcal{R}} t':\sigma$.

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

Proposition 11 (Subject reduction for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$)

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

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

Proof: The property is stable by context. Therefore we can assume that the reduction takes place at the outermost position of the term t (resp. of the context k). Here we just show several cases. We will use GL as an abbreviation for Generation lemma for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$ (Lemma 5).
Case (β): Let $\Gamma \vdash_{\mathcal{R}} (\lambda x.t)(u :: k) : \sigma$. We show that $\Gamma' \vdash_{\mathcal{R}} u(\widehat{x}.tk) : \sigma$.
From $\Gamma \vdash_{\mathcal{R}} (\lambda x.t)(u :: k) : \sigma$ and from GL(v) it follows that $\Gamma = \Gamma'' \sqcup_{\mathsf{c}} \Delta$, $\Gamma'' = \Gamma''_1 \sqcup ... \sqcup \Gamma''_m$ and that there is a type $\cap^m_j \tau_j$ such that for all $j = 1,\dots,m$, $\Gamma''_j \vdash_{\mathcal{R}} \lambda x.t : \tau_j$, and $\Delta;\cap^m_j \tau_j \vdash_{\mathcal{R}} u :: k : \sigma$. From GL(iii) we have that for all $j = 1,\dots,m$, $\tau_j \equiv \alpha_j \to \rho_j$ and $\Gamma''_j, x : \alpha_j \vdash_{\mathcal{R}} t : \rho_j$. From GL(vi) it follows that $\Delta = \Delta' \sqcup_{\mathsf{c}} \Delta''$, $\Delta' = \Delta'_1 \sqcup ... \sqcup \Delta'_n$, $\Delta'_i \vdash_{\mathcal{R}} u : \sigma_i$ and $\Delta'';\cap^m_j \rho_j \vdash_{\mathcal{R}} k : \sigma$, for $\cap^m_j \tau_j \equiv \cap^m_j (\cap^n_i \sigma_i \to \rho_j)$. Also, we conclude that $\alpha_j \equiv \cap^n_i \sigma_i$. Now,

$$\dfrac{\dfrac{\Delta'_1 \vdash_{\mathcal{R}} u : \sigma_1 \;...\; \Delta'_n \vdash_{\mathcal{R}} u : \sigma_n \qquad \dfrac{\Gamma''_1, x : \cap^n_i \sigma_i \vdash_{\mathcal{R}} t : \rho_1 \;...\; \Gamma''_m, x : \cap^n_i \sigma_i \vdash_{\mathcal{R}} t : \rho_m \quad \Delta'';\cap^m_j \rho_j \vdash k : \sigma}{\dfrac{\Gamma'' \sqcup_{\mathsf{c}} \Delta'', x : \cap^n_i \sigma_i \vdash_{\mathcal{R}} tk : \sigma}{\Gamma'' \sqcup_{\mathsf{c}} \Delta'';\cap^n_i \sigma_i \vdash_{\mathcal{R}} \widehat{x}.tk : \sigma}\,(Sel)}\,(Cut)}{\Gamma'' \sqcup_{\mathsf{c}} \Delta'' \sqcup_{\mathsf{c}} (\Delta'_1 \sqcup ... \sqcup \Delta'_n) \vdash_{\mathcal{R}} u(\widehat{x}.tk) : \sigma.}\,(Cut)$$

which is exactly what we wanted, since $\Gamma = \Gamma'' \sqcup_{\mathsf{c}} \Delta'' \sqcup_{\mathsf{c}} (\Delta'_1 \sqcup ... \sqcup \Delta'_n)$.
Case (γ_6): Let $\Gamma, x : \alpha;\beta \vdash_{\mathcal{R}} x <^{x_1}_{x_2} (t :: k) : \sigma$. We show that $\Gamma, x : \alpha;\beta \vdash t :: x <^{x_1}_{x_2} k : \sigma$.
From $\Gamma, x : \alpha;\beta \vdash_{\mathcal{R}} x <^{x_1}_{x_2} (t :: k) : \sigma$ by GL(x) we have that $\alpha \equiv \gamma \cap \delta$ and $\Gamma, x_1 : \gamma, x_2 : \delta;\beta \vdash_{\mathcal{R}} (t :: k) : \sigma$. Next, GL(vi) and the reduction side-condition $x_1, x_2 \notin Fv(t)$ imply that $\Gamma = \Gamma'_1 \sqcup ... \sqcup \Gamma'_n, \Gamma''$, $\beta \equiv \cap^m_j (\cap^n_i \tau_i \to \rho_j)$, $\Gamma'_i \vdash_{\mathcal{R}} t : \tau_i$ for $i = 1...n$ and $\Gamma'', x_1 : \gamma, x_2 : \delta;\cap^m_j \rho_j \vdash_{\mathcal{R}} k : \sigma$. Now,

$$\dfrac{\Gamma'_1 \vdash_{\mathcal{R}} t : \tau_1 \;...\; \Gamma'_n \vdash_{\mathcal{R}} t : \tau_n \qquad \dfrac{\Gamma'', x_1 : \gamma, x_2 : \delta;\cap^m_j \rho_j \vdash_{\mathcal{R}} k : \sigma}{\Gamma'', x : \gamma \cap \delta;\cap^m_j \rho_j \vdash_{\mathcal{R}} x <^{x_1}_{x_2} k : \sigma}\,(Cont_k)}{\Gamma'_1 \sqcup ... \sqcup \Gamma'_n, \Gamma'', x : \gamma \cap \delta;\cap^m_j (\cap^n_i \tau_i \to \rho_j) \vdash_{\mathcal{R}} t :: x <^{x_1}_{x_2} k : \sigma}\,(\to_L)$$

which is exactly what we needed.
Case (ω_4): Let $\Gamma, y : \alpha;\beta \vdash_{\mathcal{R}} \widehat{x}.y \odot t : \sigma$. We show that $\Gamma, y : \alpha;\beta \vdash_{\mathcal{R}} y \odot \widehat{x}.t : \sigma$.
From $\Gamma, y : \alpha;\beta \vdash_{\mathcal{R}} \widehat{x}.y \odot t : \sigma$ by GL(iv) we have that $\Gamma, y : \alpha, x : \beta \vdash_{\mathcal{R}} y \odot t : \sigma$ and afterwards by GL(viii) that $\Gamma, x : \beta \vdash_{\mathcal{R}} t : \sigma$. Now,

$$\dfrac{\dfrac{\Gamma, x : \beta \vdash_{\mathcal{R}} t : \sigma}{\Gamma;\beta \vdash_{\mathcal{R}} \widehat{x}.t : \sigma}\,(Sel)}{\Gamma, y : \alpha;\beta \vdash_{\mathcal{R}} y \odot \widehat{x}.t : \sigma.}\,(Weak_k)$$

□

4 Typeability ⇒ SN in all systems of the resource control cube

4.1 Typeability ⇒ SN in $\lambda_{\mathcal{R}}\cap$

The *reducibility method* is a well known approach for proving reduction properties of λ-terms typeable in different type assignment systems. It was introduced by Tait [Tai67] for proving the strong normalisation property of simply typed λ-calculus. It was developed further to prove *strong normalisation property* of various calculi in [Tai75, Gir71, Kri90, Ghi96], *confluence* of βη-reduction in [Kol85, Sta85] and to characterise certain classes of λ-terms such as strongly normalising, normalising, head normalising, and weak head normalising terms by their typeability in various intersection type systems in [Gal98, DCHM00, DCG03, DCGL04].

The principal idea of the reducibility method is to connect the terms typeable in a certain type assignment system with the terms satisfying certain reduction properties (e.g., strong normalisation, confluence). To this aim, types are interpreted by suitable sets of lambda terms which satisfy certain realizability properties. Then the soundness of type assignment with respect to these interpretations is obtained. As a consequence of soundness, every typeable term belongs to the interpretation of its type, and as such satisfies a desired reduction property.

In the remainder of the paper we consider $\Lambda_{\mathcal{R}}$ to be the applicative structures whose domains are $\lambda_{\mathcal{R}}$-terms and where the application is just the application of $\lambda_{\mathcal{R}}$-terms. In addition, $\Lambda^{\circ}_{\mathcal{R}}$ is the set of *closed* $\lambda_{\mathcal{R}}$-terms. We first recall some notions from [Bar92].

Definition 3 *A $\lambda_{\mathcal{R}}$-term M is called* strongly normalizing *if and only if all reduction sequences starting with M terminate. By a* reduction sequence *we mean a sequence of terms $(M_i)_{i\geq 0}$ such that $M_i \to M_{i+1}$ or $M_i \equiv M_{i+1}$. We denote the set of all strongly normalizing $\lambda_{\mathcal{R}}$-terms with $\mathcal{SN}_{\mathcal{R}}$ and the set of all closed strongly normalizing $\lambda_{\mathcal{R}}$-terms with $\mathcal{SN}^{\circ}_{\mathcal{R}}$.*

Definition 4 *For $\mathcal{M},\mathcal{N} \subseteq \Lambda^{\circ}_{\mathcal{R}}$, we define $\mathcal{M} \longrightarrow \mathcal{N} \subseteq \Lambda^{\circ}_{\mathcal{R}}$ as*

$$\mathcal{M} \longrightarrow \mathcal{N} = \{M \in \Lambda^{\circ}_{\mathcal{R}} \mid \forall N \in \mathcal{M} \quad MN \in \mathcal{N}\}.$$

First of all, we introduce the following notion of *type interpretation*.

Definition 5 (Type interpretation) *The type interpretation $[\![-]\!]^{\mathcal{R}} : Types \to 2^{\Lambda^{\circ}_{\mathcal{R}}}$ is defined by:*

(I1) $[\![p]\!]^{\mathcal{R}} = \mathcal{SN}^{\circ}_{\mathcal{R}}$, *where p is a type atom;*

(I2) $[\![\cap^n_i \sigma_i]\!]^{\mathcal{R}} = \cap^n_i [\![\sigma_i]\!]^{\mathcal{R}}$;

(I3) $[\![\alpha \to \sigma]\!]^{\mathcal{R}} = [\![\alpha]\!]^{\mathcal{R}} \longrightarrow [\![\sigma]\!]^{\mathcal{R}}$.

Next, we introduce the notion of *saturation property*, obtained by modifying the saturation property given in [Bar92].

Definition 6 *A set $X \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}$ is called* $\mathcal{R}$-saturated *if it satisfies the following two properties:*

- $INH(X)$: $(\exists n \geq 0)\ (\forall p \geq n)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in X$.
- $SAT_{\beta}(X)$: $(\forall n \geq 0)\,(\forall M_1,\ldots,M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}})\,(\forall N \in \mathcal{SN}^{\circ}_{\mathcal{R}})$
  $M[N/x]M_1 \ldots M_n \in X \ \Rightarrow\ (\lambda x.M)NM_1 \ldots M_n \in X$.

Proposition 12 *Let $\mathcal{M},\mathcal{N} \subseteq \Lambda^{\circ}_{\mathcal{R}}$.*

(i) $\mathcal{SN}^{\circ}_{\mathcal{R}}$ *is $\mathcal{R}$-saturated.*

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

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

(iv) For all types $\varphi \in \mathsf{Types}$, $[\![\varphi]\!]^{\mathcal{R}}$ is $\mathcal{R}$-saturated.

Proof:

(i) $\mathcal{SN}^{\circ}_{\mathcal{R}} \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}$ and the condition $\mathsf{INH}(\mathcal{SN}^{\circ}_{\mathcal{R}})$ trivially hold.
For $\mathsf{SAT}_{\beta}(\mathcal{SN}^{\circ}_{\mathcal{R}})$, suppose that

$$M[N/x]M_1 \ldots M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}} \text{ and } N,M_1,\ldots,M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}.$$

We have to prove that

$$(\lambda x.M)NM_1 \ldots M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}.$$

Since $M[N/x]$ is a subterm of a term in $\mathcal{SN}^{\circ}_{\mathcal{R}}$, we know that $M \in \mathcal{SN}^{\circ}_{\mathcal{R}}$. By assumption, $N, M_1, \ldots, M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}$, so the reductions inside of these terms terminate. After finitely many reduction steps, we obtain

$$(\lambda x.M)NM_1 \ldots M_n \to \ldots \to (\lambda x.M')N'M'_1 \ldots M'_n$$

where $M \to M',\ N \to N',\ M_1 \to M'_1, \ldots, M_n \to M'_n$. After contracting $(\lambda x.M')N'M'_1 \ldots M'_n$ to $M'[N'/x]M'_1 \ldots M'_n$, we obtain a reduct of $M[N/x]M_1 \ldots M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}$. Hence, also the initial term $(\lambda x.M)NM_1 \ldots M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}$.

(ii) First, we prove that $\mathcal{M} \longrightarrow \mathcal{N} \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}$. Suppose that $M \in \mathcal{M} \longrightarrow \mathcal{N}$. Then, for all $N \in \mathcal{M}$, $MN \in \mathcal{N}$. Since $\mathcal{M}$ is $\mathcal{R}$-saturated, $\mathsf{INH}(\mathcal{M})$ holds and we have that

$$(\exists n \geq 0)\ (\forall p \geq n)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in \mathcal{M}, \text{ hence}$$

$$(\exists n \geq 0)\ (\forall p \geq n)\ M(\lambda x_1 \ldots \lambda x_p.\lambda y.y) \in \mathcal{N} \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}.$$

From here we can deduce that $M \in \mathcal{SN}^{\circ}_{\mathcal{R}}$.

Next, we prove that $\mathsf{INH}(\mathcal{M} \longrightarrow \mathcal{N})$ holds i.e.

$$(\exists n \geq 0)\ (\forall p \geq n)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in \mathcal{M} \longrightarrow \mathcal{N}.$$

By the induction hypothesis $\mathcal{N}$ is $\mathcal{R}$-saturated, hence $\mathsf{INH}(\mathcal{N})$ and $\mathsf{SAT}(\mathcal{N})$ hold. Since $\mathsf{INH}(\mathcal{N})$ holds, we have that

$$(\exists n \geq 0)\ (\forall p \geq n)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in \mathcal{N}.$$

By taking $p = n+1$ we obtain that for all $N \in \mathcal{M}$,

$$(\lambda x_1 \ldots \lambda x_n \lambda x_{n+1}.\lambda y.y)N \to \lambda x_1 \ldots \lambda x_n \lambda y.y$$

By $\mathsf{INH}(\mathcal{N})$ we get that $\lambda x_1 \ldots \lambda x_n \lambda y.y \in \mathcal{N}$ and then by $\mathsf{SAT}(N)$ we get that $(\lambda x_1 \ldots \lambda x_n \lambda x_{n+1}.\lambda y.y)N \in \mathcal{N}$.

$$(\lambda x_1 \ldots \lambda x_n \lambda x_{n+1}.\lambda y.y) \in \mathcal{M} \longrightarrow \mathcal{N}.$$

Similarly, we can prove the above property for all $p \geq n+1$, i.e.

$$(\lambda x_1 \ldots \lambda x_p.\lambda y.y) \in \mathcal{M} \longrightarrow \mathcal{N}.$$

Finally, for $\mathsf{SAT}_{\beta}(\mathcal{M} \longrightarrow \mathcal{N})$, suppose that

$$M[N/x]M_1 \ldots M_n \in \mathcal{M} \longrightarrow \mathcal{N} \text{ and } N, M_1, \ldots, M_n \in \mathcal{SN}^{\circ}_{\mathcal{R}}.$$

This means that for all $P \in \mathcal{M}$

$$M[N/x]M_1 \ldots M_n P \in \mathcal{N}.$$

But $\mathcal{N}$ is $\mathcal{R}$-saturated, so $\mathsf{SAT}_{\beta}(\mathcal{N})$ holds and we have that for all $P \in \mathcal{N}$

$$(\lambda x.M)NM_1 \ldots M_n P \in \mathcal{N}$$

This means that $(\lambda x.M)NM_1 \ldots M_n \in \mathcal{M} \longrightarrow \mathcal{N}$.

(iii) $\mathcal{M} \cap \mathcal{N} \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}$ is straightforward.
For $\mathsf{INH}(\mathcal{M} \cap \mathcal{N})$, since $\mathsf{INH}(\mathcal{M})$ and $\mathsf{INH}(\mathcal{N})$ hold we have that

$$(\exists n_1 \geq 0)\ (\forall p_1 \geq n_1)\ \lambda x_1 \ldots \lambda x_{p_1}.\lambda y.y \in \mathcal{M}$$

$$(\exists n_2 \geq 0)\ (\forall p_2 \geq n_2)\ \lambda x_1 \ldots \lambda x_{p_2}.\lambda y.y \in \mathcal{N}.$$

By taking $n = max(n_1, n_2)$ we obtain

$$(\exists n \geq 0)\ (\forall p \geq n)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in \mathcal{M} \cap \mathcal{N}, \text{ i.e. } \mathsf{INH}(\mathcal{M} \cap \mathcal{N}) \text{ holds.}$$

$\mathsf{SAT}_{\beta}(\mathcal{M} \cap \mathcal{N})$ is straightforward.

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

- If $\varphi \equiv p$, p is type atom, then $[\![\varphi]\!]^{\mathcal{R}} = \mathcal{SN}^{\circ}_{\mathcal{R}}$, so it is $\mathcal{R}$-saturated using (i).
- If $\varphi \equiv \alpha \to \sigma$, then $[\![\varphi]\!]^{\mathcal{R}} = [\![\alpha]\!]^{\mathcal{R}} \longrightarrow [\![\sigma]\!]^{\mathcal{R}}$. Since $[\![\alpha]\!]^{\mathcal{R}}$ and $[\![\sigma]\!]^{\mathcal{R}}$ are $\mathcal{R}$-saturated by IH, we can use (ii).
- If $\varphi \equiv \cap_i^n \sigma_i$, then $[\![\varphi]\!]^{\mathcal{R}} = [\![\cap_i^n \sigma_i]\!]^{\mathcal{R}} = \cap_i^n [\![\sigma_i]\!]^{\mathcal{R}}$ and for all $i = 1, \ldots, n$, $[\![\sigma_i]\!]^{\mathcal{R}}$ are $\mathcal{R}$-saturated by IH, so we can use (iii).

□

We further define a *valuation of terms* $[\![-]\!]^{\mathcal{R}}_{\rho} : \Lambda_{\mathcal{R}} \to \Lambda^{\circ}_{\mathcal{R}}$ and the *semantic satisfiability relation* $\models_{\mathcal{R}}$ which connects the type interpretation with the term valuation.

Definition 7 *Let* $\rho : \mathtt{var} \to \Lambda^{\circ}_{\mathcal{R}}$ *be a valuation of term variables in* $\Lambda^{\circ}_{\mathcal{R}}$*. Then, for the term* $M \in \Lambda_{\mathcal{R}}$ *with free variables* $Fv(M) = \{x_1, \ldots, x_n\}$*, the* term valuation $[\![-]\!]^{\mathcal{R}}_{\rho} : \Lambda_{\mathcal{R}} \to \Lambda^{\circ}_{\mathcal{R}}$ *is defined as follows*

$$[\![M]\!]^{\mathcal{R}}_{\rho} = M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n].$$

Lemma 1

(i) $[\![MN]\!]^{\mathcal{R}}_{\rho} \equiv [\![M]\!]^{\mathcal{R}}_{\rho} [\![N]\!]^{\mathcal{R}}_{\rho}$.

(ii) $[\![\lambda x.M]\!]^{\mathcal{R}}_{\rho} N \to [\![M]\!]^{\mathcal{R}}_{\rho(N/x)}$, *where* $N \in \Lambda^{\circ}_{\mathcal{R}}$.

(iii) $[\![x \odot M]\!]^{\mathcal{R}}_{\rho} \equiv [\![M]\!]^{\mathcal{R}}_{\rho}$.

(iv) $[\![z <^x_y M]\!]^{\mathcal{R}}_{\rho} \equiv [\![M]\!]^{\mathcal{R}}_{\rho(N/x,N/y)}$ *where* $N = \rho(z) \in \Lambda^{\circ}_{\mathcal{R}}$.

Proof:

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

(ii) If $Fv(\lambda x.M) = \{x_1, \ldots, x_n\}$, then
$[\![\lambda x.M]\!]^{\mathcal{R}}_{\rho} N \equiv (\lambda x.M)[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n]N \to (M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n])[N/x] \equiv$
$M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n, N/x]$.
The last equivalence holds, since all $\rho(x_i)$ are closed terms so $x \notin FV(\rho(x_i))$.

(iii) If $Fv(M) = \{x_1, \ldots, x_n\}$, then
$[\![x \odot M]\!]^{\mathcal{R}}_{\rho} \equiv (x \odot M)[\rho(x)/x, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] \equiv$
$Fv(\rho(x)) \odot M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] \equiv [\![M]\!]^{\mathcal{R}}_{\rho}$,
since $Fv(\rho(x)) = \emptyset$ because $\rho(x) \in \Lambda^{\circ}_{\mathcal{R}}$.

(iv) If $Fv(M) = \{x_1, \ldots, x_n\}$ and we denote $\rho(z) = N \in \Lambda^{\circ}_{\mathcal{R}}$, then
$[\![z <^x_y M]\!]^{\mathcal{R}}_{\rho} \equiv (z <^x_y M)[\rho(z)/z, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] \equiv$
$Fv(N) <^{Fv(N_1)}_{Fv(N_2)} M[N_1/x, N_2/y, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] \equiv$
$M[N/x, N/y, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n] \equiv [\![M]\!]^{\mathcal{R}}_{\rho(N/x,N/y)}$
since $N \in \Lambda^{\circ}_{\mathcal{R}}$, hence $Fv(N) = \emptyset$, $N_1 = N_2 = N$ (no renaming takes place) and $Fv(N_1) = Fv(N_2) = \emptyset$.

□

Definition 8

(i) $\rho \models_{\mathcal{R}} M : \alpha \quad \Longleftrightarrow \quad [\![M]\!]^{\mathcal{R}}_{\rho} \in [\![\alpha]\!]^{\mathcal{R}}$;

(ii) $\rho \models_{\mathcal{R}} \Gamma \iff (\forall (x:\alpha) \in \Gamma) \quad \rho(x) \in [\![\alpha]\!]^{\mathcal{R}}$;

(iii) $\Gamma \models_{\mathcal{R}} M : \alpha \iff (\forall \rho) \quad (\rho \models_{\mathcal{R}} \Gamma \Rightarrow \rho \models_{\mathcal{R}} M : \alpha)$.

Proposition 13 (Soundness of $\lambda_{\mathcal{R}}\cap$) *If* $\Gamma \vdash_{\mathcal{R}} M : \alpha$, *then* $\Gamma \models_{\mathcal{R}} M : \alpha$.

Proof: By induction on the derivation of $\Gamma \vdash_{\mathcal{R}} M : \alpha$.

- The last rule applied is (Ax_{iw}). Suppose that $\rho \models \Gamma, x : \cap_1^n \sigma_i$. From here we deduce that $\rho(x) \in [\![\cap_1^n \sigma_i]\!]^{\mathcal{R}} = \cap_1^n [\![\sigma_i]\!]^{\mathcal{R}}$ which means that for all $i \in \{1,\ldots,n\}, \rho(x) \in [\![\sigma_i]\!]^{\mathcal{R}}$.

- (Ax_{ew}). Similar to the previous case.

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

$$\Gamma, x : \alpha \vdash_{\mathcal{R}} M : \sigma \Rightarrow \Gamma \vdash_{\mathcal{R}} \lambda x.M : \alpha \rightarrow \sigma.$$

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

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

$$\forall N \in [\![\alpha]\!]^{\mathcal{R}}.\ [\![\lambda x.M]\!]^{\mathcal{R}}_{\rho} N \in [\![\sigma]\!]^{\mathcal{R}}.$$

  Suppose that $N \in [\![\alpha]\!]^{\mathcal{R}}$. Then, let us consider a new valuation $\rho' = \rho(N/x)$, which can be constructed because N is a closed term. We have that $\rho' \models_{\mathcal{R}} \Gamma, x : \alpha$ since $\rho \models_{\mathcal{R}} \Gamma$, $x \notin \Gamma$ and $\rho'(x) = N \in [\![\alpha]\!]^{\mathcal{R}}$. Then by the IH $\rho' \models_{\mathcal{R}} M : \sigma$, hence we can conclude that $[\![M]\!]^{\mathcal{R}}_{\rho'} \in [\![\sigma]\!]^{\mathcal{R}}$. Applying Lemma 1(ii) we get $[\![\lambda x.M]\!]^{\mathcal{R}}_{\rho} N \rightarrow [\![M]\!]^{\mathcal{R}}_{\rho'}$. Since $[\![M]\!]^{\mathcal{R}}_{\rho'} \in [\![\sigma]\!]^{\mathcal{R}}$ and $[\![\sigma]\!]^{\mathcal{R}}$ is saturated, we obtain $[\![\lambda x.M]\!]^{\mathcal{R}}_{\rho} N \in [\![\sigma]\!]^{\mathcal{R}}$.

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

$$\Gamma \vdash_{\mathcal{R}} M : \cap_1^n \tau_i \rightarrow \sigma,\ \Delta_1 \vdash_{\mathcal{R}} N : \tau_1\ \ \ldots\ \ \Delta_n \vdash_{\mathcal{R}} N : \tau_n \Rightarrow \Gamma \sqcup_c (\Delta_1 \sqcup \ldots \sqcup \Delta_n) \vdash_{\mathcal{R}} MN : \sigma.$$

  By the IH $\Gamma \models_{\mathcal{R}} M : \cap_1^n \tau_i \rightarrow \sigma$, $\Delta_1 \models_{\mathcal{R}} N : \tau_1, \ldots, \Delta_n \models_{\mathcal{R}} N : \tau_n$. Suppose that $\rho \models_{\mathcal{R}} \Gamma \sqcup_c (\Delta_1 \sqcup \ldots \sqcup \Delta_n)$, in order to show $[\![MN]\!]^{\mathcal{R}}_{\rho} \in [\![\sigma]\!]^{\mathcal{R}}$. Then for all $x : \alpha \in \Gamma \sqcup_c (\Delta_1 \sqcup \ldots \sqcup \Delta_n)$, $\rho(x) \in [\![\alpha]\!]^{\mathcal{R}}$. This means that
  if $x : \alpha \in \Gamma$ and $x \notin Dom(\Delta_1) \cap \ldots \cap Dom(\Delta_n)$, then $\rho(x) \in [\![\alpha]\!]^{\mathcal{R}}$;
  if there is i, such that $x : \alpha_i \in \Delta_i$ and $x \notin Dom(\Gamma) \cap Dom(\Delta_j)$, $j \neq i$, then $\rho(x) \in [\![\alpha_i]\!]^{\mathcal{R}}$;
  if $x : \alpha \cap \alpha_1 \cap \ldots \cap \alpha_n$ and $x : \alpha \in \Gamma$, $x : \alpha_1 \in \Delta_1, \ldots, x : \alpha_n \in \Delta_n$, then $\rho(x) \in [\![\alpha]\!]^{\mathcal{R}} \cap [\![\alpha_1]\!]^{\mathcal{R}} \cap \ldots \cap [\![\alpha_n]\!]^{\mathcal{R}}$.
  From this we deduce that $\rho \models_{\mathcal{R}} \Gamma, \rho \models_{\mathcal{R}} \Delta_1, \ldots, \rho \models_{\mathcal{R}} \Delta_n$. From $\rho \models_{\mathcal{R}} \Gamma$, using the IH we deduce that $[\![M]\!]^{\mathcal{R}}_{\rho} \in [\![\cap_1^n \tau_i \rightarrow \sigma]\!]^{\mathcal{R}} = [\![\cap_1^n \tau_i]\!]^{\mathcal{R}} \rightarrow [\![\sigma]\!]^{\mathcal{R}}$. From $\rho \models_{\mathcal{R}} \Delta_1, \ldots, \rho \models_{\mathcal{R}} \Delta_n$, using the IH we deduce that $[\![N]\!]^{\mathcal{R}}_{\rho} \in [\![\tau_1]\!]^{\mathcal{R}}, \ldots, [\![N]\!]^{\mathcal{R}}_{\rho} \in [\![\tau_n]\!]^{\mathcal{R}}$. This means that $[\![N]\!]^{\mathcal{R}}_{\rho} \in \cap_i^n [\![\tau_i]\!]^{\mathcal{R}}_{\rho} = [\![\cap_i^n \tau_i]\!]^{\mathcal{R}}_{\rho}$. Using Lemma 1(i) we obtain that $[\![M]\!]^{\mathcal{R}}_{\rho} [\![N]\!]^{\mathcal{R}}_{\rho} = [\![MN]\!]^{\mathcal{R}}_{\rho} \in [\![\sigma]\!]^{\mathcal{R}}$.

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

$$\Gamma \vdash_{\mathcal{R}} M : \alpha \Rightarrow \Gamma, x : \beta \vdash_{\mathcal{R}} x \odot M : \alpha.$$

  By the IH $\Gamma \models_{\mathcal{R}} M : \alpha$. Suppose that $\rho \models_{\mathcal{R}} \Gamma, x : \beta \Leftrightarrow \rho \models_{\mathcal{R}} \Gamma$ and $\rho \models_{\mathcal{R}} x : \beta$. From $\rho \models_{\mathcal{R}} \Gamma$ and $\Gamma \models_{\mathcal{R}} M : \alpha$ we obtain $[\![M]\!]^{\mathcal{R}}_{\rho} \in [\![\alpha]\!]^{\mathcal{R}}$. But $[\![x \odot M]\!]^{\mathcal{R}}_{\rho} = [\![M]\!]^{\mathcal{R}}_{\rho}$ by Lemma 1(iii), hence $[\![x \odot M]\!]^{\mathcal{R}}_{\rho} \in [\![\alpha]\!]^{\mathcal{R}}$.

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

$$\Gamma, x : \alpha, y : \beta \vdash_{\mathcal{R}} M : \sigma \Rightarrow \Gamma, z : \alpha \cap \beta \vdash_{\mathcal{R}} z <_y^x M : \sigma.$$

  By the IH $\Gamma, x : \alpha, y : \beta \models_{\mathcal{R}} M : \sigma$. Suppose that $\rho \models_{\mathcal{R}} \Gamma, z : \alpha \cap \beta$, in order to prove $[\![z <_y^x M]\!]^{\mathcal{R}}_{\rho} \in [\![\sigma]\!]^{\mathcal{R}}$. This means that $\rho \models_{\mathcal{R}} \Gamma$ and $\rho \models_{\mathcal{R}} z : \alpha \cap \beta \Leftrightarrow \rho(z) \in [\![\alpha]\!]^{\mathcal{R}}$ and $\rho(z) \in [\![\beta]\!]^{\mathcal{R}}$. For the sake of simplicity let

$\rho(z) \equiv N \in \Lambda^{\circ}_{\mathcal{R}}$. We define a new valuation ρ' such that $\rho' = \rho(N/x, N/y)$. Then $\rho' \models_{\mathcal{R}} \Gamma, x:\alpha, y:\beta$ since $x, y \notin Dom(\Gamma)$, $N \in [\![\alpha]\!]^{\mathcal{R}}$ and $N \in [\![\beta]\!]^{\mathcal{R}}$. By the IH $[\![M]\!]^{\mathcal{R}}_{\rho'} \in [\![\sigma]\!]^{\mathcal{R}}$. By the definition of term valuation (Definition 7) and Lemma 1(iv) we obtain $[\![M]\!]^{\mathcal{R}}_{\rho'} = [\![M]\!]^{\mathcal{R}}_{\rho(N/x,N/y)} = [\![z <^{x}_{y} M]\!]^{\mathcal{R}}_{\rho}$, since $\rho(z) = N$. Hence, $[\![z <^{x}_{y} M]\!]^{\mathcal{R}}_{\rho} \in [\![\sigma]\!]^{\mathcal{R}}$.

□

Lemma 2 *Let $[\![-]\!]^{\mathcal{R}}_{\rho} : \Lambda_{\mathcal{R}} \to \Lambda^{\circ}_{\mathcal{R}}$ be a term valuation. Then*

$$[\![M]\!]^{\mathcal{R}}_{\rho} \in \mathcal{SN} \;\Rightarrow\; M \in \mathcal{SN}.$$

Proof: The proof is straightforward since although the terms M and $[\![M]\!]^{\mathcal{R}}_{\rho}$ are different, the latter is an instantiation of the former $[\![M]\!]^{\mathcal{R}}_{\rho} = M[N/x, N/y, \rho(x_1)/x_1, \ldots, \rho(x_n)/x_n]$. This is a well-known property in λ-calculus and adding contraction and weakening still preserves this property. For this reason, the strong normalisation property is preserved. □

Theorem 1 (SN for $\lambda_{\mathcal{R}}\cap$) *If $\Gamma \vdash_{\mathcal{R}} M : \alpha$, then M is strongly normalizing, i.e. $M \in \mathcal{SN}$.*

Proof: Suppose $\Gamma \vdash_{\mathcal{R}} M : \alpha$. By Proposition 13 $\Gamma \models_{\mathcal{R}} M : \alpha$. According to Definition 8(iii), this means that for all ρ such that $\rho \models_{\mathcal{R}} \Gamma$ we have that $\rho \models_{\mathcal{R}} M : \alpha$. Suppose $\Gamma = \{x_1 : \alpha_1, \ldots x_k : \alpha_1 k\}$. Since for each α_i by Proposition 12(iv), $[\![\alpha_i]\!]^{\mathcal{R}}$ is saturated, this implies that the property $\mathsf{INH}([\![\alpha_i]\!]^{\mathcal{R}})$. Thus for each $[\![\alpha_i]\!]^{\mathcal{R}}$

$$(\exists n_i \geq 0)\ (\forall p \geq n_i)\ \lambda x_1 \ldots \lambda x_p.\lambda y.y \in [\![\alpha_i]\!]^{\mathcal{R}}.$$

Let us obtain a particular valuation ρ_0 so that for each $x_i \in Dom(\Gamma)$

$$\rho_0(x_i) = \lambda x_1 \ldots \lambda x_{n_i}.\lambda y.y$$

Since $\mathsf{INH}([\![\alpha_i]\!]^{\mathcal{R}})$ it follows that $\rho_0(x_i) \in [\![\alpha_i]\!]^{\mathcal{R}}$ for all $x_i : \alpha_i \in \Gamma$. Therefore, by Definition 8(ii) $\rho_0 \models_{\mathcal{R}} \Gamma$ and by Definition 8(iii) $\rho_0 \models_{\mathcal{R}} M : \alpha$. Now by Definition 8(i) we can conclude that $[\![M]\!]^{\mathcal{R}}_{\rho_0} \in [\![\alpha]\!]^{\mathcal{R}}$. By Proposition 12(iv) $[\![\alpha]\!]^{\mathcal{R}} \subseteq \mathcal{SN}^{\circ}_{\mathcal{R}}$, so by applying Lemma 2 we get $M \in \mathcal{SN}^{\circ}_{\mathcal{R}}$.

□

There are a few differences with respect to the traditional reducibility method presented in [Bar92]. First of all, the interpretation of types only contains closed $\lambda_{\mathcal{R}}$-terms, as opposed to all λ-terms. Next, instead of VAR condition which ensures that the saturated sets contain variables, we introduce the condition INH which ensures that the terms of the form $\lambda_1 \ldots \lambda_p.\lambda y.y$ belong to all $\mathcal{R}$-saturated sets. Further, valuations map variables to closed terms and term valuations map $\lambda_{\mathcal{R}}$-terms to closed $\lambda_{\mathcal{R}}$-terms.. Finally, in the proof of Theorem 1, instead of the valuation $\rho_0(x) = x$ which maps all the variables to themselves, we need a valuation $\rho_0(x) = \lambda x_1 \ldots \lambda x_n.\lambda y.y.$ that maps every variable to a closed term (which is provided by $\mathsf{INH}([\![\beta]\!]^{\mathcal{R}})$).

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

In this subsection, we prove the strong normalisation property of the $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi with intersection types. The termination is proved by showing that the reductions on the sets $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ of the typeable $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions are included in particular well-founded relations, which we define as the lexicographic products of several well-founded component relations. The first ones are based on the mappings of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions into $\lambda_{\mathcal{R}}$-terms. We show that these mappings preserve types and that all $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reductions can be simulated by the reductions or identities of the corresponding $\lambda_{\mathcal{R}}$-calculi. The other well-founded orders are based on the introduction of quantities designed to decrease a global measure associated with specific $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-expressions during the computation.

Definition 9 *The mappings* $\lfloor\ \rfloor^{\mathcal{R}} : \mathrm{T}^{\mathsf{Gtz}}_{\mathcal{R}} \to \Lambda_{\mathcal{R}}$ *are defined together with the auxiliary mappings*

$$\lfloor\ \rfloor^{\mathcal{R}}_k : \mathrm{K}^{\mathsf{Gtz}}_{\mathcal{R}} \to (\Lambda_{\mathcal{R}} \to \Lambda_{\mathcal{R}})$$

in the following way:

$$\begin{array}{lcllcl}
\lfloor x\rfloor^{\mathcal{R}} & = & x & \lfloor \widehat{x}.t\rfloor^{\mathcal{R}}_k(M) & = & (\lambda x.\lfloor t\rfloor^{\mathcal{R}})M \\
\lfloor \lambda x.t\rfloor^{\mathcal{R}} & = & \lambda x.\lfloor t\rfloor^{\mathcal{R}} & \lfloor t :: k\rfloor^{\mathcal{R}}_k(M) & = & \lfloor k\rfloor^{\mathcal{R}}_k(M\lfloor t\rfloor^{\mathcal{R}}) \\
\lfloor x \odot t\rfloor^{\mathcal{R}} & = & x \odot \lfloor t\rfloor^{\mathcal{R}} & \lfloor x \odot k\rfloor^{\mathcal{R}}_k(M) & = & \{x\} \setminus Fv(M) \odot \lfloor k\rfloor^{\mathcal{R}}_k(M) \\
\lfloor x <^y_z t\rfloor^{\mathcal{R}} & = & x <^y_z \lfloor t\rfloor^{\mathcal{R}} & \lfloor x <^y_z k\rfloor^{\mathcal{R}}_k(M) & = & x <^y_z \lfloor k\rfloor^{\mathcal{R}}_k(M) \\
\lfloor tk\rfloor^{\mathcal{R}} & = & \lfloor k\rfloor^{\mathcal{R}}_k(\lfloor t\rfloor^{\mathcal{R}}) & & &
\end{array}$$

Lemma 3

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

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

Proof: The proof directly yields from Definition 9 and the substitution definitions. □

We prove that the mappings $\lfloor\ \rfloor^{\mathcal{R}}$ and $\lfloor\ \rfloor^{\mathcal{R}}_k$ preserve types. In the sequel, we use $\vdash_{\lambda_{\mathcal{R}}}$ to denote derivations in $\lambda_{\mathcal{R}}\cap$. The notation $\Lambda_{\mathcal{R}(\Gamma\vdash_{\lambda_{\mathcal{R}}}\alpha)}$ stands for $\{M \mid M \in \Lambda_{\mathcal{R}} \ \&\ \Gamma \vdash_{\lambda_{\mathcal{R}}} M : \alpha\}$.

Proposition 14 (Type preservation with $\lfloor\ \rfloor^{\mathcal{R}}$)

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

(ii) If $\Gamma; \cap^n_j \tau_j \vdash_{\mathcal{R}} k : \sigma$, *then* $\lfloor k\rfloor^{\mathcal{R}}_k : \Lambda_{\mathcal{R}(\Gamma'_j\vdash_{\lambda_{\mathcal{R}}}\tau_j)} \to \Lambda_{\mathcal{R}(\Gamma\sqcup_{\mathsf{c}}\Gamma'\vdash_{\lambda_{\mathcal{R}}}\sigma)}$, *for all* $j \in \{1,\ldots,n\}$ *and for some* $\Gamma' = \Gamma'_1 \sqcup \ldots \sqcup \Gamma'_n$.

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

- Cases (Ax_{iw}), (Ax_{ew}), $(\to_R)$, $(Weak_t)$ and $(Cont_t)$ are easy, because the system $\lambda_{\mathcal{R}}\cap$ has exactly the same rules.
- Case (Sel): the derivation ends with the rule

$$\frac{\Gamma', x : \alpha \vdash_{\mathcal{R}} t : \sigma}{\Gamma'; \alpha \vdash_{\mathcal{R}} \widehat{x}.t : \sigma}\ (Sel)$$

By IH we have that $\Gamma', x : \alpha \vdash_{\lambda_{\mathcal{R}}} \lfloor t\rfloor^{\mathcal{R}} : \sigma$. For any $M \in \Lambda_{\mathcal{R}}$ such that $\Gamma'' \vdash_{\lambda_{\mathcal{R}}} M : \alpha$, for some Γ'', we have

$$\frac{\dfrac{\Gamma', x : \alpha \vdash_{\lambda_{\mathcal{R}}} \lfloor t\rfloor^{\mathcal{R}} : \sigma}{\Gamma' \vdash_{\lambda_{\mathcal{R}}} \lambda x.\lfloor t\rfloor^{\mathcal{R}} : \alpha \to \sigma}\ (\to_I) \qquad \Gamma'' \vdash_{\lambda_{\mathcal{R}}} M : \alpha}{\Gamma' \sqcup_{\mathsf{c}} \Gamma'' \vdash_{\lambda_{\mathcal{R}}} (\lambda x.\lfloor t\rfloor^{\mathcal{R}})M : \sigma}\ (\to_E)$$

Since $(\lambda x.\lfloor t\rfloor^{\mathcal{R}})M = \lfloor \widehat{x}.t\rfloor^{\mathcal{R}}_k(M)$, we conclude that $\lfloor \widehat{x}.t\rfloor^{\mathcal{R}}_k : \Lambda_{\mathcal{R}(\Gamma''\vdash_{\lambda_{\mathcal{R}}}\alpha)} \to \Lambda_{\mathcal{R}(\Gamma'\sqcup_{\mathsf{c}}\Gamma''\vdash_{\lambda_{\mathcal{R}}}\sigma)}$.

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

$$\frac{\Gamma_1 \vdash_{\mathcal{R}} t : \sigma_1 \; ... \; \Gamma_n \vdash_{\mathcal{R}} t : \sigma_n \quad \Delta; \cap_j^m \tau_j \vdash k : \rho}{\Gamma \sqcup_{\mathsf{c}} \Delta; \cap_j^m (\cap_i^n \sigma_i \to \tau_j) \vdash_{\mathcal{R}} t :: k : \rho} \; (\to_L)$$

where $\Gamma = \Gamma_1 \sqcup ... \sqcup \Gamma_n$. By IH we have that $\Gamma_i \vdash_{\lambda_{\mathcal{R}}} \lfloor t \rfloor^{\mathcal{R}} : \sigma_i$, for $i = 1...n$. For any $M \in \Lambda_{\mathcal{R}}$ such that $\Gamma'_j \vdash_{\lambda_{\mathcal{R}}} M : \cap_i^n \sigma_i \to \tau_j$, $j = 1, \ldots, m$ we have

$$\frac{\Gamma'_j \vdash_{\lambda_{\mathcal{R}}} M : \cap_i^n \sigma_i \to \tau_j \quad \Gamma \vdash_{\lambda_{\mathcal{R}}} \lfloor t \rfloor^{\mathcal{R}} : \sigma_i}{\Gamma \sqcup_{\mathsf{c}} \Gamma'_j \vdash_{\lambda_{\mathcal{R}}} M \lfloor t \rfloor^{\mathcal{R}} : \tau_j} \; (\to_E)$$

From the right-hand side premise in the $(\to_L)$ rule, by IH, we get that $\lfloor k \rfloor_k^{\mathcal{R}}$ is the function with the scope $\lfloor k \rfloor_k^{\mathcal{R}} : \Lambda_{\mathcal{R}(\Gamma'''_j \vdash_{\lambda_{\mathcal{R}}} \tau_j)} \to \Lambda_{\mathcal{R}(\Gamma''' \sqcup_{\mathsf{c}} \Gamma'' \vdash_{\lambda_{\mathcal{R}}} \rho)}$, for $\Gamma''' = \Gamma'''_1 \sqcup ... \sqcup \Gamma'''_n$. For $\Gamma''' \equiv \Gamma \sqcup_{\mathsf{c}} \Gamma'$ and by taking $M \lfloor t \rfloor^{\mathcal{R}}$ as the argument of the function $\lfloor k \rfloor_k^{\mathcal{R}}$, we get $\Gamma \sqcup_{\mathsf{c}} \Delta \sqcup_{\mathsf{c}} \Gamma' \vdash_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(M \lfloor t \rfloor^{\mathcal{R}}) : \rho$. Since $\lfloor k \rfloor_k^{\mathcal{R}}(M \lfloor t \rfloor^{\mathcal{R}}) = \lfloor t :: k \rfloor_k^{\mathcal{R}}(M)$, we have that $\Gamma \sqcup_{\mathsf{c}} \Delta \sqcup_{\mathsf{c}} \Gamma' \vdash_{\lambda_{\mathcal{R}}} \lfloor t :: k \rfloor_k^{\mathcal{R}}(M) : \rho$. This holds for any M of the appropriate type, yielding $\lfloor t :: k \rfloor_k^{\mathcal{R}} : \Lambda_{\mathcal{R}(\Gamma' \vdash_{\lambda_{\mathcal{R}}} \cap_i^n \sigma_i \to \tau_j)} \to \Lambda_{\mathcal{R}(\Gamma \sqcup_{\mathsf{c}} \Delta \sqcup_{\mathsf{c}} \Gamma' \vdash_{\lambda_{\mathcal{R}}} \rho)}$, which is exactly what we need.

- Case (Cut): the derivation ends with the rule

$$\frac{\Gamma_1 \vdash_{\mathcal{R}} t : \tau_1 \ldots \Gamma_n \vdash_{\mathcal{R}} t : \tau_n \quad \Delta; \cap \tau_i^n \vdash_{\mathcal{R}} k : \sigma}{(\Gamma \sqcup \ldots \sqcup \Gamma_n) \sqcup_{\mathsf{c}} \Delta \vdash_{\mathcal{R}} tk : \sigma} \; (Cut)$$

By IH we have that $\Gamma \vdash_{\lambda_{\mathcal{R}}} \lfloor t \rfloor^{\mathcal{R}} : \tau_i$ and $\lfloor k \rfloor_k^{\mathcal{R}} : \Lambda_{\mathcal{R}(\Gamma' \vdash_{\lambda_{\mathcal{R}}} \tau_i)} \to \Lambda_{\mathcal{R}(\Gamma', \Delta \vdash_{\lambda_{\mathcal{R}}} \sigma)}$ for all $i = 1, \ldots, n$. Hence, for any $M \in \Lambda^{\lambda_{\mathcal{R}}}$ such that $\Gamma' \vdash_{\lambda_{\mathcal{R}}} M : \tau_i$, $\Gamma' \sqcup_{\mathsf{c}} \Delta \vdash_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(M) : \sigma$ holds. By taking $M \equiv \lfloor t \rfloor^{\mathcal{R}}$ and $\Gamma' \equiv \Gamma$, we get $\Gamma, \Delta \vdash_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(\lfloor t \rfloor^{\mathcal{R}}) : \sigma$. But $\lfloor k \rfloor_k^{\mathcal{R}}(\lfloor t \rfloor^{\mathcal{R}}) = \lfloor tk \rfloor^{\mathcal{R}}$, so the proof is done.

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

$$\frac{\Gamma; \beta \vdash_{\mathcal{R}} k : \sigma}{\Gamma, x : \alpha; \beta \vdash_{\mathcal{R}} x \odot k : \sigma} \; (Weak_k)$$

By IH we have that $\lfloor k \rfloor_k^{\mathcal{R}}$ is the function with the scope $\lfloor k \rfloor_k^{\mathcal{R}} : \Lambda_{\mathcal{R}(\Gamma' \vdash_{\lambda_{\mathcal{R}}} \beta)} \to \Lambda_{\mathcal{R}(\Gamma \sqcup_{\mathsf{c}} \Gamma' \vdash_{\lambda_{\mathcal{R}}} \sigma)}$, meaning that for each $M \in \Lambda^{\lambda_{\mathcal{R}}}$ such that $\Gamma' \vdash_{\lambda_{\mathcal{R}}} M : \beta$ holds $\Gamma' \sqcup_{\mathsf{c}} \Gamma'' \vdash_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(M) : \sigma$. Now, we can apply $(Weak)$ rule:

$$\frac{\Gamma \sqcup_{\mathsf{c}} \Gamma' \vdash_{\mathcal{R}} \lfloor k \rfloor_k^{\mathcal{R}}(M) : \sigma}{\Gamma \sqcup_{\mathsf{c}} \Gamma', x : \alpha \vdash_{\mathcal{R}} x \odot \lfloor k \rfloor_k^{\mathcal{R}}(M) : \sigma} \; (Weak)$$

Since $x \odot \lfloor k \rfloor_k^{\mathcal{R}}(M) = \lfloor x \odot k \rfloor_k^{\mathcal{R}}(M)$, this means that $\lfloor x \odot k \rfloor_k^{\mathcal{R}} : \Lambda_{\mathcal{R}(\Gamma' \vdash_{\lambda_{\mathcal{R}}} \beta)} \to \Lambda_{\mathcal{R}(\Gamma \sqcup_{\mathsf{c}} \Gamma', x : \alpha \vdash_{\lambda_{\mathcal{R}}} \sigma)}$, which is exactly what we wanted to get.

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

For the given encoding $\lfloor \ \rfloor^{\mathcal{R}}$, we show that each $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-reduction step can be simulated by $\lambda_{\mathcal{R}}$-reduction or identity. In order to do so, we first prove the following lemmas by induction on the structure of the contexts. The proofs of Lemma 5 and Lemma 6 also use Regnier's σ reductions, investigated in [Reg94].

Lemma 4 *If* $M \to_{\lambda_{\mathcal{R}}} M'$, *then* $\lfloor k \rfloor_k^{\mathcal{R}}(M) \to_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(M')$.

Lemma 5 $\lfloor k \rfloor_k^{\mathcal{R}}((\lambda x.P)N) \to_{\lambda_{\mathcal{R}}} (\lambda x. \lfloor k \rfloor_k^{\mathcal{R}}(P))N$.

Lemma 6 *If $M \in \Lambda_{\mathcal{R}}$ and $k, k' \in \mathrm{K}_{\mathcal{R}}^{\mathsf{Gtz}}$, then $\lfloor k' \rfloor_k^{\mathcal{R}} \circ \lfloor k \rfloor_k^{\mathcal{R}}(M) \to_{\lambda_{\mathcal{R}}} \lfloor k@k' \rfloor_k^{\mathcal{R}}(M)$.*

Lemma 7

(i) *If $x \notin Fv(k)$, then $(\lfloor k \rfloor_k^{\mathcal{R}}(M))[N/x] = \lfloor k \rfloor_k^{\mathcal{R}}(M[N/x])$.*

(ii) *If $x, y \notin Fv(k)$, then $z <_y^x (\lfloor k \rfloor_k^{\mathcal{R}}(M)) \to_{\lambda_{\mathcal{R}}} \lfloor k \rfloor_k^{\mathcal{R}}(z <_y^x M)$.*

(iii) $\lfloor k \rfloor_k^{\mathcal{R}}(x \odot M) \to_{\lambda_{\mathcal{R}}} \{x\} \setminus Fv(k) \odot \lfloor k \rfloor_k^{\mathcal{R}}(M)$.

Now we can prove that the reduction rules of $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi can be simulated by the reduction rules or identities in the corresponding $\lambda_{\mathcal{R}}$-calculi. Moreover, the equivalences of $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi are preserved in $\lambda_{\mathcal{R}}$-calculi.

Theorem 2 (Simulation of $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-reductions by $\lambda_{\mathcal{R}}$-reductions)

(i) *If a term $M \to M'$, then $\lfloor M \rfloor^{\mathcal{R}} \to_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$.*

(ii) *If a context $k \to k'$ by γ_6 or ω_6 reduction, then $\lfloor k \rfloor_k^{\mathcal{R}}(M) \equiv \lfloor k' \rfloor_k^{\mathcal{R}}(M)$, for any $M \in \Lambda_{\mathcal{R}}$.*

(iii) *If a context $k \to k'$ by some other reduction, then $\lfloor k \rfloor_k^{\mathcal{R}}(M) \to_{\lambda_{\mathcal{R}}} \lfloor k' \rfloor_k^{\mathcal{R}}(M)$, for any $M \in \Lambda_{\mathcal{R}}$.*

(iv) *If $M \equiv M'$, then $\lfloor M \rfloor^{\mathcal{R}} \equiv_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$, and if $k \equiv k'$, then $\lfloor k \rfloor_k^{\mathcal{R}}(M) \equiv_{\lambda_{\mathcal{R}}} \lfloor k' \rfloor_k^{\mathcal{R}}(M)$, for any $M \in \Lambda_{\mathcal{R}}$.*

Proof: Without losing generality, we prove the statement only for the outermost reductions. The rules $\gamma_0, \gamma_0', \gamma_1, \gamma_2, \gamma_3$, $\omega_1, \omega_2, \omega_3, \gamma\omega_1, \gamma\omega_2$ are simulated by the corresponding rules of $\lambda_{\mathcal{R}}$. For the remaining rules we have:

(β) $(\lambda x.t)(u :: k) \to u(\widehat{x}.tk)$.

On the one hand $\lfloor M \rfloor^{\mathcal{R}} = \lfloor (\lambda x.t)(u :: k) \rfloor^{\mathcal{R}} = \lfloor u :: k \rfloor_k^{\mathcal{R}}(\lfloor \lambda x.t \rfloor^{\mathcal{R}}) = \lfloor k \rfloor_k^{\mathcal{R}}((\lambda x.\lfloor t \rfloor^{\mathcal{R}})\lfloor u \rfloor^{\mathcal{R}})$

On the other hand, $\lfloor M' \rfloor^{\mathcal{R}} = \lfloor u(\widehat{x}.tk) \rfloor^{\mathcal{R}} = \lfloor \widehat{x}.tk \rfloor_k^{\mathcal{R}}(\lfloor u \rfloor^{\mathcal{R}}) = (\lambda x.\lfloor tk \rfloor^{\mathcal{R}})\lfloor u \rfloor^{\mathcal{R}} =$
$(\lambda x.\lfloor k \rfloor_k^{\mathcal{R}}(\lfloor t \rfloor^{\mathcal{R}}))\lfloor u \rfloor^{\mathcal{R}}$. So, $\lfloor M \rfloor^{\mathcal{R}} \to_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$ by Lemma 5.

(σ) $T(\widehat{x}.v) \to v[T/x]$.

$\lfloor M \rfloor^{\mathcal{R}} = \lfloor T(\widehat{x}.x) \rfloor^{\mathcal{R}} = \lfloor \widehat{x}.x \rfloor_k^{\mathcal{R}}(\lfloor T \rfloor^{\mathcal{R}}) = (\lambda x.\lfloor v \rfloor^{\mathcal{R}})(\lfloor T \rfloor^{\mathcal{R}})$,
$\lfloor M' \rfloor^{\mathcal{R}} = \lfloor v[T/x] \rfloor^{\mathcal{R}} = \lfloor v \rfloor^{\mathcal{R}}[\lfloor T \rfloor^{\mathcal{R}}/x]$ by Lemma 3, so $\lfloor M \rfloor^{\mathcal{R}} \to_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$ by β-reduction in the $\lambda_{\mathcal{R}}$-calculus.

(π) $(tk)k' \to t(k@k')$

$\lfloor M \rfloor^{\mathcal{R}} = \lfloor (tk)k' \rfloor^{\mathcal{R}} = \lfloor k' \rfloor_k^{\mathcal{R}}(\lfloor tk \rfloor^{\mathcal{R}}) = \lfloor k' \rfloor_k^{\mathcal{R}}(\lfloor k \rfloor_k^{\mathcal{R}}(\lfloor t \rfloor^{\mathcal{R}}))$
$\lfloor M' \rfloor^{\mathcal{R}} = \lfloor t(k@k') \rfloor^{\mathcal{R}} = \lfloor k@k' \rfloor_k^{\mathcal{R}}(\lfloor t \rfloor^{\mathcal{R}})$.
Applying Lemma 6 we get that $\lfloor M \rfloor^{\mathcal{R}} \to_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$.

(μ) $\widehat{x}.xk \to k$.

This reduction reduces context to context, so we have:
$\lfloor \widehat{x}.xk \rfloor_k^{\mathcal{R}}(M) = (\lambda x.\lfloor xk \rfloor^{\mathcal{R}})(M) = (\lambda x.\lfloor k \rfloor_k^{\mathcal{R}}(x))(M)$.

This reduces to $\lfloor k \rfloor_k^{\mathcal{R}}(M)$ by β-reduction in the $\lambda_{\mathcal{R}}$-calculus.

(γ_4) $x <_{x_2}^{x_1} (\widehat{y}.t) \to \widehat{y}.(x <_{x_2}^{x_1} t)$

$\lfloor K \rfloor_k^{\mathcal{R}}(M) = x <_{x_2}^{x_1} (\lambda y.\lfloor t \rfloor^{\mathcal{R}})M$.
On the other hand,
$\lfloor K' \rfloor_k^{\mathcal{R}}(M) = (\lambda y.x <_{x_2}^{x_1} \lfloor t \rfloor^{\mathcal{R}})M$.
So $\lfloor M \rfloor^{\mathcal{R}} \to_{\lambda_{\mathcal{R}}} \lfloor M' \rfloor^{\mathcal{R}}$ by the rule γ_2.

(γ_5) $x <^{x_1}_{x_2} (t :: k) \to (x <^{x_1}_{x_2} t) :: k$, if $x_1, x_2 \notin Fv(k)$

$\lfloor K \rfloor^{\mathcal{R}}_k(M) = x <^{x_1}_{x_2} (\lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}}))$.

$\lfloor K' \rfloor^{\mathcal{R}}_k(M) = \lfloor k \rfloor^{\mathcal{R}}_k(M(x <^{x_1}_{x_2} \lfloor t \rfloor^{\mathcal{R}}))$.

$x_1, x_2 \notin Fv(k)$ implies $x_1, x_2 \notin Fv(\lfloor k \rfloor^{\mathcal{R}}_k(M))$ so we can apply Lemma 7 followed by reduction γ_3 and conclude that $\lfloor K \rfloor^{\mathcal{R}}_k(M) \to_{\lambda_{\mathcal{R}}} \lfloor K' \rfloor^{\mathcal{R}}_k(M)$.

(γ_6) $x <^{x_1}_{x_2} (t :: k) \to t :: (x <^{x_1}_{x_2} k)$, if $x_1, x_2 \notin Fv(t)$

$\lfloor K \rfloor^{\mathcal{R}}_k(M) = \lfloor x <^{x_1}_{x_2} (t :: k) \rfloor^{\mathcal{R}}_k(M) = x <^{x_1}_{x_2} \lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}})$.

On the other hand,
$\lfloor K' \rfloor^{\mathcal{R}}_k(M) = \lfloor t :: (x <^{x_1}_{x_2} k) \rfloor^{\mathcal{R}}_k(M) = x <^{x_1}_{x_2} \lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}})$.

So $\lfloor K \rfloor^{\mathcal{R}}_k(M) = \lfloor K' \rfloor^{\mathcal{R}}_k(M)$.

(ω_4) $\widehat{x}.(y \odot t) \to y \odot (\widehat{x}.t)$, $x \neq y$

$\lfloor K \rfloor^{\mathcal{R}}_k(M) = \lfloor \widehat{x}.(y \odot t) \rfloor^{\mathcal{R}}_k(M) = (\lambda x.y \odot \lfloor t \rfloor^{\mathcal{R}})M$.

$\lfloor K' \rfloor^{\mathcal{R}}_k(M) = \lfloor y \odot (\widehat{x}.t) \rfloor^{\mathcal{R}}_k(M) = y \odot (\lambda x. \lfloor t \rfloor^{\mathcal{R}})M$.

So $\lfloor K \rfloor^{\mathcal{R}}_k(M) \to_{\lambda_{\mathcal{R}}} \lfloor K' \rfloor^{\mathcal{R}}_k(M)$ by the rule ω_2.

(ω_5) $(x \odot t) :: k \to \{x\} \setminus Fv(k) \odot (t :: k)$

$\lfloor K \rfloor^{\mathcal{R}}_k(M) = \lfloor (x \odot t) :: k \rfloor^{\mathcal{R}}_k(M) = \lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor x \odot t \rfloor^{\mathcal{R}}) = \lfloor k \rfloor^{\mathcal{R}}_k(Mx \odot \lfloor t \rfloor^{\mathcal{R}})$.

$\lfloor K' \rfloor^{\mathcal{R}}_k(M) = \lfloor \{x\} \setminus Fv(k) \odot (t :: k) \rfloor^{\mathcal{R}}_k(M) = (\{x\} \setminus Fv(k)) \setminus Fv(M) \odot \lfloor t :: k \rfloor^{\mathcal{R}}_k(M) = (\{x\} \setminus Fv(k)) \setminus Fv(M) \odot \lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}})$.

Applying the rule ω_3 of $\lambda_{\mathcal{R}}$ and Lemma 7 we get that

$\lfloor K \rfloor^{\mathcal{R}}_k(M) \to_{\lambda_{\mathcal{R}}} \lfloor K' \rfloor^{\mathcal{R}}_k(M)$.

(ω_6) $t :: (x \odot k) \to \{x\} \setminus Fv(t) \odot (t :: k)$

$\lfloor K \rfloor^{\mathcal{R}}_k(M) = \lfloor t :: (x \odot t) \rfloor^{\mathcal{R}}_k(M) = \lfloor x \odot k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}}) = \{x\} \setminus Fv(M \lfloor t \rfloor^{\mathcal{R}}) \odot \lfloor k \rfloor^{\mathcal{R}}_k(M \lfloor t \rfloor^{\mathcal{R}}) = \lfloor K' \rfloor^{\mathcal{R}}_k(M)$.

The proof of (iv) is trivial, since the equivalences of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-calculi and $\lambda_{\mathcal{R}}$-calculi coincide. □

The previous proposition shows that β, π, σ, μ, γ_0 - γ_5, ω_1 - ω_5, $\gamma\omega_1$ and $\gamma\omega_2$ $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reductions are interpreted by $\lambda_{\mathcal{R}}$-reductions and that γ_6 and ω_6 $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reductions are interpreted by an identity in the $\lambda_{\mathcal{R}}$. Since the set of equivalences of the two bases of the resource control cube coincide, they are trivially preserved. If one wants to prove that there is no infinite sequence of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reductions one has to prove that there cannot exist an infinite sequence of $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reductions which are all interpreted as identities. To prove this, one shows that if a term is reduced with such a $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-reduction, it is reduced for another order that forbids infinite decreasing chains. This order is itself composed of several orders, free of infinite decreasing chains (Definition 11).

Definition 10 *The functions $\mathcal{S}^{\mathcal{R}}$, $\| \|^{\mathcal{R}}_{\mathsf{c}}$, $\| \|^{\mathcal{R}}_{\mathsf{w}} : \Lambda^{\mathsf{Gtz}}_{\mathcal{R}} \to \mathbb{N}$ are defined in Figure 18.*

Lemma 8 *For all $e, e' \in \lambda^{\mathsf{Gtz}}_{\mathcal{R}}$:*

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

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

(iii) If $e \equiv e'$, then $\|e\|^{\mathcal{R}}_{\mathsf{c}} = \|e'\|^{\mathcal{R}}_{\mathsf{c}}$.

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

Figure 18: Definitions of $\mathcal{S}^{\mathcal{R}}(e), \|e\|^{\mathcal{R}}_{\mathsf{c}}, \|e\|^{\mathcal{R}}_{\mathsf{w}}$

Lemma 9 *For all $e, e' \in \lambda^{\mathsf{Gtz}}_{\mathcal{R}}$:*

(i) If $e \rightarrow_{\omega_6} e'$, then $\|e\|^{\mathcal{R}}_{\mathsf{w}} > \|e'\|^{\mathcal{R}}_{\mathsf{w}}$.

(ii) If $e \equiv e'$, then $\|e\|^{\mathcal{R}}_{\mathsf{w}} = \|e'\|^{\mathcal{R}}_{\mathsf{w}}$.

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

Definition 11 *We define the following strict orders and equivalences on $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$:*

(i) $t >_{\lambda_{\mathcal{R}}} t'$ *iff* $\lfloor t \rfloor^{\mathcal{R}} \rightarrow^{+}_{\lambda_{\mathcal{R}}} \lfloor t' \rfloor^{\mathcal{R}}$; $t =_{\lambda_{\mathcal{R}}} t'$ *iff* $\lfloor t \rfloor^{\mathcal{R}} \equiv \lfloor t' \rfloor^{\mathcal{R}}$;
$k >_{\lambda_{\mathcal{R}}} k'$ *iff* $\lfloor k \rfloor^{\mathcal{R}}_{k}(M) \rightarrow^{+}_{\lambda_{\mathcal{R}}} \lfloor k' \rfloor^{\mathcal{R}}(M)$ *for every* $M \in \Lambda_{\mathcal{R}}$;
$k =_{\lambda_{\mathcal{R}}} k'$ *iff* $\lfloor k \rfloor^{\mathcal{R}}_{k}(M) \equiv \lfloor k' \rfloor^{\mathcal{R}}_{k}(M)$ *for every* $M \in \Lambda_{\mathcal{R}}$;

(ii) $e >_{\mathsf{c}} e'$ *iff* $\|e\|^{\mathcal{R}}_{\mathsf{c}} > \|e'\|^{\mathcal{R}}_{\mathsf{c}}$; $e =_{\mathsf{c}} e'$ *iff* $\|e\|^{\mathcal{R}}_{\mathsf{c}} = \|e'\|^{\mathcal{R}}_{\mathsf{c}}$;

(iii) $e >_{\mathsf{w}} e'$ *iff* $\|e\|^{\mathcal{R}}_{\mathsf{w}} > \|e'\|^{\mathcal{R}}_{\mathsf{w}}$; $e =_{\mathsf{w}} e'$ *iff* $\|e\|^{\mathcal{R}}_{\mathsf{w}} = \|e'\|^{\mathcal{R}}_{\mathsf{w}}$;

A lexicographic product of two orders $>_1$ and $>_2$ is usually defined as follows ([BN98]):
$a >_1 \times_{lex} >_2 b \Leftrightarrow a >_1 b$ *or* $(a =_1 b$ *and* $a >_2 b)$.

Definition 12 *We define the relations $\gg^{\mathcal{R}}$ on $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}$ as the lexicographic products:*

$$\gg^{\mathcal{R}} \;=\; >_{\lambda_{\mathcal{R}}} \times_{lex} >_{\mathsf{c}} \times_{lex} >_{\mathsf{w}} .$$

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

Proposition 15 *For each $e \in \Lambda^{\mathsf{Gtz}}_{\mathcal{R}}$: if $e \rightarrow e'$, then $e \gg^{\mathcal{R}} e'$.*

Proof: The proof is by case analysis on the kind of reduction and the structure of $\gg^{\mathcal{R}}$.
If $e \rightarrow e'$ by β, σ, π, μ, γ_0, $\gamma_0{}'$, γ_1 γ_2, γ_3, γ_4 γ_5, ω_1, ω_2, ω_3 ω_4 or ω_5, $\gamma\omega_1$, $\gamma\omega_2$, reduction, then $e >_{\lambda_{\mathcal{R}}} e'$ by Proposition 2.
If $e \rightarrow e'$ by γ_6, then $e =_{\lambda_{\mathcal{R}}} e'$ by Proposition 2, and $e >_{\mathsf{c}} e'$ by Lemma 8.
Finally, if $e \rightarrow e'$ by ω_6, then $e =_{\lambda_{\mathcal{R}}} e'$ by Proposition 2, $e =_{\mathsf{c}} e'$ by Lemma 8 and $e >_{\mathsf{w}} e'$ by Lemma 9. □

SN 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 3 (Strong normalization) *Each expression in $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$ is SN.*

Proof: The reduction $\to$ is well-founded on $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$ as it is included (Proposition 15) in the relation $\gg^{\mathcal{R}}$ which is well-founded as the lexicographic product of the well-founded relations $>_{\lambda_{\mathcal{R}}}$, $>_{\mathsf{c}}$ and $>_{\mathsf{w}}$. The relation $>_{\lambda_{\mathcal{R}}}$ is based on the interpretation $\lfloor\ \rfloor^{\mathcal{R}} : \Lambda^{\mathsf{Gtz}}_{\mathcal{R}} \to \Lambda_{\mathcal{R}}$. By Proposition 14 typeability is preserved by the interpretation $\lfloor\ \rfloor^{\mathcal{R}}$ and $\to_{\lambda_{\mathcal{R}}}$ is SN (i.e., well-founded) on $\Lambda_{\mathcal{R}}\cap$ (Section 4.1), hence $>_{\lambda_{\mathcal{R}}}$ is well-founded on $\Lambda^{\mathsf{Gtz}}_{\mathcal{R}}\cap$. Similarly, $>_{\mathsf{c}}$ and $>_{\mathsf{w}}$ are well-founded, as they are based on interpretations into the well-founded relation $>$ on the set $\mathbb{N}$ of natural numbers. □

5 SN ⇒ Typeability in all systems of the resource control cube

Let us turn our attention to the most unique property of intersection types systems that all strongly normalising terms are typeable by intersection types. We will prove this property first for $\lambda_{\mathcal{R}}$-terms and then for $\lambda^{\mathsf{Gtz}}_{\mathcal{R}}$-terms.

5.1 SN ⇒ Typeability in $\lambda_{\mathcal{R}}\cap$

We want to prove that if a $\lambda_{\mathcal{R}}$-term is SN, then it is typeable in the system $\lambda_{\mathcal{R}}\cap$. We proceed in two steps:

1. we show that all $\lambda_{\mathcal{R}}$-normal forms are typeable and
2. we prove the head subject expansion property.

First, let us observe the structure of the $\lambda_{\mathcal{R}}$-normal forms, given by the following abstract syntax:

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

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

Proposition 16 *$\lambda_{\mathcal{R}}$-normal forms are typeable in the system $\lambda_{\mathcal{R}}\cap$.*

Proof: By an easy induction on the structure of M_{nf} and W_{nf}. Notice that the typing rules for introducing the explicit resource control operators change only the left-hand side of the sequents while the types of the expressions on the right-hand side stay unchanged. □

Proposition 17 (Inverse substitution lemma) *Let $\Gamma \vdash_{\mathcal{R}} M[N/x] : \sigma$ and N typeable. Then, there are Γ', $\Delta = \Delta_1 \sqcup ... \sqcup \Delta_n$ and τ_i, $i = 1,...,n$ such that $\Gamma = \Gamma' \sqcup_{\mathsf{c}} \Delta$, $\Delta_i \vdash_{\mathcal{R}} N : \tau_i$ for all $i = 1,\ldots,n$ and $\Gamma', x : \cap^n_i \tau_i \vdash_{\mathcal{R}} M : \sigma$.*

Proof: By induction on the structure of M. We will just show the basic case and the cases related to resource operators.

- Basic case:
  - $M \equiv x$. Then $M[N/x] = x[N/x] = N$. For $\Gamma' \equiv \emptyset$, $\Delta \equiv \Gamma$ and $\tau_i \equiv \sigma$ we get $\Delta \vdash_{\mathcal{R}} N : \sigma$ from the premise and $x : \cap^n_i \sigma_i \vdash_{\mathcal{R}} x : \sigma$ from the axiom.
  - $M \equiv y$. Then $M[N/x] = y[N/x] = y$. This case is possible only if $\mathsf{w} \notin \mathcal{R}$, so we can assume that $\Gamma = \Gamma \sqcup \Delta$, where $\Delta \vdash_{\mathcal{R}} N : \tau$, from the premise that N is typeable, and $\Gamma \vdash_{\mathcal{R}} y : \sigma$ from the implicit weakening axiom.
- Case $M \equiv x \odot M'$. Then $M[N/x] = (x \odot M')[N/x] = Fv(N) \setminus Fv(M') \odot M'$. From the premise $\Gamma \vdash_{\mathcal{R}} (x \odot M')[N/x] : \sigma$ we have $\Gamma \vdash_{\mathcal{R}} Fv(N) \setminus Fv(M') \odot M' : \sigma$, hence by the generation lemma $\Gamma' \vdash_{\mathcal{R}} M' : \sigma$, for $\Gamma' = \Gamma \setminus (Fv(N) \setminus Fv(M'))$. Now, for an arbitrary type α, we have $\Gamma', x : \alpha \vdash_{\mathcal{R}} x \odot M' : \sigma$. From the premise that N is typeable, knowing that $\mathsf{w} \in \mathcal{R}$, we get $\Delta \vdash_{\mathcal{R}} N : \beta$ where $Dom(\Delta) = Fv(N)$. The proposition is proved by taking $\alpha \equiv \beta$.

- Case $M \equiv y \odot M'$. Then $M[N/x] = (y \odot M')[N/x] = \{y\} \setminus Fv(N) \odot M'[N/x]$. From $\Gamma \vdash_{\mathcal{R}} \{y\} \setminus Fv(N) \odot M'[N/x] : \sigma$ by generation lemma, we have that $\Gamma' = \Gamma \setminus (\{y\} \setminus Fv(N)), y : \alpha$ and $\Gamma' \vdash_{\mathcal{R}} M'[N/x] : \sigma$. Now, by IH we get that $\Gamma' = \Gamma'' \sqcup_{\mathsf{c}} \Delta$, $\Delta = \Delta_1 \sqcup ... \sqcup \Delta_n$, $\Delta_i \vdash_{\mathcal{R}} N : \tau_i$ for all $i = 1, \ldots, n$ and $\Gamma'', x : \cap_i^n \tau_i \vdash_{\mathcal{R}} M' : \sigma$. Since $y \notin M'$, we get $\Gamma'', y : \alpha, x : \cap_i^n \tau_i \vdash_{\mathcal{R}} y \odot M' : \sigma$.

- Case $M \equiv y <_{y_2}^{y_1} M'$ and $x \neq y$. Then $M[N/x] = (y <_{y_2}^{y_1} M')[N/x] = y <_{y_2}^{y_1} M'[N/x]$. From the premise $\Gamma \vdash_{\mathcal{R}} y <_{y_2}^{y_1} M'[N/x] : \sigma$ using the generation lemma we get that $\Gamma = \Gamma', y : \alpha \cap \beta$ and $\Gamma', y_1 : \alpha, y_2 : \beta \vdash_{\mathcal{R}} M'[N/x] : \sigma$. By IH we get that $\Gamma' = \Gamma'' \sqcup_{\mathsf{c}} \Delta$, $\Delta = \Delta_1 \sqcup ... \sqcup \Delta_n$, $\Delta_i \vdash_{\mathcal{R}} N : \tau_i$ for all $i = 1, \ldots, n$ and $\Gamma'', y_1 : \alpha, y_2 : \beta, x : \cap_i^n \tau_i \vdash_{\mathcal{R}} M' : \sigma$. Using $(Cont)$ rule we get $\Gamma'', y : \alpha \cap \beta, x : \cap_i^n \tau_i \vdash_{\mathcal{R}} y <_{y_2}^{y_1} M' : \sigma$.

- Case $M \equiv x <_{x_2}^{x_1} M'$. Then $M[N/x] = (x <_{x_2}^{x_1} M')[N/x] = Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M'[N_1/x_1, N_2/x_2]$. From the premise $\Gamma \vdash_{\mathcal{R}} Fv(N) <_{Fv(N_2)}^{Fv(N_1)} M'[N_1/x_1, N_2/x_2] : \sigma$, using the generation lemma we get that for $Fv(N) = \{y_1, ..., y_n\}$ holds $\Gamma = \Gamma', y_1 : \alpha_1 \cap \beta_1, ..., y_n : \alpha_n \cap \beta_n$ and $\Gamma', y_1' : \alpha_1, y_1'' : \beta_1, ..., y_n' : \alpha_n, y_n'' : \beta_n \vdash_{\mathcal{R}} M[N_1/x_1, N_2/x_2] : \sigma$. Applying IH two times, we obtain $\Gamma' = \Gamma'' \sqcup_{\mathsf{c}} \Delta' \sqcup_{\mathsf{c}} \Delta''$, where $\Delta' = \Delta_1' \sqcup ... \sqcup \Delta_n'$ and $\Delta'' = \Delta_1'' \sqcup ... \sqcup \Delta_n''$, $\Delta_i' \vdash_{\mathcal{R}} N_1 : \tau_i'$ for all $i = 1, \ldots, n$, $\Delta_i'' \vdash_{\mathcal{R}} N_2 : \tau_i''$ for all $i = 1, \ldots, n$, and $\Gamma'', y_1' : \alpha_1, y_1'' : \beta_1, ..., y_n' : \alpha_n, y_n'' : \beta_n, x_1 : \cap_i^n \tau_i', x_2 : \cap_i^n \tau_i'' \vdash_{\mathcal{R}} M' : \sigma$. Since N_1 and N_2 are obtained by renaming N we have that $\cap_i^n \tau_i' \equiv \cap_i^n \tau_i'' \equiv \cap_i^n \tau_i$, $\Delta_i \vdash_{\mathcal{R}} N : \tau_i$ for all $i = 1, \ldots, n$ and for $\Delta_i = \Delta_i' \sqcup_{\mathsf{c}} \Delta_i''$. Finally, by $(Cont)$ rule we get $\Gamma'', y_1' : \alpha_1, y_1'' : \beta_1, ..., y_n' : \alpha_n, y_n'' : \beta_n, x : \cap_i^n \tau_i \vdash_{\mathcal{R}} x <_{x_2}^{x_1} M' : \sigma$ and the proof is done. □

Proposition 18 (Head subject expansion) *For every $\lambda_{\mathcal{R}}$-term M: if $M \to M'$, M is contracted redex and $\Gamma \vdash_{\mathcal{R}} M' : \sigma$, then $\Gamma \vdash_{\mathcal{R}} M : \sigma$, provided that if $M \equiv (\lambda x.N)P \to_\beta N[P/x] \equiv M'$, P is typeable.*

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

Theorem 4 (SN ⇒ typeability) *All strongly normalising $\lambda_{\mathcal{R}}$-terms are typeable in the $\lambda_{\mathcal{R}}\cap$ system.*

Proof: 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. We also use Proposition 3.

- If M is a normal form, then M is typeable by Proposition 16.

- 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 18. Notice that, if $M \equiv (\lambda x.N)P \to_\beta N[P/x] \equiv M'$, then, by IH, P is typeable - since the length of the longest reduction path out of P is not larger 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: $\lambda x.N$, $\lambda x.x \odot N$, $xN_1...N_m$, $x \odot N$, or $x <_{x_2}^{x_1} NP$, $x_1 \in Fv(N)$, $x_2 \in Fv(P)$ (in each case some N_i or P are *not* in normal form). N_i and P are typeable by IH, as subterms of M. Then, it is easy to build the typing for M. □

5.2 SN ⇒ Typeability in $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$

Finally, we want to prove that if a $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-term is SN, then it is typeable in the system $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$. We follow the procedure used in Section 5.1. The proofs are similar to the ones in Section 5.1 and are omitted.

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

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

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

Proposition 19 *$\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-normal forms are typeable in the system $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$.*

Proof: The proof goes by an easy induction on the structure of e_{nf}. □
The following two lemmas explain the behavior of the meta operators [/] and @ during expansion.

Lemma 10 (Inverse substitution lemma)

(i) Let $\Gamma \vdash_{\mathcal{R}} t[u/x] : \sigma$ *and u typeable. Then, there exist* $\Delta = \Delta_1 \sqcup ... \sqcup \Delta_n$ *and* $\cap_i^n \tau_i$, $i = 1,...,n$ *such that* $\Delta_i \vdash_{\mathcal{R}} u : \tau_i$ *for all* $i = 1,\ldots,n$ *and* $\Gamma', x : \cap\tau_i \vdash_{\mathcal{R}} t : \sigma$, *where* $\Gamma = \Gamma' \sqcup_{\mathsf{c}} \Delta$.

(ii) Let $\Gamma; \alpha \vdash_{\mathcal{R}} k[u/x] : \sigma$ *and u typeable. Then, there exist* $\Delta = \Delta_1 \sqcup ... \sqcup \Delta_n$ *and* $\cap_i^n \tau_i$, $i = 1,...,n$ *such that* $\Delta_i \vdash_{\mathcal{R}} u : \tau_i$ *for all* $i = 1,\ldots,n$ *and* $\Gamma', x : \cap\tau_i; \alpha \vdash_{\mathcal{R}} k : \sigma$, *where* $\Gamma = \Gamma' \sqcup_{\mathsf{c}} \Delta$.

Proof: The proof goes straightforward by mutual induction on the structure of terms and contexts. □

Lemma 11 (Inverse append lemma) *If* $\Gamma; \alpha \vdash_{\mathcal{R}} k@k' : \sigma$, *then* $\Gamma = \Gamma' \sqcup_{\mathsf{c}} \Gamma''$ *where* $\Gamma' = \Gamma'_1 \sqcup ... \sqcup \Gamma'_n$ *and there is a type* $\cap_i^n \tau_i$, $i = 1,...,n$ *such that* $\Gamma'_i; \alpha \vdash_{\mathcal{R}} k : \tau_i$ *for all* $i = 1,\ldots,n$ *and* $\Gamma''; \cap_i^n \tau_i \vdash_{\mathcal{R}} k' : \sigma$.

Proof: The proof goes by the induction on the structure of the context k. □
Now we prove that the type of an expression is preserved during the expansion.

Proposition 20 (Head subject expansion)

(i) For every $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$*-term t: if* $t \to t'$, *t is contracted redex and* $\Gamma \vdash_{\mathcal{R}} t' : \sigma$, *then* $\Gamma \vdash_{\mathcal{R}} t : \sigma$.

(ii) For every $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$*-context k: if* $k \to k'$, *k is contracted redex and* $\Gamma; \alpha \vdash_{\mathcal{R}} k' : \sigma$, *then* $\Gamma; \alpha \vdash_{\mathcal{R}} k' : \sigma$.

Proof: The proof goes by the case study according to the applied reduction. □

Theorem 5 (SN ⇒ typeability) *All strongly normalising* $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$ *expressions are typeable in the* $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$ *systems.*

Proof: The proof is by induction over the length of the longest reduction path out of a strongly normalising expression e, with a subinduction on the size of e. We also use Proposition 10. □
The complete characterisation of strongly normalising terms by intersection types

- in the natural deduction ND-base of the resource control cube is a corollary to Theorem 1 and 4
- in the sequent LJ-base of the resource control cube is a corollary to Theorem 3 and 5.

Theorem 6 (Complete characteristation of SN)

- *A* $\lambda_{\mathcal{R}}$*-term is strongly normalising if and only if it is typeable in* $\lambda_{\mathcal{R}}\cap$.
- *A* $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$*-term is strongly normalising if and only if it is typeable in* $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}\cap$.

6 Conclusions

In this paper, we proposed intersection type assignment systems for $\lambda_{\mathcal{R}}$ and $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi, two systems of lambda calculi parametrized with respect to $\mathcal{R} \subseteq \{\mathsf{c}, \mathsf{w}\}$, where c is a contraction and w is a weakening. These two families of lambda calculi form the so-called *resource control cube*. Four $\lambda_{\mathcal{R}}$-calculi form the "natural deduction base" of the cube, corresponding to the "implicit base" of [KR09], whereas "the sequent base" contains four $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi, generalization of $\ell\lambda^{\mathsf{Gtz}}$-calculus of [GILŽ11]. In each base, the calculi differ by the implicit/explicit treatment of the resource operators contraction and weakening.

The intersection type systems proposed here, for resource control lambda and sequent lambda calculus, give a complete characterisation of strongly normalising terms for all eight calculi of the resource cube. We propose general proofs for each base handled by various side conditions in some cases. In order to prove the strong normalisation of

typeable resource lambda terms, we use an appropriate modification of the reducibility method. The same property for resource sequent lambda expressions is proved by using a well-founded lexicographic order based on suitable embedding into the former calculi. This paper expands the range of the intersection type techniques and combines different methods in the strict types environment. Unlike the approach of introducing non-idempotent intersection into the calculus with some kind of resource management [PR10], our intersection is idempotent.

It would be interesting to investigate the relation between the resource control enabled via explicit operators used here and the approach used in [PR10], where the resources are managed via applicative bags with multiplicities. Another direction will involve the investigation of the use of intersection types in constructing models for sequent lambda calculi, since intersection types are known powerful means for building models of lambda calculus ([BCDC83, DCGL04]). On the other hand it should be noticed that the substitutions in our $\lambda_{\mathcal{R}}^{\mathsf{Gtz}}$-calculi are implicit. Considering explicit susbtitutions would complete the sequent part of the cube as Kesner and Renaud have done for $\lambda_{\mathcal{R}}$ in their prismoid.

Furthermore, resource control lambda and sequent lambda calculi are good candidates to investigate the computational content of substructural logics [SHD93], both in natural deduction and sequent calculus. The motivation for these logics comes from philosophy (Relevant and Affine Logic), linguistics (Lambek Calculus) and computing (Linear Logic). The basic idea of resource control is to explicitly handle structural rules, so that the absence of (some) structural rules in substructural logics such as weakening, contraction, commutativity, associativity can possibly be handled by resource control operators. This is in the domain of further research.

From a more pragmatic perspective, resources need to be controlled tightly in computer applications. For instance, Kristoffer Rose has undertaken the description of compilers by rules with binders [Ros11b, Ros11a]. He noticed that the implementation of substitutions of linear variables by inlining is efficient, whereas substitutions of duplicated variables require a cumbersome and time consuming mechanism, based on pointers. It is therefore important to precisely control duplications. On the other hand, strong control of erasing does not require a garbage collector and prevents memory leaking. Another line of application of resource control is related to object-oriented languages. Alain Mycroft [Myc11] presented resource aware type-systems for multi-core program efficiency. In this framework an identified "memory isolation" property enables multi-core programs to avoid slowdown due to cache contention. The existing work on Kilim and its isolation-type system is related to both substructural types and memory isolation.

Finally, the two calculi with both resource control operators explicit, namely $\lambda_{\mathtt{cw}}$ of [KR09] and $\ell\lambda^{\mathsf{Gtz}}$-calculus of [GILŽ11] deserve particular attention. Due to the multiplicative style of the typing rules, and the reductions' orientation of propagating the contraction in the term and extracting the weakening out of the term, these calculi exhibit optimization in terms of the minimal total size of the bases used for the type assignments. The consequences of this property, particularly for the implementation related issues, should be investigated.

Acknowledgements Above all, our gratitude goes to two anonymous referees for the present submission. Their extensive and detailed comments helped us tremendously to improve our work. We would also like to thank Dragiša Žunić for fruitful discussion.

References

[AH03] Z. M. Ariola and H. Herbelin. Minimal classical logic and control operators. In J. C. M. Baeten, J. K. Lenstra, and G. J. Woeginger J. Parrow, editors, *30th International Colloquium on Automata, Languages and Programming, ICALP '03*, volume 2719 of *LNCS*, pages 871–885. Springer, 2003.

[Bar84] H. P. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. North-Holland, Amsterdam, revised edition, 1984.

[Bar92] H. P. Barendregt. Lambda calculi with types. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, UK, 1992.

[BB96] F. Barbanera and S. Berardi. A symmetric lambda calculus for classical program extraction. *Inform. Comput.*, 125(2):103–117, 1996.

[BCDC83] H. P. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *J. Symb. Logic*, 48(4):931–940 (1984), 1983.

[BCL99] G. Boudol, P.-L. Curien, and C. Lavatelli. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science*, 9(4):437–482, 1999.

[BG00] H. P. Barendregt and S. Ghilezan. Lambda terms for natural deduction, sequent calculus and cut-elimination. *J. Funct. Program.*, 10(1):121–134, 2000.

[Bie98] G. M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In L. Brim, J. Gruska, and J. Zlatuska, editors, *23rd International Symposium on Mathematical Foundations of Computer Science, MFCS '98*, volume 1450 of *LNCS*, pages 336–345. Springer, 1998.

[BN98] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, UK, 1998.

[Bou93] G. Boudol. The lambda-calculus with multiplicities (abstract). In E. Best, editor, *4th International Conference on Concurrency Theory, CONCUR '93*, volume 715 of *LNCS*, pages 1–6. Springer, 1993.

[BR95] R. Bloo and K. H. Rose. Preservation of strong normalisation in named lambda calculi with explicit substitution and garbage collection. In *Computer Science in the Netherlands, CSN '95*, pages 62–72, 1995.

[CDC78] M. Coppo and M. Dezani-Ciancaglini. A new type-assignment for lambda terms. *Archiv für Mathematische Logik*, 19:139–156, 1978.

[CDC80] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame J. Formal Logic*, 21(4):685–693, 1980.

[CDCV80] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal type schemes and λ-calculus semantics. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 535–560. Academic Press, London, 1980.

[CH00] P.-L. Curien and H. Herbelin. The duality of computation. In *5th International Conference on Functional Programming, ICFP'00*, pages 233–243. ACM Press, 2000.

[Chu41] A. Church. *The Calculi of Lambda-Conversion*. Princeton University Press, Princeton, 1941.

[DCG03] M. Dezani-Ciancaglini and S. Ghilezan. Two behavioural lambda models. In H. Geuvers and F. Wiedijk, editors, *Types for Proofs and Programs*, volume 2646 of *LNCS*, pages 127–147. Springer, 2003.

[DCGL04] M. Dezani-Ciancaglini, S. Ghilezan, and S. Likavec. Behavioural Inverse Limit Models. *Theor. Comput Sci.*, 316(1–3):49–74, 2004.

[DCHM00] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional characterization of λ-terms using intersection types. In *25th International Symposium on Mathematical Foundations of Computer Science, MFCS '00*, volume 1893 of *LNCS*, pages 304–314. Springer, 2000.

[dG94] Ph. de Groote. On the relation between the $\lambda\mu$-calculus and the syntactic theory of sequential control. In F. Pfenning, editor, *5th International Conference on Logic Programming and Artificial Reasoning, LPAR'94*, volume 822 of *LNCS*, pages 31–43. Springer, 1994.

[DGL08] D. J. Dougherty, S. Ghilezan, and P. Lescanne. Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: extending the Coppo-Dezani heritage. *Theor. Comput Sci.*, 398:114–128, 2008.

[EGI08] J. Espírito Santo, S. Ghilezan, and J. Ivetić. Characterising strongly normalising intuitionistic sequent terms. In *International Workshop TYPES'07 (Selected Papers)*, volume 4941 of *LNCS*, pages 85–99. Springer, 2008.

[EIL11] J. Espírito Santo, J. Ivetić, and S. Likavec. Characterising strongly normalising intuitionistic terms. *Fundamenta Informaticae*, 2011. to appear.

[EP03] J. Espírito Santo and L. Pinto. Permutative conversions in intuitionistic multiary sequent calculi with cuts. In *6th International Conference on Typed Lambda Calculi and Applications, TLCA '03*, volume 2071 of *LNCS*, pages 286–300, 2003.

[ER03] T. Ehrhard and L. Regnier. The differential lambda-calculus. *Theor. Comput. Sci.*, 309(1-3):1–41, 2003.

[Esp07a] J. Espírito Santo. Completing Herbelin's programme. In S. Ronchi Della Rocca, editor, *9th International Conference on Typed Lambda Calculi and Applications, TLCA '07*, volume 4583 of *LNCS*, pages 118–132. Springer, 2007.

[Esp07b] J. Espírito Santo. Delayed substitutions. In F. Baader, editor, *18th International Conference on Term Rewriting and Applications, RTA'07*, LNCS, pages 169–183. Springer, 2007.

[Gal98] J. Gallier. Typing untyped λ-terms, or reducibility strikes again! *Ann. Pure Appl. Logic*, 91:231–270, 1998.

[Ghi96] S. Ghilezan. Strong normalization and typability with intersection types. *Notre Dame J. Formal Logic*, 37(1):44–52, 1996.

[GILL11] S. Ghilezan, J. Ivetić, P. Lescanne, and S. Likavec. Intersection types for the resource control lambda calculi. In A. Cerone and P. Pihlajasaari, editors, *8th International Colloquium on Theoretical Aspects of Computing, ICTAC '11*, volume 6916 of *LNCS*, pages 116–134. Springer, 2011.

[GILŽ11] S. Ghilezan, J. Ivetić, P. Lescanne, and D. Žunić. Intuitionistic sequent-style calculus with explicit structural rules. In *8th International Tbilisi Symposium on Language, Logic and Computation*, volume 6618 of *LNAI*, pages 101–124, 2011.

[Gir71] J.-Y. Girard. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'elimination des coupures dans l'analyse et la théorie des types. In J. E. Fenstad, editor, *2nd Scandinavian Logic Symposium*, pages 63–92. North-Holland, 1971.

[Gir87] J.-Y. Girard. Linear logic. *Theor. Comput Sci.*, 50:1–102, 1987.

[GL09] S. Ghilezan and S. Likavec. Computational interpretations of logics. In Z. Ognjanović, editor, *Collection of Papers, special issue Logic in Computer Science 20(12)*, pages 159–215. Mathematical Institute of Serbian Academy of Sciences and Arts, 2009.

[Gri90] T. Griffin. A formulae-as-types notion of control. In *9th Annual ACM Symposium on Principles Of Programming Languages, POPL '90*, pages 47–58. ACM Press, 1990.

[Her95] H. Herbelin. A lambda calculus structure isomorphic to Gentzen-style sequent calculus structure. In L. Pacholski and J. Tiuryn, editors, *Computer Science Logic, CSL '94*, volume 933 of *LNCS*, pages 61–75. Springer, 1995.

[HG08] H. Herbelin and S. Ghilezan. An approach to call-by-name delimited continuations. In *35th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '08*, pages 383–394. ACM Press, 2008.

[How80] W. A. Howard. The formulas-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, London, 1980.

[Kik07] K. Kikuchi. Simple proofs of characterizing strong normalisation for explicit substitution calculi. In F. Baader, editor, *18th International Conference on Term Rewriting and Applications, RTA'07*, volume 4533 of *LNCS*, pages 257–272. Springer, 2007.

[KL07] D. Kesner and S. Lengrand. Resource operators for lambda-calculus. *Inform. Comput.*, 205(4):419–473, 2007.

[Kle52] S. C. Kleene. *Introduction to Metamathematics*. North-Holland, Amsterdam, 1952.

[Kol85] G. Koletsos. Church-Rosser theorem for typed functionals. *J. Symb. Logic*, 50:782–790, 1985.

[KR09] D. Kesner and F. Renaud. The prismoid of resources. In R. Královič and D. Niwiński, editors, *34th International Symposium on Mathematical Foundations of Computer Science, MFCS '09*, volume 5734 of *LNCS*, pages 464–476. Springer, 2009.

[KR11] D. Kesner and F. Renaud. A prismoid framework for languages with resources. *Theor. Comput. Sci.*, 412(37):4867–4892, 2011.

[Kri90] J.-L. Krivine. *Lambda-calcul types et modèles*. Masson, Paris, 1990.

[Mat00] R. Matthes. Characterizing strongly normalizing terms of a λ-calculus with generalized applications via intersection types. In J. Rolin et al., editor, *ICALP Workshops 2000*. Carleton Scientific, 2000.

[Myc11] A. Mycroft. Using Kilim's Isolation Types for Multicore Efficiency. Invited talk at FoVeOOS 2011 - 2nd International Conference on Formal Verification of Object-Oriented Software, October 2011.

[Nee05] P. M. Neergaard. Theoretical pearls: A bargain for intersection types: a simple strong normalization proof. *J. Funct. Program.*, 15(5):669–677, 2005.

[OS97] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '97*, pages 215–227. ACM, 1997.

[Par92] M. Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *3rd International Conference on Logic Programming and Automated Reasoning, LPAR '92*, volume 624 of *LNCS*, pages 190–201. Springer, 1992.

[Par97] M. Parigot. Proofs of strong normalisation for second order classical natural deduction. *J. Symb. Logic*, 62(4):1461–1479, 1997.

[Pot80] G. Pottinger. A type assignment for the strongly normalizable λ-terms. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 561–577. Academic Press, London, 1980.

[PR10] M. Pagani and S. Ronchi Della Rocca. Solvability in resource lambda-calculus. In C.-H. L. Ong, editor, *13th International Conference on Foundations of Software Science and Computational Structures, FOSSACS 2010*, volume 6014 of *LNCS*, pages 358–373. Springer, 2010.

[Reg94] L. Regnier. Une équivalence sur les lambda-termes. *Theor. Comput Sci.*, 126(2):281–292, 1994.

[Ros11a] K H. Rose. CRSX - Combinatory Reduction Systems with Extensions. In Manfred Schmidt-Schauß, editor, *22nd International Conference on Rewriting Techniques and Applications, RTA'11*, volume 10 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 81–90. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2011.

[Ros11b] K. H. Rose. Implementation Tricks That Make CRSX Tick. Talk at IFIP 1.6 workshop, RDP'2011, May 2011.

[Sal78] P. Sallé. Une extension de la théorie des types en lambda-calcul. In G. Ausiello and C. Böhm, editors, *5th International Conference on Automata, Languages and Programming, ICALP '78*, volume 62 of *LNCS*, pages 398–410. Springer, 1978.

[SHD93] P. Schroeder-Heister and K. Došen. *Substructural Logics*. Oxford University Press, UK, 1993.

[Sta85] R. Statman. Logical relations and the typed λ-calculus. *Inform. Control*, 65:85–97, 1985.

[SU06] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard isomorphism*. Studies in Logic and the Foundations of Mathematics, 149. Elsevier, 2006.

[Tai67] W. W. Tait. Intensional interpretations of functionals of finite type I. *J. Symb. Logic*, 32:198–212, 1967.

[Tai75] W. W. Tait. A realizability interpretation of the theory of species. In R. Parikh, editor, *Logic Colloquium*, volume 453 of *Lecture Notes in Mathematics*, pages 240–251. Springer, 1975.

[vB92] S. van Bakel. Complete restrictions of the intersection type discipline. *Theor. Comput Sci.*, 102(1):135–163, 1992.

[vO01] V. van Oostrom. Net-calculus. Course notes, http://www.phil.uu.nl/ oostrom/oudonderwijs/cmitt/00-01/net.ps, 2001.

[Ž07] D. Žunić. *Computing with sequents and diagrams in classical logic - calculi ${}^*\mathcal{X}$, ${}^d\mathcal{X}$ and ${}^{\copyright}\mathcal{X}$*. Phd thesis, École Normale Supérieure de Lyon, December 2007.