Intersection and Union Type Assignment and Polarised $\bar{\lambda}\mu\tilde{\mu}$

Takeshi Tsukada[1] and Koji Nakazawa[2]

[1] The University of Tokyo
[2] Nagoya University

Abstract. Intersection and union type assignment systems are powerful tools for reasoning programs that completely characterise many semantic properties such as strong normalisation. At the same time, they are known to be subtle, particularly in the presence of computational effect. To address the difficulty, this paper develops an approach based on polarities and refinement intersection type systems.

We introduce a simply-typed polarised calculus, in which a type is either positive or negative, based on Curien and Herbelin's calculus $\bar{\lambda}\mu\tilde{\mu}$. This polarised calculus interacts well with intersection and union types: by adding intersection on positive types and union on negative types, we obtain a sound and complete type system. One can design an intersection and union type system for another calculus, guided by a translation to the polarised calculus. To demonstrate usefulness of the approach, we derive the first intersection and union type system for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$, which satisfies expected properties.

1 Introduction

Intersection and union type assignment systems [6,5,23] are an attractive research area because of their expressive power and of their subtlety. They can completely characterise interesting semantic properties such as solvability and (weak and strong) normalisation of terms [7]. It recently serves as a basis of model-checking higher-order programs [18] and automated verification of functional programs [17]. At the same time, it is well-known that intersection and union types are subtle. For example, fairly natural introduction and elimination rules for union types lead to a type system in which neither subject reduction nor subject expansion do not hold (see, e.g., [5, Section 2]). The situation is more difficult in the presence of computational effect. Davies and Pfenning [10] observed that the standard rules for intersection types are unsound for a call-by-value language with references and show that a value restriction on intersection introduction leads to a sound system. As for unions, Dunfield and Pfennig [13] noticed that the standard rules are unsound for effectful languages and one needs evaluation-context restriction on union elimination.

Hence it is important to investigate a principled way to design sound (and, if possible, complete) intersection and union type systems for an effectful calculus.

This paper studies an approach based on polarities and refinement intersection and union types. Polarities [15] are a logical notion that allows us to describe evaluation strategies at the level of formulas (or types). A refinement type [14] describes a detailed property of a program that is already typed by a coarser type system.

In this paper, we employ a polarised system as a coarser type system and develop an intersection and union type system that refines the polarised system. What we found is that intersection and union interacts quite well with a polarised system, in contrast to call-by-name or call-by-value calculi. One can design an intersection and union type system for a call-by-value calculus guided by a translation from the call-by-value calculus to the polarised calculus.

To demonstrate usefulness of this approach, we derive an intersection and union type system for the (untyped version of the) call-by-value $\bar{\lambda}\mu\tilde{\mu}$ [9]. This has been still open, although several researches has addressed it (e.g. [11,12,2]).

The connection between polarities and refinement intersection and union types has already been observed and discussed by Zeilberger [30], using his calculus based on focusing proofs [29]. His work and our work are in the complementary relationship: see Section 6.

Organisation of the paper Section 2 introduces our polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$, to which an intersection and union type system is developed in Section 3. We discuss translations from call-by-value and call-by-name $\bar{\lambda}\mu\tilde{\mu}$ to the polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$ in Section 4. Section 5 describes how to design an intersection and union type system from a translation to $\bar{\lambda}\mu\tilde{\mu}_P$. Related work is discussed in Section 6.

2 Polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$

This section introduces a polarised calculus named $\bar{\lambda}\mu\tilde{\mu}_P$, as well as that with recursive types. As the name suggests, the calculus is based on Curien and Herbelin's $\bar{\lambda}\mu\tilde{\mu}$ [9], a symmetric calculus corresponding to the classical sequent calculus. The main difference from $\bar{\lambda}\mu\tilde{\mu}$ lies in the cut rule: $\bar{\lambda}\mu\tilde{\mu}$ has one cut rule, whereas $\bar{\lambda}\mu\tilde{\mu}_P$ has two rules, one for each polarity. The evaluation strategy is built-in as the directions of cuts. Because of this change, $\bar{\lambda}\mu\tilde{\mu}_P$ has no critical pair and enjoys confluence. The calculus is carefully designed so that it has a strong connection to the typed λ-calculus with pairs.

We shall call the simple types of the calculus *sorts* in order to avoid confusion with intersection and union types introduced in the next section.

2.1 Preliminaries: $\bar{\lambda}\mu\tilde{\mu}$

First we briefly review the syntax and operational semantics of $\bar{\lambda}\mu\tilde{\mu}$ [9]. We shall not see the simple type system in [9].

The syntax of *terms*, *commands* and *co-terms* is given by:

$$v ::= x \mid \lambda x.v \mid \mu\alpha.c \qquad c ::= \langle v \mid e \rangle \qquad e ::= \alpha \mid v \cdot e \mid \tilde{\mu}x.c,$$

where x and α are variables of different kinds. *Values* and *co-values* are distinguished terms and co-terms, respectively, defined by $V ::= x \mid \lambda x.v$ and $E ::= \alpha \mid v \cdot e$. There are three reduction rules:

$$\langle \lambda x.v \mid v' \cdot e \rangle \longrightarrow \langle v' \mid \tilde{\mu} x.\langle v \mid e \rangle \rangle \quad \langle \mu\alpha.c \mid e \rangle \longrightarrow c[e/\alpha] \quad \langle v \mid \tilde{\mu} x.c \rangle \longrightarrow c[v/x].$$

The calculus has a critical pair $\langle \mu\alpha.c \mid \tilde{\mu} x.c' \rangle$, to which both of the last two rules are applicable, and is not confluent. The conflict is resolved by an evaluation strategy. The former rule has the priority in the call-by-value calculus, and the latter has in call-by-name. Hence, in the call-by-value calculus, the third rule is applicable only if $v \neq \mu\alpha.c'$, and thus changed to $\langle V \mid \tilde{\mu} x.c \rangle \longrightarrow c[V/x]$. Similarly, in call-by-name, the second rule is changed to $\langle \mu\alpha.c \mid E \rangle \longrightarrow c[E/\alpha]$.

We write $=_\beta^{\mathrm{cbv}}$ for the equivalence relation on expressions induced by the call-by-value reduction relation, and $=_\beta^{\mathrm{cbn}}$ for call-by-name.

2.2 Terms of $\bar{\lambda}\mu\tilde{\mu}_P$

Similar to $\bar{\lambda}\mu\tilde{\mu}$, the calculus $\bar{\lambda}\mu\tilde{\mu}_P$ has three kinds of *expressions*: positive ones called *terms*, negative ones called *co-terms*, and neutral ones called *commands*. Assume infinite sets of *positive variables*, ranged over by x, y and z, and *negative variables*, ranged over by α, β and γ. The syntax of expressions is given by:

$$\begin{array}{ll} \textit{(Terms)} & v ::= x \mid \lambda x.v \mid \mu\alpha.c \mid v \triangleleft e \\ \textit{(Commands)} & c ::= \langle v \downarrow e \rangle \mid \langle v \uparrow e \rangle \\ \textit{(Co-terms)} & e ::= \alpha \mid \lambda\alpha.e \mid \mu x.c \mid v \triangleright e. \end{array}$$

We use a metavariable t for expressions. Unlike $\bar{\lambda}\mu\tilde{\mu}$, we do not put tilde on the μ-construct for co-terms and the two μ-constructs are distinguished by their binding variable.

We have two command constructors $\langle v \downarrow e \rangle$ and $\langle v \uparrow e \rangle$ that correspond to $\langle v \mid e \rangle$ of $\bar{\lambda}\mu\tilde{\mu}$ [9]. The arrow indicates which of the term and the co-term has the priority: the term v has the priority in $\langle v \downarrow e \rangle$, whereas the co-term e has the priority in $\langle v \uparrow e \rangle$. The distinction controls the reduction as we shall see in the next subsection.

2.3 Reduction

We write $t[v/x]$ for the capture-avoiding substitution of v for x in t, and $t[e/\alpha]$ for the substitution of e for α in t. The calculus has 4 reduction rules, namely, positive/negative λ/μ reductions. Note that the positive and negative rules are completely symmetric. The rules are listed below:

$$\begin{array}{ll} \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle \longrightarrow \langle (v_1[v_2/x]) \downarrow e \rangle & \langle (\mu\alpha.c) \downarrow e \rangle \longrightarrow c[e/\alpha] \\ \langle (v \triangleleft e_1) \uparrow (\lambda\alpha.e_2) \rangle \longrightarrow \langle v \uparrow (e_2[e_1/\alpha]) \rangle & \langle v \uparrow (\mu x.c) \rangle \longrightarrow c[v/x] \end{array}$$

The reduction is supposed to be full, i.e. any redex at any position can be reduced. We write $\longrightarrow^*$ for the reflexive and transitive closure of $\longrightarrow$, and $=_\beta$ for

the least equivalence containing $\longrightarrow$. Some expression gets stuck, e.g. $\langle (\lambda x.v_1) \Rsh (v_2 \rhd e) \rangle$ and $\langle (\lambda x.v_1) \Lsh (\mu y.c) \rangle$.

Our calculus $\bar{\lambda}\mu\tilde{\mu}_P$ has no critical pair and enjoys confluence. In other words, the result of the evaluation is not affected by the order of reductions. The evaluation strategy is just built-in as the direction of cuts.

Proposition 1 (Confluence). *If $t \longrightarrow^* t_1$ and $t \longrightarrow^* t_2$, there exists t' such that $t_1 \longrightarrow^* t'$ and $t_2 \longrightarrow^* t'$.*

Remark 1. Recall that $\bar{\lambda}\mu\tilde{\mu}$ [9] has a critical pair $\langle \mu\alpha.c \mid \tilde{\mu}x.c' \rangle$:

$$c[\tilde{\mu}x.c'/\alpha] \longleftarrow \langle \mu\alpha.c \mid \tilde{\mu}x.c' \rangle \longrightarrow c'[\mu\alpha.c/x].$$

Each of the two cuts in $\bar{\lambda}\mu\tilde{\mu}_P$ allows only one of the above reductions:

$$c[\tilde{\mu}x.c'/\alpha] \longleftarrow \langle \mu\alpha.c \Lsh \mu x.c' \rangle \quad \text{or} \quad \langle \mu\alpha.c \Rsh \mu x.c' \rangle \longrightarrow c'[\mu\alpha.c/x].$$

2.4 Simple Sort System

Polarised sorts A sort of the calculus $\bar{\lambda}\mu\tilde{\mu}_P$ has its *polarity* [15], i.e. a sort is either *positive* or *negative*. We assume countably infinite sets of positive and negative atomic sorts, ranged over by p and n, respectively. The positive sorts are those for terms, and the negative sorts are those for co-terms.

The syntax of *simple polarised sorts* is given by:

$$\begin{array}{ll} \textit{(Positive sorts)} & P, Q ::= p \mid P \leftarrow N \mid {\downarrow}N \\ \textit{(Negative sorts)} & N, M ::= n \mid P \to N \mid {\uparrow}P. \end{array}$$

The sort $P \to N$ is the sort for the *continuations of functions* from P to N and the sort $\uparrow P$ is for continuations with a hole of sort P. Note that a continuation $\uparrow P$ with a positive hole is a negative sort. An abstraction $\lambda x.v$ is considered as a continuation of continuations of functions and has sort $\downarrow(P \to N)$.

It might be helpful to think that positive sorts are for values and negative sorts are for continuations. By the symmetry, one can also consider that negatives are for (co-)values and positives are for continuations of negatives.

Sort environments and judgements A *positive sort environment* Γ (resp. *negative sort environment* Δ) is a finite set of sort bindings of the form $x :: P$ (resp. $\alpha :: N$). We use double colon for sort bindings, reserving single colon for type bindings. A sort environment is considered as a set and the order of sort bindings is not significant. We write Γ, Γ' for $\Gamma \cup \Gamma'$.

There are tree kinds of *sort judgements*, which have both positive and negative sort environments and a subject depending on its kind. The first one $\Gamma \vdash v :: P \mid \Delta$ means that the term v has the positive sort P under the environments Γ and Δ. The second one $\Gamma \mid e :: N \vdash \Delta$, which is the dual to the first one, means that the co-term e has the negative sort N. The third one $c :: (\Gamma \vdash \Delta)$ means that the command c is well-sorted.

Sorting rules The sorting rules are listed below.

$$\frac{}{\Gamma, x :: P \vdash x :: P \mid \Delta} \qquad \frac{\Gamma, x :: P \vdash v :: \downarrow N \mid \Delta}{\Gamma \vdash \lambda x.v :: \downarrow(P \to N) \mid \Delta} \qquad \frac{c :: (\Gamma \vdash \alpha :: N, \Delta)}{\Gamma \vdash \mu\alpha.c :: \downarrow N \mid \Delta}$$

$$\frac{}{\Gamma \mid \alpha :: N \vdash \alpha :: N, \Delta} \qquad \frac{\Gamma \mid e :: \uparrow P \vdash \alpha :: N, \Delta}{\Gamma \mid \lambda\alpha.e :: \uparrow(P \leftarrow N) \vdash \Delta} \qquad \frac{c :: (\Gamma, x :: P \vdash \Delta)}{\Gamma \mid \mu x.c :: \uparrow P \vdash \Delta}$$

$$\frac{\Gamma \vdash v :: Q \mid \Delta \qquad \Gamma \mid e :: N \vdash \Delta}{\Gamma \vdash v \triangleleft e :: Q \leftarrow N \mid \Delta} \qquad \frac{\Gamma \vdash v :: \downarrow N \mid \Delta \qquad \Gamma \mid e :: N \vdash \Delta}{\langle v \downarrow e \rangle :: (\Gamma \vdash \Delta)}$$

$$\frac{\Gamma \mid e :: N \vdash \Delta \qquad \Gamma \vdash v :: P \mid \Delta}{\Gamma \mid v \triangleright e :: P \to N \vdash \Delta} \qquad \frac{\Gamma \mid e :: \uparrow P \vdash \Delta \qquad \Gamma \vdash v :: P \mid \Delta}{\langle v \uparrow e \rangle :: (\Gamma \vdash \Delta)}$$

By ignoring the subject, separator $\mid$ and shifts $\uparrow\cdot$ and $\downarrow\cdot$ in sorts, they are the rules of the classical seqent calculus (where $P \leftarrow N$ is understood as $P \wedge (\neg N)$).

Proposition 2. *If $\Gamma \vdash v :: P \mid \Delta$ and $v \longrightarrow^* v'$, then $\Gamma \vdash v' :: P \mid \Delta$. The similar statements hold for reduction of co-terms and commands.*

2.5 Recursive Sorts

The simply-sorted calculus $\bar{\lambda}\mu\tilde{\mu}_P$ is so weak that untyped calculi (e.g. the untyped call-by-value $\bar{\lambda}\mu\tilde{\mu}$) cannot be embedded into $\bar{\lambda}\mu\tilde{\mu}_P$.

We extend $\bar{\lambda}\mu\tilde{\mu}_P$ by recursive sorts. Assume finite sets of *positive and negative sort variables*, for which we use the typewriter font $\mathtt{T}$. Each positive (resp. negative) sort variable $\mathtt{T}$ is equipped with an equation $\mathtt{T} = P$ (resp. $\mathtt{T} = N$), where P (resp. N) is a sort constructed from atomic sorts and sort variables. We require that, for every equation $\mathtt{T} = P$ (resp. $\mathtt{T} = N$), P (resp. N) is not a sort variable. Let us write $\sim$ for the congruence induced by the equations. For example, if we have a positive sort variable $\mathtt{V} = \downarrow(\mathtt{V} \to \uparrow\mathtt{V})$, then $(\mathtt{V} \to n) \sim (\downarrow(\mathtt{V} \to \uparrow\mathtt{V}) \to n)$. In the extended system, we deal with sorts modulo $\sim$.

The extended calculus also enjoys Confluence and Subject Reduction. Furthermore one can embed some untyped calculi into the extended calculus. For example, $\mathtt{V}$ above is the sort used in the embedding of the call-by-value $\bar{\lambda}\mu\tilde{\mu}$.

2.6 Translation to $\boldsymbol{\lambda^{\to\times}}$

The calculus $\bar{\lambda}\mu\tilde{\mu}_P$ is strongly related to the typed λ-calculus $\lambda^{\to\times}$ (with recursive sorts) via the translation described below.

We assume a fixed simple sort O of $\lambda^{\to\times}$, called the *response sort*. Polarised sorts P and N of $\bar{\lambda}\mu\tilde{\mu}_P$ are translated into sorts P^* and N^* of $\lambda^{\to\times}$ as follows:

$$(P \leftarrow N)^* := P^* \times N^* \qquad (P \to N)^* := P^* \times N^*$$
$$(\downarrow N)^* := (N^*) \to O \qquad (\uparrow P)^* := (P^*) \to O,$$

where we assume a fixed translation of atomic sorts. Note that a function type of $\lambda^{\to\times}$ appearing in the translation must be of the form $(-) \to O$ as in Thielecke-style CPS translation [28] into a calculus with "negation" and pairs.

For example, the translation of the sort $\downarrow(P \to N)$ for (positive) lambda abstractions is $(\downarrow(P \to N))^* = (P^* \times N^*) \to O$. Translation of a type environment is component-wise, e.g., $(x_1 : P_1, \ldots, x_k : P_k)^* := x_1 : P_1^*, \ldots, x_k : P_k^*$. A recursive sort defined by $\mathtt{T} = P$ is translated to $\mathtt{T}^*$ with $\mathtt{T}^* = P^*$.

Translation of expressions is defined by:

$$\begin{array}{llll} (x)^* := x & (\lambda x.v)^* := \lambda(x,\beta).(v^*\,\beta) & (\mu\alpha.c)^* := \lambda\alpha.(c^*) & (v \triangleleft e)^* := (v^*, e^*) \\ (\alpha)^* := \alpha & (\lambda\alpha.e)^* := \lambda(y,\alpha).(e^*\,y) & (\mu x.c)^* := \lambda x.(c^*) & (v \triangleright e)^* := (v^*, e^*) \\ \langle e \mathrel{\triangleright\!\!\!\cdot} v\rangle^* := v^*\,e^* & \langle e \mathrel{\triangleleft\!\!\!\cdot} v\rangle^* := e^*\,v^*, & & \end{array}$$

where y and β are fresh variables not appearing in v nor e. Cut expressions, which mean context-filling, are translated into the application of the context to the term. The direction of the application is controlled by the direction, or the polarity, of the cut. In other words, the cut expression tells us which is a context and which is a term. This is in contrast to $\bar\lambda\mu\tilde\mu$, in which the direction of the cut is globally determined by the evaluation strategy.

The translation preserves typing and reduction.

Proposition 3. *We have*

$$\left\{\begin{array}{l} \Gamma \vdash v : P \mid \Delta \\ \Gamma \mid e : N \vdash \Delta \\ c : (\Gamma \vdash \Delta) \end{array}\right\} \text{ in } \bar\lambda\mu\tilde\mu_P \implies \left\{\begin{array}{l} \Gamma^*, \Delta^* \vdash v^* : P^* \\ \Gamma^*, \Delta^* \vdash e^* : N^* \\ \Gamma^*, \Delta^* \vdash c^* : O. \end{array}\right\} \text{ in } \lambda^{\to\times}.$$

Proof. Easy induction on the structure of the expression of $\bar\lambda\mu\tilde\mu_P$. □

Lemma 1. *Let t be a (possibly not well-sorted) expression of $\bar\lambda\mu\tilde\mu_P$.*

- *If $t \longrightarrow u$ in $\bar\lambda\mu\tilde\mu_P$, then $t^* \longrightarrow u^*$ in $\lambda^{\to\times}$.*
- *If $t^* \longrightarrow u$ in $\lambda^{\to\times}$, then $t \longrightarrow s$ in $\bar\lambda\mu\tilde\mu_P$ and $s^* = u$ for some s.*

Corollary 1. *A $\bar\lambda\mu\tilde\mu_P$-expression t is strongly normalising iff so is t^*.*

3 Refinement Intersection and Union Type System

This section develops a refinement intersection and union type system for $\bar\lambda\mu\tilde\mu_P$ that has intersection on positive types and union on negative types. Perhaps surprisingly, this simple extension works quite well. For example, (1) the set of types for an expression is preserved by $=_\beta$ and (2) strongly normalising expressions are completely characterised by the type system.

Fix finite sets of positive and negative sort variables as well as associated equations. Expressions in this section are implicitly equipped with their sorts.

Remark 2. The results of this section does not rely on well-sortedness of expressions (though the statement of Theorem 2 needs a slight modification). Expressions in the images of the translations from call-by-name and call-by-value $\bar\lambda\mu\tilde\mu$ are well-sorted, and the sort will play an important rôle in Section 5.

3.1 Intersection and Union Types

Syntax Assume a (possibly empty) set $\mathcal{X}_P$ (resp. $\mathcal{X}_N$) of atomic types for each positive sort P (resp. negative sort N). For simplicity, we assume $\mathcal{X}_P$, $\mathcal{X}_Q$, $\mathcal{X}_M$ and $\mathcal{X}_N$ are pairwise disjoint if $P \neq Q$ and $M \neq N$. We use a_P (resp. a_N) for a metavariable that ranges over $\mathcal{X}_P$ (reps. $\mathcal{X}_N$). The syntax of types is given by

$$\begin{array}{ll} \textit{(Positive raw types)} & \tau, \sigma ::= a_P \mid \tau \leftarrow \theta \mid \downarrow\theta \mid \tau \wedge \tau \mid \top_P \\ \textit{(Negative raw types)} & \theta, \delta ::= a_N \mid \tau \rightarrow \theta \mid \uparrow\tau \mid \theta \vee \theta \mid \bot_N. \end{array}$$

Compared with the syntax of sorts, we simply add intersection on positive and union on negative. A raw type is $\{\top,\bot\}$-*free* if it does not contain $\top_P$ nor $\bot_N$.

Refinement relation A *type* is a raw type that follows the structure of a sort. To describe this notion formally, we introduce the *refinement relation*:

$$\frac{a_P \in \mathcal{X}_P}{a_P :: P} \qquad \frac{\tau :: P \qquad \theta :: N}{\tau \leftarrow \theta :: P \leftarrow N} \qquad \frac{\theta :: N}{\downarrow\theta :: \downarrow N} \qquad \frac{\tau_1, \tau_2 :: P}{\tau_1 \wedge \tau_2 :: P} \qquad \frac{}{\top_P :: P}$$

$$\frac{a_N \in \mathcal{X}_N}{a_N :: N} \qquad \frac{\tau :: P \qquad \theta :: N}{\tau \rightarrow \theta :: P \rightarrow N} \qquad \frac{\tau :: P}{\uparrow\tau :: \uparrow P} \qquad \frac{\theta_1, \theta_2 :: N}{\theta_1 \vee \theta_2 :: N} \qquad \frac{}{\bot_N :: N}$$

Every raw type has at most one sort.

Example 1. A negative raw type $(\uparrow a) \vee (b \rightarrow c)$ is not a type.

Subtyping The subtyping relation is defined by the following rules.

$$\frac{a_P \in \mathcal{X}_P}{a_P \preceq_P a_P} \qquad \frac{\tau \preceq_P \sigma \qquad \theta \succeq_N \delta}{(\tau \leftarrow \theta) \preceq_{P\leftarrow N} (\tau \leftarrow \theta)} \qquad \frac{\theta \preceq_N \delta}{\downarrow\theta \preceq_{\downarrow N} \downarrow\delta}$$

$$\frac{a_N \in \mathcal{X}_N}{a_N \preceq_N a_N} \qquad \frac{\tau \succeq_P \sigma \qquad \theta \preceq_N \delta}{(\tau \rightarrow \theta) \preceq_{P\rightarrow N} (\sigma \rightarrow \delta)} \qquad \frac{\tau \preceq_P \sigma}{\uparrow\tau \preceq_{\uparrow P} \uparrow\sigma}$$

$$\frac{\tau \preceq_P \sigma_1 \qquad \tau \preceq_P \sigma_2}{\tau \preceq_P \sigma_1 \wedge \sigma_2} \qquad \frac{\tau_1 \wedge \tau_2 :: P \quad \exists i \in \{1,2\}.\ \tau_i \preceq_P \sigma}{\tau_1 \wedge \tau_2 \preceq_P \sigma} \qquad \frac{\tau :: P}{\tau \preceq_P \top_P}$$

$$\frac{\theta_1 \preceq_N \delta \qquad \theta_2 \preceq_N \delta}{\theta_1 \vee \theta_2 \preceq_N \delta} \qquad \frac{\delta_1 \vee \delta_2 :: N \quad \exists i \in \{1,2\}.\ \theta \preceq_N \delta_i}{\theta \preceq_N \delta_1 \vee \delta_2} \qquad \frac{\theta :: N}{\bot_N \preceq_N \theta}$$

If $\tau \preceq_P \sigma$, then $\tau :: P$ and $\sigma :: P$. We simply write $\tau \preceq \sigma$ if $\tau \preceq_P \sigma$ for some P. It is easy to show that the subtyping relation is transitive.

3.2 Typing Rules

Let us write Θ for a type environment for positive variables and Ξ for negative variables: $\Theta ::= \cdot \mid \Theta, x : \tau$ and $\Xi ::= \cdot \mid \Xi, \alpha : \theta$. For type environments, we assume that each variable occurs at most once. The refinement relation on

environments is defined by point-wise refinement: $\Theta :: \Gamma$ just if $\{x \mid x :: P \in \Gamma\} = \{x \mid x : \tau \in \Theta\}$ and, for all x, $x : \tau \in \Theta$ and $x :: P \in \Delta$ implies $\tau :: P$. The subtyping relation on environments is also defined by point-wise subtyping: given $\Theta_1 :: \Gamma$ and $\Theta_2 :: \Gamma$, we write $\Theta_1 \preceq_\Gamma \Theta_2$ (or simply $\Theta_1 \preceq \Theta_2$) if $x : \tau_1 \in \Theta_1$ and $x : \tau_2 \in \Theta_2$ implies $\tau_1 \preceq_P \tau_2$ (where $x :: P \in \Gamma$).

Recall that expressions and variables are implicitly equipped with their sorts. Hereafter we assume that a type environment respects the sorts. For example, if $x : \tau \in \Theta$ for a positive variable x of sort P, then we assume that $\tau :: P$. It might be better to write $x : \tau :: P$ to make the sort explicit, but we do not choose this heavy notation.

The intersection and union type system has the rules of the sort system in Section 2 (but for types, not sorts) and the following additional rules.

$$\frac{\Theta \vdash v : \tau_1 \mid \Xi \qquad \Theta \vdash v : \tau_2 \mid \Xi}{\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi} \qquad \frac{v :: P}{\Theta \vdash v : \top_P \mid \Xi} \qquad \frac{\Theta \vdash v : \tau \mid \Xi \quad \tau \preceq_P \tau'}{\Theta \vdash v : \tau' \mid \Xi}$$

$$\frac{\Theta \mid e : \theta_1 \vdash \Xi \qquad \Theta \mid e : \theta_2 \vdash \Xi}{\Theta \mid e : \theta_1 \vee \theta_2 \vdash \Xi} \qquad \frac{e :: N}{\Theta \mid e : \bot_N \vdash \Xi} \qquad \frac{\Theta \mid e : \theta \vdash \Xi \quad \theta' \preceq_N \theta}{\Theta \mid e : \theta' \vdash \Xi}$$

The conditions of $\top_P$ and $\bot_N$ rules ensures that $\Theta \vdash v : \tau \mid \Xi$ and $v :: P$ implies $\tau :: P$. Since $\tau_1 \wedge \tau_2 \preceq_P \tau_1$ and $\theta_1 \preceq_N \theta_1 \vee \theta_2$, the elimination rules are admissible:

$$\frac{\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi}{\Theta \vdash v : \tau_1 \mid \Xi} \qquad \frac{\Theta \mid e : \theta_1 \vee \theta_2 \vdash \Xi}{\Theta \mid e : \theta_1 \vdash \Xi}$$

The subtyping rule for type environments is also admissible.

Lemma 2. *Assume that $\Theta \vdash v : \tau \mid \Xi$ and $\Theta' \preceq \Theta$ and $\Xi \preceq \Xi'$. Then $\Theta' \vdash v : \tau \mid \Xi'$ is derivable. The similar statement holds for co-terms and commands.*

Proof. Easy induction on the structure of the derivation. □

By the previous lemma, one can introduce intersection to the positive type environment and union to the negative type environment:

$$\frac{\Theta, x : \sigma \vdash v : \tau \mid \Xi}{\Theta, x : \sigma \wedge \sigma' \vdash v : \tau \mid \Xi} \qquad \frac{\Theta \vdash v : \tau \mid \alpha : \theta, \Xi}{\Theta \vdash v : \tau \mid \alpha : \theta \vee \theta', \Xi}$$

We *cannot* derive $\Theta \vdash v : \tau \vee \sigma \mid \Xi$ from $\Theta \vdash v : \tau \mid \Xi$ since union is not defined on positive types. Instead $\Theta \vdash v : \downarrow(\theta \vee \delta) \mid \Xi$ is derivable from $\Theta \vdash v : \downarrow\theta \mid \Xi$.

3.3 Subject Reduction

Subject Reduction (Proposition 4) can be proved by the standard technique using Substitution Lemma and Inversion (Lemmas 3 and 4). The proof of Inversion needs some case because of the subtyping rule.

Lemma 3 (Substitution). *Assume $\Theta \vdash v : \tau \mid \Xi$, which refines $\Gamma \vdash v :: P \mid \Delta$.*

- *If* $\Theta, x : \tau \vdash v' : \sigma \mid \Xi$, *then* $\Theta \vdash v'[v/x] : \sigma \mid \Xi$.
- *If* $\Theta, x : \tau \mid e : \theta \vdash \Xi$, *then* $\Theta \mid e[v/x] : \theta \mid \Xi$.
- *If* $c : (\Theta, x : \tau \vdash \Xi)$, *then* $c[v/x] : (\Theta \vdash \Xi)$.

The similar statements hold for substitution of co-terms.

Lemma 4 (Inversion).

1. *If* $\Theta \vdash \lambda x.v : \downarrow\theta \mid \Xi$, *there are* σ *and* δ *s.t.* $\Theta, x : \sigma \vdash v : \downarrow\delta \mid \Xi$ *and* $\sigma \to \delta \preceq \theta$.
2. *If* $\Theta \vdash \mu\alpha.v : \downarrow\theta \mid \Xi$, *then* $c : (\Theta \vdash \alpha : \theta, \Xi)$.
3. *If* $\Theta \vdash e \triangleleft v : \sigma \leftarrow \theta \mid \Xi$, *then* $\Theta \vdash v : \sigma \mid \Xi$ *and* $\Theta \mid e : \theta \vdash \Xi$.

The similar statements hold for co-terms.

Proposition 4 (Subject Reduction). *If* $c : (\Theta \vdash \Xi)$ *and* $c \longrightarrow c'$, *then* $c' : (\Theta \vdash \Xi)$. *The similar statements hold for terms and co-terms.*

Proof (Sketch). We prove one of the base cases of which the proof is most complicated. Consider the case that $c = \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ and $c' = \langle (v_1[v_2/x]) \downarrow e \rangle$. By the assumption, $\langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle : (\Theta \vdash \Xi)$. Then $\Theta \vdash \lambda x.v_1 : \downarrow\theta \mid \Xi$ and $\Theta \mid v_2 \triangleright e : \theta \vdash \Xi$. By Lemma 4, one has $\Theta, x : \sigma \vdash v_1 : \downarrow\delta \mid \Xi$ and $\sigma \to \delta \preceq \theta$ for some σ and δ. By the subtyping rule, we have $\Theta \mid v_2 \triangleright e : \sigma \to \delta \vdash \Xi$. Again, by Lemma 4, $\Theta \vdash v_2 : \sigma \mid \Xi$ and $\Theta \mid e : \delta \vdash \Xi$. By Substitution Lemma (Lemma 3), $\Theta \vdash v_1[v_2/x] : \downarrow\delta \mid \Xi$. Hence $\langle v_1[v_2/x] \downarrow e \rangle : (\Theta \vdash \Xi)$. □

3.4 Subject Expansion

The proof of Subject Expansion is rather straightforward.

Lemma 5 (De-substitution). *Suppose that* $\Gamma \vdash v :: P \mid \Delta$, $\Theta :: \Gamma$ *and* $\Xi :: \Delta$. *Let* x *be a variable of sort* P.

- *If* $\Theta \vdash v'[v/x] : \sigma \mid \Xi$, *then* $\Theta, x : \tau \vdash v' : \sigma \mid \Xi$ *and* $\Theta \vdash v : \tau \mid \Xi$ *for some* $\tau :: P$.
- *If* $\Theta \mid e[v/x] : \theta \vdash \Xi$, *then* $\Theta, x : \tau \mid e : \theta \vdash \Xi$ *and* $\Theta \vdash v : \tau \mid \Xi$ *for some* $\tau :: P$.
- *If* $(c[v/x]) : (\Theta \vdash \Xi)$, *then* $c : (\Theta, x : \tau \vdash \Xi)$ *and* $\Theta \vdash v : \tau \mid \Xi$ *for some* $\tau :: P$.

The similar statements hold for substitution of co-terms.

Proposition 5 (Subject Expansion). *If* $c \longrightarrow c'$ *and* $c' : (\Theta \vdash \Xi)$, *then* $c : (\Theta \vdash \Xi)$. *The similar statements hold for terms and co-terms.*

Proof (Sketch). We prove one of the base cases of which the proof is most complicated. Consider the case that Case $c = \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ and $c' = \langle (v_1[v_2/x]) \downarrow e \rangle$. By the assumption, $\langle (v_1[v_2/x]) \downarrow e \rangle : (\Theta \vdash \Xi)$. Then $\Theta \vdash v_1[v_2/x] : \downarrow\theta \mid \Xi$ and $\Theta \mid e : \theta \vdash \Xi$ for some θ. By De-substitution Lemma (Lemma 5), we have σ such that $\Theta, x : \sigma \vdash v_1 : \downarrow\theta \mid \Xi$ and $\Theta \vdash v_2 : \sigma \mid \Xi$. Hence we have $\Theta \vdash \lambda x.v_1 : \downarrow(\sigma \to \theta) \mid \Xi$ and $\Theta \mid v_2 \triangleright e : \sigma \to \theta \vdash \Xi$. Now $\langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle : (\Theta \vdash \Xi)$. □

Corollary 2. *Validity of a judgement is preserved by* $=_\beta$ *on subjects.*

3.5 Normalisation

Let us consider the $\{\top, \bot\}$-free subsystem of the intersection and union type system, in which one cannot use $\top_P$ nor $\bot_N$ at all. We write $\Gamma \Vdash v : \tau \parallel \Xi$ to mean $\Gamma \vdash v : \tau \mid \Xi$ is derivable in the subsystem. Like an intersection type system for the λ-calculus [7], this subsystem completely characterises strongly normalising expressions.

Here we prove soundness, appealing to the connection to the calculus $\lambda^{\to\times}$ (see Section 2.6). The intersection type system for $\bar{\lambda}\mu\tilde{\mu}_P$ is also closely related to that for $\lambda^{\to\times}$ by design.[3] Because all connectives except for intersection and union are in the sort system as well, what we shall define is the translation of intersection and union, which is given by:

$$(\tau_1 \wedge \tau_2)^* := \tau_1^* \wedge \tau_2^* \qquad (\theta_1 \vee \theta_2)^* := \theta_1^* \wedge \theta_2^*.$$

Note that the union on negatives is understood as an intersection.

Lemma 6. *In the respective intersection (and union) type systems, we have*

$$\left\{\begin{array}{c}\Theta \Vdash v : \tau \parallel \Xi \\ \Theta \parallel e : \theta \Vdash \Xi \\ c : (\Theta \Vdash \Xi)\end{array}\right\} \text{ in } \bar{\lambda}\mu\tilde{\mu}_P \implies \left\{\begin{array}{c}\Theta^*, \Xi^* \Vdash v^* : \tau^* \\ \Theta^*, \Xi^* \Vdash e^* : \theta^* \\ \Theta^*, \Xi^* \Vdash c^* : O.\end{array}\right\} \text{ in } \lambda^{\to\times}.$$

Recall that the translation $(-)^*$ preserves and reflects the reduction relation (Lemma 1), and thus t is strongly normalising if and only if so is t^* (Corollary 1). Hence soundness of our subsystem follows from that for the intersection type system for the $\lambda^{\to\times}$-calculus with recursive sorts.

Theorem 1. *An expression typable in the subsystem is strongly normalising.*

3.6 Completeness

We prove completeness of the $\{\top, \bot\}$-free subsystem with respect to strong normalisation. Recall that our intersection and union type system is parameterised by finite sets of recursive sorts as well as families of atomic types $\{\mathcal{X}_P\}_P$ and $\{\mathcal{X}_N\}_N$. Obviously, if $\mathcal{X}_P = \mathcal{X}_N = \emptyset$ for every P and N, completeness fails, since there is no $\{\bot, \top\}$-free (raw) type. To avoid this, here we assume the following.

Every sort has at least one $\{\top, \bot\}$-free refinement type. $(\star)$

Lemma 7. *Every well-sorted normal form is typable in the subsystem.*

Theorem 2 (Completeness). *A strongly normalising (well-sorted) expression is typable in the $\{\top, \bot\}$-free subsystem.*

[3] The intersection type system for the λ-calculus with pairs that we consider is the straightforward extension of the simple type system $\lambda^{\to\times}$.

Proof (Sketch). We prove the claim by induction on the length of the longest reduction sequence starting from the expression. If the expression is in normal form, we use Lemma 7. For reducible expressions, we prove the claim by induction on the structure of the derivation of the reduction relations. For example, consider a base case $\langle(\lambda x.v_1) \downarrow (v_2 \triangleright e)\rangle \longrightarrow \langle(v_1[v_2/x]) \downarrow e\rangle$. By the induction hypothesis, $\langle(v_1[v_2/x]) \downarrow e\rangle$ is typable in the $\{\top, \bot\}$-free subsystem. Assume that $\langle(v_1[v_2/x]) \downarrow e\rangle : (\Theta \Vdash \Xi)$, and hence $\Theta \Vdash v_1[v_2/x] : \downarrow\theta \parallel \Xi$ and $\Theta \parallel e : \theta \Vdash \Xi$ for some θ. There are two cases. If x occurs freely in v_1, then one can prove De-substitution Lemma for the subsystem, which completes the proof for this case. If x does not occur in v_1, the naïve adaptation of De-substitution Lemma does not work, since the judgement obtained by the lemma is $\Theta \vdash v_2 : \top_P \mid \Xi$, which is not derivable in the subsystem. However, in this case, $\Theta, x : \tau \Vdash v_1 : \theta \parallel \Xi$ for every $\{\top, \bot\}$-free type τ (which refines the sort of x). By the induction hypothesis, one has $\Theta' \Vdash v_2 : \tau \parallel \Xi'$ in the $\{\top, \bot\}$-free subsystem for some Θ', Ξ', and τ'. Then we have $\Theta' \wedge \Theta \Vdash \lambda x.v_1 : \downarrow(\tau \to \theta) \parallel \Xi \vee \Xi'$ and $\Theta' \wedge \Theta \parallel v_2 \triangleright e : \tau \to \theta \Vdash \Xi \vee \Xi'$, and thus $\langle(\lambda x.v_1) \downarrow (v_2 \triangleright e)\rangle$ is typable in the subsystem. □

4 Translations to the Polarised Calculus

We shall see how to translate call-by-value and call-by-name $\bar{\lambda}\mu\tilde{\mu}$ into the polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$, using the polarity to encode evaluation strategies.

We introduce a convenient abbreviation that will be used in this section: $\Downarrow v := \mu\alpha.\langle v \Uparrow \alpha\rangle$. If $\vdash v : P$, then $\vdash \Downarrow v : \downarrow\uparrow P$ (hence the notation). Similarly $\Uparrow e$ is defined as $\mu x.\langle x \downarrow e\rangle$. We have $\langle \Downarrow v \downarrow e\rangle \longrightarrow \langle v \Uparrow e\rangle$ and $\langle v \Uparrow \Uparrow e\rangle \longrightarrow \langle v \downarrow e\rangle$. Do not confuse them with "η-expansion" $v \mapsto \mu\alpha.\langle v \downarrow \alpha\rangle$, which does not change the meaning including the sort. (If $\vdash v :: \downarrow N$, then $\vdash \mu\alpha.\langle v \downarrow \alpha\rangle :: \downarrow N$.) The double-shifting $\Downarrow(-)$ corresponds to $\lambda k.k(-)$ (i.e. passing a value to the continuation) in the λ-calculus.

4.1 Embedding of CBV $\bar{\lambda}\mu\tilde{\mu}$

Given an expression t of $\bar{\lambda}\mu\tilde{\mu}$, we define an expression t^v of $\bar{\lambda}\mu\tilde{\mu}_P$ as follows.

(Values)	$\Phi(x) = x$	$\Phi(\lambda x.v) = \lambda x.v^v$
(Terms)	$V^v = \Downarrow\Phi(V)$	$(\mu\alpha.c)^v = \mu\alpha.c^v$
(Co-terms)	$\alpha^v = \alpha$	$(\tilde{\mu}x.c)^v = \mu x.c^v$
	$(v \cdot e)^v = \mu f.\langle v^v \downarrow \mu y.\langle f \downarrow y \triangleright e^v\rangle\rangle$	
(Commands)	$\langle v \mid e\rangle^v = \langle v^v \downarrow e^v\rangle$.	

In the translation of cuts of the call-by-value calculus, the term side has the priority. We focus on the co-term side when the term side is evaluated to a value: $\langle V^v \downarrow e^v\rangle = \langle \Downarrow\Phi(V) \downarrow e^v\rangle \longrightarrow \langle \Phi(V) \Uparrow e^v\rangle$.

The translation preserves operational semantics in the following sense.

Proposition 6. *If $t \longrightarrow u$, then $t^v \longrightarrow^+ u^v$.*

Corollary 3. *$t =^{\mathrm{cbv}}_{\beta} u$ implies $t^v =_\beta u^v$. If t^v is SN, then so is t in call-by-value.*

Although we do not assume well-sortedness of the source expression, the image of the translation is always well-sorted. Let $\mathtt{V}$ be a positive sort variable with the equation $\mathtt{V} = \downarrow(\mathtt{V} \to (\uparrow\mathtt{V}))$. Then one has $\overrightarrow{x :: \mathtt{V}} \vdash \Phi(V) :: \mathtt{V} \mid \overrightarrow{\alpha :: \uparrow\mathtt{V}}$ and

$$\overrightarrow{x :: \mathtt{V}} \vdash v^n :: \downarrow\uparrow\mathtt{V} \mid \overrightarrow{\alpha :: \uparrow\mathtt{V}}, \qquad \overrightarrow{x :: \mathtt{V}} \mid e^n :: \uparrow\mathtt{V} \vdash \overrightarrow{\alpha :: \uparrow\mathtt{V}}, \quad \text{and} \quad c^n :: (\overrightarrow{x :: \mathtt{V}} \vdash \overrightarrow{\alpha :: \uparrow\mathtt{V}}).$$

4.2 CBN $\bar{\lambda}\mu\tilde{\mu}$

Given an expression t of $\bar{\lambda}\mu\tilde{\mu}$, we define an expression t^n of $\bar{\lambda}\mu\tilde{\mu}_P$ as follows.

(Terms)	$x^n = x$	$(\lambda x.v)^n = \lambda x.(\Downarrow v^n)$	$(\mu\alpha.c)^n = \mu\alpha.c^n$
(Co-values)	$\Psi(\alpha) = \alpha$	$\Psi(v \cdot e) = v^n \triangleright e^n$	
(Co-terms)	$E^n = \Downarrow\Psi(E)$	$(\tilde{\mu}x.c)^n = \mu x.c^n$	
(Commands)	$\langle v \mid e\rangle^n = \langle v^n \Rsh e^n\rangle$.		

In the interpretation of cuts of the call-by-name calculus, the co-term side has priority. When the co-term side is a co-value E, then we focus on the term side by the rule $\langle v^n \Rsh \Downarrow\Phi(E)\rangle \longrightarrow \langle v^n \Lsh \Phi(E)\rangle$.

The translation preserves operational semantics in the following sense.

Proposition 7. *$t =_\beta^{\mathrm{cbn}} u$ implies $t^n =_\beta u^n$. If t^n is SN, so is t in call-by-name.*

Unfortunately it is not the case that $t \longrightarrow u$ in $\bar{\lambda}\mu\tilde{\mu}$ implies $t^n \longrightarrow^* u^n$ in $\bar{\lambda}\mu\tilde{\mu}_P$. This holds if we choose $\langle \lambda x.v \mid v' \cdot e\rangle \longrightarrow \langle v[v'/x] \mid e\rangle$ as the β-rule.

The image of this translation is well-sorted. Let $\mathtt{N}$ be a negative sort variable in $\bar{\lambda}\mu\tilde{\mu}_P$ equipped with $\mathtt{N} = (\downarrow\mathtt{N}) \to (\uparrow\downarrow\mathtt{N})$. Then

$$\overrightarrow{x :: \downarrow\mathtt{N}} \vdash v^n :: \downarrow\mathtt{N} \mid \overrightarrow{\alpha :: \mathtt{N}}, \qquad \overrightarrow{x :: \downarrow\mathtt{N}} \mid e^n :: \uparrow\downarrow\mathtt{N} \vdash \overrightarrow{\alpha :: \mathtt{N}}, \quad \text{and} \quad c^n :: (\overrightarrow{x :: \downarrow\mathtt{N}} \vdash \overrightarrow{\alpha :: \mathtt{N}}).$$

5 Intersection and Union Types for Untyped CBV $\bar{\lambda}\mu\tilde{\mu}$

This section develops a sound and complete type assignment system for the untyped version of call-by-value $\bar{\lambda}\mu\tilde{\mu}$ as an application of our development. The types for an expression is preserved by β-equivalence, i.e. the type system enjoys both Subject Reduction and Subject Expansion. Furthermore the $\{\top, \bot\}$-free subsystem completely characterises strongly normalising expressions, as the traditional intersection type systems do for the untyped λ-calculus.

The type assignment system is induced from the translation of CBV $\bar{\lambda}\mu\tilde{\mu}$ to $\bar{\lambda}\mu\tilde{\mu}_P$. Given an expression t of $\bar{\lambda}\mu\tilde{\mu}$, the types for t^v is preserved by β-equivalence, because the translation preserves reduction (Proposition 6) and the type system for $\bar{\lambda}\mu\tilde{\mu}_P$ enjoys Subject Reduction and Expansion (Propositions 4 and 5). Now the goal is to develop a type system for $\bar{\lambda}\mu\tilde{\mu}$ that is equivalent to typing after the translation.

We first decide the syntax of types by using the refinement relation. Recall that, given a $\bar{\lambda}\mu\tilde{\mu}$-expression t, the result t^v of the translation is well-sorted by using the recursive sort defined by $\mathtt{V} = \downarrow(\mathtt{V} \to (\uparrow\mathtt{V}))$. The sorts we need to

show well-sortedness of t^v are $\mathtt{V}$, $\uparrow\mathtt{V}$, $\mathtt{V} \to (\uparrow\mathtt{V})$, and $\downarrow\uparrow\mathtt{V}$. This suggests that the syntax of types for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$ should have four classes. We define *the intersection and union types for the call-by-value* $\bar{\lambda}\mu\tilde{\mu}$ by the following grammar:

$$\psi ::= a \mid \nu \to \varphi \mid \psi \vee \psi \mid \bot_{\mathtt{V}\to(\uparrow\mathtt{V})} \qquad \varphi ::= \uparrow\nu \mid \varphi \vee \varphi \mid \bot_{\uparrow\mathtt{V}}$$
$$\nu ::= \downarrow\psi \mid \nu \wedge \nu \mid \top_{\mathtt{V}} \qquad \varrho ::= \downarrow\varphi \mid \rho \wedge \rho \mid \top_{\downarrow\uparrow\mathtt{V}}.$$

It is easy to see that the above syntax properly defines refinement types of $\mathtt{V} \to (\uparrow\mathtt{V})$, $\mathtt{V}$, $\uparrow\mathtt{V}$, and $\downarrow\uparrow\mathtt{V}$, respectively.

A positive type environment Θ is a finite set of type binding of the form $x : \nu$, since the sort environment is $\overrightarrow{x :: \mathtt{V}}$. By the same reason, a negative type environment Ξ is a set of type binding of the form $\alpha : \varphi$.

We use $\vdash^{\mathrm{cbv}}$ for judgements of the type system for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$. We have two kinds of judgements for terms. The first one is $\Theta \vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$ for values. The second one is $\Theta \vdash^{\mathrm{cbv}} v : \varrho \mid \Xi$ for terms. Even if v is a value, we distinguish the two judgements.

The typing rules are listed below. To save the space, we omit the intersection and union introduction rules and the subtyping rules.

$$\frac{}{\Theta, x : \nu \vdash^{\mathrm{cbv}} x : \nu \fatsemi \Xi} \qquad \frac{\Theta \vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi}{\Theta \vdash^{\mathrm{cbv}} V : \downarrow\uparrow\nu \mid \Xi} \qquad \frac{\Theta, x : \nu \vdash^{\mathrm{cbv}} v : \downarrow\varphi \mid \Xi}{\Theta \vdash^{\mathrm{cbv}} \lambda x.v : \downarrow(\nu \to \varphi) \fatsemi \Xi}$$

$$\frac{c : (\Theta \vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)}{\Theta \vdash^{\mathrm{cbv}} \mu\alpha.c : \downarrow\varphi \mid \Xi} \qquad \frac{}{\Theta \mid \alpha : \varphi \vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi} \qquad \frac{c : (\Theta, x : \nu \vdash^{\mathrm{cbv}} \Xi)}{\Theta \mid \tilde{\mu}x.c : \uparrow\nu \vdash^{\mathrm{cbv}} \Xi}$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} v : \downarrow(\bigvee_{i\in I}(\uparrow\nu_i)) \mid \Xi \quad \Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi}{\Theta \mid v \cdot e : \uparrow(\bigwedge_{i\in I}\downarrow(\nu_i \to \varphi)) \vdash^{\mathrm{cbv}} \Xi} \qquad \frac{\Theta \vdash^{\mathrm{cbv}} v : \downarrow\varphi \mid \Xi \quad \Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi}{\langle v \mid e \rangle : (\Theta \vdash^{\mathrm{cbv}} \Xi)}$$

Lemma 8. *We have*

$$\Theta \vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi \iff \Theta \vdash \Phi(V) : \nu \mid \Xi$$
$$\Theta \vdash^{\mathrm{cbv}} v : \varrho \mid \Xi \iff \Theta \vdash v^v : \varrho \mid \Xi$$
$$\Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi \iff \Theta \mid e^v : \varphi \vdash \Xi$$
$$c : (\Theta \vdash^{\mathrm{cbv}} \Xi) \iff c^v : (\Theta \vdash \Xi).$$

The similar statements hold for the $\{\top, \bot\}$*-free subsystems.*

By Lemma 8 and Corollary 3, the set of valid type judgements for a $\bar{\lambda}\mu\tilde{\mu}$-expression is preserved by call-by-value β-reduction.

Theorem 3. *Validity of a type judgement is preserved by* $=^{\mathrm{cbv}}_{\beta}$ *on subjects.*

Another consequence of Lemma 8 and Corollary 3 is that a $\bar{\lambda}\mu\tilde{\mu}$-expression typable in the $\{\bot, \top\}$-free subsystem is strongly normalising with respect to the call-by-value reduction. If t is typable in the $\{\top, \bot\}$-free subsystem, then t^v is typable in the subsystem and thus t^v is strongly normalising by Theorem 1, which implies that t is strongly normalising in the call-by-value evaluation strategy. Completeness can be proved directly as in the proof of Theorem 2.

Theorem 4. *A $\bar{\lambda}\mu\tilde{\mu}$-expression is typable in the $\{\bot, \top\}$-free subsystem if and only if it is strongly normalising with respect to the call-by-value reduction.*

6 Related work

The connection between polarity and intersection and union types has been pointed out by Zeilberger [30]. He aimed to clarify without trial and error valid typing and subtyping rules, which heavily depends on computational effect and the evaluation strategy of the calculus (see, e.g. [10,13]). His approach is based on a refinement of a focusing proof, closely related to the notion of polarity. In his focused calculus, a value has a sort different from a term, and the unrestricted intersection introduction rule is admissible for refinements of a sort for values but not for refinements of a sort for terms.[4]

We found a complementary relationship between his work and our work. Our starting point is a well-know calculus $\bar{\lambda}\mu\tilde{\mu}$ [9], and the notion of polarities is introduced by a simple change, whereas his calculus looks quite different from the conventional calculi. We showed that intersection interacts well with positives and unions with negatives, and this conclusion seems different from [30]. We proved properties that one expects for an intersection (and union) type system, including a completeness result. Usefulness of our idea is demonstrated by deriving an intersection and union type system for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$.

Intersection and union types for $\bar{\lambda}\mu\tilde{\mu}$ and related calculi Dougherty et al. [11] proposed an intersection and union type system $\mathcal{M}^{\cap\cup}$ for the untyped $\bar{\lambda}\mu\tilde{\mu}$ (that they called Gemini) and claimed Subject Reduction of the system (as well as strong normalisation of well-typed expressions). However it contains an error (see Section 4.2 and related work of [12]). Then they gave another type system $\mathcal{M}^{\cap}$ based on intersection and "negation", and stated that it enjoys Subject Reduction and Expansion [12]. However van Bakel [2] constructed counterexamples of both properties (see Section 8 of [2]).

Van Bakel [2] investigated the failure of Subject Reduction and Expansion of (a slight variant of) $\mathcal{M}^{\cap\cup}$ and introduced several variants. The system $\mathcal{M}^{c}$ has additional typing rules and satisfies Subject Expansion but not Subject Reduction. He also developed type systems for call-by-name and call-by-value $\bar{\lambda}\mu\tilde{\mu}$, which enjoy Subject Reduction but not Subject Expansion.

To the best of our knowledge, development of an intersection (and union) type system that satisfies Subject Reduction and Expansion and characterises strongly normalising expression for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$ had been open.

In those systems, a left-hand environment Θ (corresponding to a positive type environment in this paper) is limited to intersection types and a right-hand type environment (corresponding to a negative type environment) is to union types (see, e.g., [11] for the reason of this design). The exception is $\mathcal{M}^{c}$ in which a positive type environment can have union type binding $x : \tau \vee \sigma$ but at the price

[4] Unfortunately his terminology is opposite to ours: in [30], a sort refines a type.

of loss of Subject Reduction. Our type system does not allow $\Theta, x : \tau \vee \sigma$ to be a positive type environment, since $\vee$ is not allowed for positive types. However, in our system, a positive type environment can have $x : \downarrow(\theta \vee \delta)$, a shifted union type. We believe that this explicit shifting plays a crucial rôle.

Van Bakel et al. [4] developed an intersection type system for $\lambda\mu$ whose syntax is induced from the domain equation for $\lambda\mu$-model. Their approach might be somewhat parallel to our approach, in which the syntax is induced by the sort equation about `V`. Other intersection (and union) type systems for the $\lambda\mu$-calculus can be found in [3,21]. We are not sure if those type systems are derivable by our method. Kikuchi and Sakurai [16] gave a translation from [4] to [21] that somewhat resembles to the translation in this paper from the intersection type system for $\bar{\lambda}\mu\tilde{\mu}_P$ to that for $\lambda^{\to\times}$.

Polarised/focalised calculi The notions of polarity [15] and focusing [1] have been extensively studied in the field of logic and computation. Our polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$ is inspired by work of many researchers including Andreoli [1], Girard [15], Laurent [19,20,22], Melliés [24], Melliés and Selinger [25], Munch-Maccagnoni [27], Zeilberger [29].

Curien and Munch-Maccagnoni [8] and Munch-Maccagnoni [26] gave focalised variants of $\bar{\lambda}\mu\tilde{\mu}$. These calculi appear different from ours, at least superficially. First positive and negative terms of their calculi have different syntax, whereas our calculus is symmetric. Second the notion of values plays an important role in their calculus, whereas our calculus does not have the notion of values (nor co-values). Third their calculi have a pair constructor $(\cdot, \cdot)$ and variant constructors *inl* and *inr*, whereas our calculus does not. A more mature comparison is left for future work.

7 Conclusion and Future Work

We introduced the polarised calculus $\bar{\lambda}\mu\tilde{\mu}_P$, to which the simple intersection and union extension works quite well. The intersection and union type system is a refinement type system, in which intersection is restricted to refinements of positive sorts and union to those of negative sorts. Using the polarised calculus as an intermediate language, we developed an intersection and union type system for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$, which enjoys both Subject Reduction and Expansion and characterises strongly normalising expressions.

The intersection type system provides us with a characterisation of weakly normalising expressions, although we do not have enough space to discuss it. Our approach is applicable to the call-by-name $\bar{\lambda}\mu\tilde{\mu}$ as well.

We are not sure if the refinement type system of this paper is the simplest one for $\bar{\lambda}\mu\tilde{\mu}_P$. We suspect that use of intersection and union can be more restrictive without losing any desired properties. It is also interesting to extend the calculus by pairs and variants. This would clarify the connection to the focalised calculi [8,26,29] and to LC [15].

References

1. Andreoli, J.M.: Logic programming with focusing proofs in linear logic. Journal of Logic and Computation 2(3), 297–347 (1992)
2. van Bakel, S.: Completeness and partial soundness results for intersection and union typing for $\bar{\lambda}\mu\tilde{\mu}$. Annals of Pure and Applied Logic 161(11), 1400–1430 (2010)
3. van Bakel, S.: Sound and complete typing for lambda-mu. Electronic Proceedings in Theoretical Computer Science 45(ITRS 2010), 31–44 (2011)
4. van Bakel, S., Barbanera, F., de'Liguoro, U.: A filter model for the $\lambda\mu$-calculus. In: TLCA 2011. Lecture Notes in Computer Science, vol. 6690, pp. 213–228. Springer Berlin Heidelberg (2011)
5. Barbanera, F., Dezani-Ciancaglini, M., de'Liguoro, U.: Intersection and union types: Syntax and semantics. Information and Computation 119(2), 202–230 (1995)
6. Barbanera, F., Dezani-Ciancaglini, M.: Intersection and union types. Theoretical Aspects of Computer Software, Volume 526 of LNCS (i), 561–674 (1991)
7. Coppo, M., Dezani-Ciancaglini, M., Venneri, B.: Functional characters of solvable terms. Mathematical Logic Quarterly 27, 45–58 (1981)
8. Curien, P.L., Munch-Maccagnoni, G.: The duality of computation under focus. In: Theoretical Computer Science. pp. 165–181 (2010)
9. Curien, P.L., Herbelin, H.: The duality of computation. In: Proceedings of the fifth ACM SIGPLAN international conference on Functional Programming (ICFP 2000). pp. 233–243 (2000)
10. Davies, R., Pfenning, F.: Intersection types and computational effects. ICFP 2000 pp. 198–208 (2000)
11. Dougherty, D.J., Ghilezan, S., Lescanne, P.: Intersection and union types in the $\bar{\lambda}\mu\tilde{\mu}$-calculus. Electronic Notes in Theoretical Computer Science 136, 153–172 (2005)
12. Dougherty, D.J., Ghilezan, S., Lescanne, P.: Characterizing strong normalization in the Curien-Herbelin symmetric lambda calculus: Extending the Coppo-Dezani heritage. Theoretical Computer Science 398(1-3), 114–128 (2008)
13. Dunfield, J., Pfenning, F.: Type assignment for intersections and unions in call-by-value languages. In: FoSSaCS 2003. pp. 250–266 (2003)
14. Freeman, T., Pfenning, F.: Refinement types for ML. In: PLDI 1991. vol. 26, pp. 268–277 (1991)
15. Girard, J.Y.: A new constructive logic: classical logic. Mathematical Structures in Computer Science 1, 255–296 (1991)
16. Kikuchi, K., Sakurai, T.: A translation of intersection and union types for the $\lambda\mu$-calculus. In: Proceedings of the 12th Asian symposium on Programming Languages and Systems (APLAS 2014). vol. 14, pp. 120–139 (2014)
17. Kobayashi, N.: Model checking higher-order programs. Journal of the ACM 60(3), 1–62 (2013)
18. Kobayashi, N., Ong, C.H.L.: A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In: LICS 2009. pp. 179–188. IEEE Computer Society (2009)
19. Laurent, O.: Polarized proof-nets: proof-nets for LC. In: Typed Lambda Calculi and Applications. Lecture Notes in Computer Science, vol. 1581, pp. 213–227. Springer Berlin Heidelberg (1999)
20. Laurent, O.: Polarized proof-nets and $\lambda\mu$-calculus. Theoretical Computer Science 290(1), 161–188 (2003)

21. Laurent, O.: On the denotational semantics of the untyped lambda-mu calculus (2004), unpublished note
22. Laurent, O.: Polarized games. Annals of Pure and Applied Logic 130(1-3), 79–123 (2004)
23. MacQueen, D., Plotkin, G., Sethi, R.: An ideal model for recursive polymorphic types. Information and Control 71(1-2), 95–130 (1986)
24. Melliès, P.a.: Dialogue categories and chiralities, unpublished manuscript (available online)
25. Melliés, P.A., Selinger, P.: Games are continuation models! (2001), talk at Full Completeness and Full Abstraction, Satellite workshop of LICS 2001 (the slides are available online)
26. Munch-maccagnoni, G.: Focalisation and classical realisability. In: Computer Science Logic. pp. 409–423 (2009)
27. Munch-Maccagnoni, G.: Models of a non-associative composition. In: FoSSaCS 2014. pp. 396–410 (2014)
28. Thielecke, H.: Categorical Structure of Continuation Passing Style. Ph.D. thesis, University of Edinburgh (1997)
29. Zeilberger, N.: On the unity of duality. Annals of Pure and Applied Logic 153(1-3), 66–96 (2008)
30. Zeilberger, N.: Refinement types and computational duality. In: Proceedings of the 3rd workshop on Programming Languages meets Program Verification (PLPV 09). pp. 15–26 (2009)

A Precise Definitions of Calculi

This section gives the precise definitions of the systems in this paper, some of which are omitted in the body.

A.1 Calculus $\bar{\lambda}\mu\tilde{\mu}_P$

Terms Assume infinite sets of *positive variables*, ranged over by x, y and z, and *negative variables*, ranged over by α, β and γ. The syntax of expressions is given by:

$$\begin{array}{ll} \textit{(Terms)} & v ::= x \mid \lambda x.v \mid \mu\alpha.c \mid v \triangleleft e \\ \textit{(Commands)} & c ::= \langle v \downarrow e \rangle \mid \langle v \Rsh e \rangle \\ \textit{(Co-terms)} & e ::= \alpha \mid \lambda\alpha.e \mid \mu x.c \mid v \triangleright e. \end{array}$$

We use a metavariable t for expressions.

Reduction We write $t[v/x]$ for the capture-avoiding substitution of v for x in t, and $t[e/\alpha]$ for the substitution of e for α in t. The calculus has 4 reduction rules, namely, positive/negative λ/μ reductions. The rules are listed below:

$$\begin{array}{ll} \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle \longrightarrow \langle (v_1[v_2/x]) \downarrow e \rangle & \langle (\mu\alpha.c) \downarrow e \rangle \longrightarrow c[e/\alpha] \\ \langle (v \triangleleft e_1) \Rsh (\lambda\alpha.e_2) \rangle \longrightarrow \langle v \Rsh (e_2[e_1/\alpha]) \rangle & \langle v \Rsh (\mu x.c) \rangle \longrightarrow c[v/x] \end{array}$$

The reduction is supposed to be full, i.e. any redex at any position can be reduced. We write $\longrightarrow^*$ for the reflexive and transitive closure of $\longrightarrow$, and $=_\beta$ for the least equivalence containing $\longrightarrow$. Some expression gets stuck, e.g. $\langle (\lambda x.v_1) \Rsh (v_2 \triangleright e) \rangle$ and $\langle (\lambda x.v_1) \downarrow (\mu y.c) \rangle$.

A.2 Sort system for $\bar{\lambda}\mu\tilde{\mu}_P$

Let $\mathcal{P}$ and $\mathcal{N}$ be finite sets of positive and negative sort variables, respectively. We write $\mathcal{E}$ for the associated equations. For each $\mathtt{P} \in \mathcal{P}$, we have a unique P such that $(\mathtt{P} = P) \in \mathcal{E}$, where P is not a sort variable.

Sorts The syntax of *simple polarised sorts* is given by:

$$\begin{array}{ll} \textit{(Positive sorts)} & P, Q \ ::= p \mid \mathtt{P} \mid P \leftarrow N \mid \downarrow N \\ \textit{(Negative sorts)} & N, M ::= n \mid \mathtt{N} \mid P \to N \mid \uparrow P. \end{array}$$

Let $\sim$ be the least relation which satisfies the rules in Fig. 1. It is easy to see that $\sim$ is an equivalence relation on sorts.

$$\frac{(\mathtt{P} = P) \in \mathcal{E}}{\mathtt{P} \sim P}\ (\sim\text{-PV-L}) \qquad \frac{(\mathtt{N} = N) \in \mathcal{E}}{\mathtt{N} \sim N}\ (\sim\text{-NV-L})$$

$$\frac{(\mathtt{P} = P) \in \mathcal{E}}{P \sim \mathtt{P}}\ (\sim\text{-PV-R}) \qquad \frac{(\mathtt{N} = N) \in \mathcal{E}}{N \sim \mathtt{N}}\ (\sim\text{-NV-R})$$

$$\frac{}{p \sim p}\ (\sim\text{-PA}) \qquad \frac{}{n \sim n}\ (\sim\text{-NA})$$

$$\frac{P \sim P' \qquad N \sim N'}{P \leftarrow N \sim P' \leftarrow N'}\ (\sim\text{-PTo}) \qquad \frac{P \sim P' \qquad N \sim N'}{P \to N \sim P' \to N'}\ (\sim\text{-NTo})$$

$$\frac{N \sim N'}{{\downarrow}N \sim {\downarrow}N'}\ (\sim\text{-PShift}) \qquad \frac{P \sim P'}{{\uparrow}P \sim {\uparrow}P'}\ (\sim\text{-NShift})$$

$$\frac{P_1 \sim P_2 \qquad P_2 \sim P_3}{P_1 \sim P_3}\ (\sim\text{-PTrans}) \qquad \frac{N_1 \sim N_2 \qquad N_2 \sim N_3}{N_1 \sim N_3}\ (\sim\text{-NTrans})$$

Fig. 1. Sort equivalence

Sort environments and judgements A *positive sort environment* Γ (resp. *negative sort environment* Δ) is a finite set of sort bindings of the form $x :: P$ (resp. $\alpha :: N$). We use double colon for sort bindings, reserving single colon for type bindings. A sort environment is considered as a set and the order of sort bindings is not significant. We write Γ, Γ' for $\Gamma \cup \Gamma'$.

There are tree kinds of *sort judgements*, which have both positive and negative sort environments and a subject depending on its kind. The first one $\Gamma \vdash v :: P \mid \Delta$ means that the term v has the positive sort P under the environments Γ and Δ. The second one $\Gamma \mid e :: N \vdash \Delta$, which is the dual to the first one, means that the co-term e has the negative sort N. The third one $c :: (\Gamma \vdash \Delta)$ means that the command c is well-sorted.

Sorting rules Figure 2 is the complete list of the sorting rules.

A.3 Refinement intersection and union type system for $\bar{\lambda}\mu\tilde{\mu}_P$

Fix finite sets of positive and negative sort variables as well as associated equations. Expressions in this subsection are implicitly equipped with their sort.

Assume a (possibly empty) set $\mathcal{X}_P$ (resp. $\mathcal{X}_N$) of atomic types for each positive sort P (resp. negative sort N). For simplicity, we assume $\mathcal{X}_P$, $\mathcal{X}_Q$, $\mathcal{X}_M$ and $\mathcal{X}_N$ are pairwise disjoint if $P \neq Q$ and $M \neq N$. We use a_P (resp. a_N) for a metavariable that ranges over $\mathcal{X}_P$ (reps. $\mathcal{X}_N$).

$$\frac{}{\Gamma, x : P \vdash x : P \mid \Delta}\ (\text{S-PVar}) \qquad \frac{}{\Gamma \mid \alpha : N \vdash \alpha : N, \Delta}\ (\text{S-NVar})$$

$$\frac{\Gamma, x : P \vdash v : \downarrow N \mid \Delta}{\Gamma \vdash \lambda x.v : \downarrow(P \to N) \mid \Delta}\ (\text{S-PAbs}) \qquad \frac{\Gamma \mid e : \uparrow P \vdash \alpha : N, \Delta}{\Gamma \mid \lambda\alpha.e : \uparrow(P \leftarrow N) \vdash \Delta}\ (\text{S-NAbs})$$

$$\frac{c : (\Gamma \vdash \alpha : N, \Delta)}{\Gamma \vdash \mu\alpha.c : \downarrow N \mid \Delta}\ (\text{S-PMu}) \qquad \frac{c : (\Gamma, x : P \vdash \Delta)}{\Gamma \mid \mu x.c : \uparrow P \vdash \Delta}\ (\text{S-NMu})$$

$$\frac{\Gamma \vdash v : Q \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta}{\Gamma \vdash v \triangleleft e : Q \leftarrow N \mid \Delta}\ (\text{S-PApp}) \qquad \frac{\Gamma \mid e : N \vdash \Delta \qquad \Gamma \vdash v : P \mid \Delta}{\Gamma \mid v \triangleright e : P \to N \vdash \Delta}\ (\text{S-NApp})$$

$$\frac{\Gamma \vdash v :: P \mid \Xi \qquad P \sim P'}{\Gamma \vdash v :: P' \mid \Xi}\ (\text{S-P}\sim) \qquad \frac{\Gamma \mid e :: N \vdash \Xi \qquad N \sim N'}{\Gamma \mid e :: N' \vdash \Xi}\ (\text{S-N}\sim)$$

$$\frac{\Gamma \mid e : \uparrow P \vdash \Delta \qquad \Gamma \vdash v : P \mid \Delta}{\langle v \Rsh e \rangle : (\Gamma \vdash \Delta)}\ (\text{S-PCmd}) \qquad \frac{\Gamma \vdash v : \downarrow N \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta}{\langle v \Lsh e \rangle : (\Gamma \vdash \Delta)}\ (\text{S-NCmd})$$

Fig. 2. Sorting rules

$$\frac{a_P \in \mathcal{X}_P}{a_P :: P}\ (\text{R-PA}) \qquad \frac{a_N \in \mathcal{X}_N}{a_N :: N}\ (\text{R-NA})$$

$$\frac{\tau :: P \qquad \theta :: N}{\tau \leftarrow \theta :: P \leftarrow N}\ (\text{R-PTo}) \qquad \frac{\tau :: P \qquad \theta :: N}{\tau \rightarrow \theta :: P \rightarrow N}\ (\text{R-NTo})$$

$$\frac{\theta :: N}{\downarrow\theta :: \downarrow N}\ (\text{R-PShift}) \qquad \frac{\tau :: P}{\uparrow\tau :: \uparrow P}\ (\text{R-NShift})$$

$$\frac{\tau_1, \tau_2 :: P}{\tau_1 \wedge \tau_2 :: P}\ (\text{R-PInt}) \qquad \frac{\theta_1, \theta_2 :: N}{\theta_1 \vee \theta_2 :: N}\ (\text{R-NUni})$$

$$\frac{}{\top_P :: P}\ (\text{R-PTop}) \qquad \frac{}{\bot_N :: N}\ (\text{R-NBot})$$

$$\frac{\tau :: P \qquad P \sim P'}{\tau :: P'}\ (\text{R-P}\sim) \qquad \frac{\theta :: N \qquad N \sim N'}{\theta :: N'}\ (\text{R-N}\sim)$$

Fig. 3. Refinement Relation

Raw types The syntax of raw types is given by

$$\begin{array}{ll} \textit{(Positive raw types)} & \tau, \sigma ::= a_P \mid \tau \leftarrow \theta \mid \downarrow\theta \mid \tau \wedge \tau \mid \top_P \\ \textit{(Negative raw types)} & \theta, \delta ::= a_N \mid \tau \rightarrow \theta \mid \uparrow\tau \mid \theta \vee \theta \mid \bot_N. \end{array}$$

Refinement relation A *type* is a raw type that follows the structure of a sort. To describe this notion formally, we introduce the *refinement relation* defined by Fig. 3.

Subtyping The subtyping relation is defined by the rules in Fig. 4.

Type environments and judgements Let us write Θ for a type environment for positive variables and Ξ for negative variables: $\Theta ::= \cdot \mid \Theta, x : \tau$ and $\Xi ::= \cdot \mid \Xi, \alpha : \theta$. For type environments, we assume that each variable occurs at most once. The refinement relation on environments is defined by point-wise refinement: $\Theta :: \Gamma$ just if $\{x \mid x :: P \in \Gamma\} = \{x \mid x : \tau \in \Theta\}$ and, for all x, $x : \tau \in \Theta$ and $x :: P \in \Delta$ implies $\tau :: P$. The subtyping relation on environments is also defined by point-wise subtyping: given $\Theta_1 :: \Gamma$ and $\Theta_2 :: \Gamma$, we write $\Theta_1 \preceq_\Gamma \Theta_2$ (or simply $\Theta_1 \preceq \Theta_2$) if $x : \tau_1 \in \Theta_1$ and $x : \tau_2 \in \Theta_2$ implies $\tau_1 \preceq_P \tau_2$ (where $x :: P \in \Gamma$).

Recall that expressions and variables are implicitly equipped with their sorts. Hereafter we assume that a type environment respects the sorts. For example, if

$$\frac{\tau :: P}{\tau \preceq_P \tau} \quad (\textsc{Sub-PRefl}) \qquad \frac{\theta :: N}{\theta \preceq_N \theta} \quad (\textsc{Sub-NRefl})$$

$$\frac{\tau \preceq_P \sigma \qquad \theta \succeq_N \delta}{(\tau \leftarrow \theta) \preceq_{P \leftarrow N} (\tau \leftarrow \theta)} \quad (\textsc{Sub-PTo}) \qquad \frac{\tau \succeq_P \sigma \qquad \theta \preceq_N \delta}{(\tau \to \theta) \preceq_{P \to N} (\sigma \to \delta)} \quad (\textsc{Sub-NTo})$$

$$\frac{\theta \preceq_N \delta}{\downarrow\theta \preceq_{\downarrow N} \downarrow\delta} \quad (\textsc{Sub-PShift}) \qquad \frac{\tau \preceq_P \sigma}{\uparrow\tau \preceq_{\uparrow P} \uparrow\sigma} \quad (\textsc{Sub-NShift})$$

$$\frac{\tau \preceq_P \sigma_1 \qquad \tau \preceq_P \sigma_2}{\tau \preceq_P \sigma_1 \wedge \sigma_2} \quad (\textsc{Sub-PInt-R}) \qquad \frac{\theta_1 \preceq_N \delta \qquad \theta_2 \preceq_N \delta}{\theta_1 \vee \theta_2 \preceq_N \delta} \quad (\textsc{Sub-NUni-L})$$

$$\frac{\tau_1 \wedge \tau_2 :: P \quad \exists i \in \{1,2\}.\ \tau_i \preceq_P \sigma}{\tau_1 \wedge \tau_2 \preceq_P \sigma} \quad (\textsc{Sub-PInt-L}) \qquad \frac{\delta_1 \vee \delta_2 :: N \quad \exists i \in \{1,2\}.\ \theta \preceq_N \delta_i}{\theta \preceq_N \delta_1 \vee \delta_2} \quad (\textsc{Sub-NUni-R})$$

$$\frac{\tau :: P}{\tau \preceq_P \top_P} \quad (\textsc{Sub-PTop}) \qquad \frac{\theta :: N}{\bot_N \preceq_N \theta} \quad (\textsc{Sub-NBot})$$

Fig. 4. Subtyping rules

$$\frac{}{\Theta, x : \tau \vdash x : \tau \mid \Xi}\ (\text{T-PVar}) \qquad \frac{}{\Theta \mid \alpha : \theta \vdash \alpha : \theta, \Xi}\ (\text{T-NVar})$$

$$\frac{\Theta, x : \tau \vdash v : \downarrow\theta \mid \Xi}{\Theta \vdash \lambda x.v : \downarrow(\tau \to \theta) \mid \Xi}\ (\text{T-PAbs}) \qquad \frac{\Theta \mid e : \uparrow\tau \vdash \alpha : \theta, \Xi}{\Theta \mid \lambda\alpha.e : \uparrow(\tau \leftarrow \theta) \vdash \Xi}\ (\text{T-NAbs})$$

$$\frac{c : (\Theta \vdash \alpha : \theta, \Xi)}{\Theta \vdash \mu\alpha.c : \downarrow\theta \mid \Xi}\ (\text{T-PMu}) \qquad \frac{c : (\Theta, x : \tau \vdash \Xi)}{\Theta \mid \mu x.c : \uparrow\tau \vdash \Xi}\ (\text{T-NMu})$$

$$\frac{\Theta \vdash v : \tau \mid \Xi \qquad \Theta \mid e : \theta \vdash \Xi}{\Theta \vdash v \triangleleft e : \tau \leftarrow \theta \mid \Xi}\ (\text{T-PApp}) \qquad \frac{\Theta \mid e : \theta \vdash \Xi \qquad \Theta \vdash v : \tau \mid \Xi}{\Theta \mid v \triangleright e : \tau \to \theta \vdash \Xi}\ (\text{T-NApp})$$

$$\frac{\Theta \vdash v : \tau_1 \mid \Xi \qquad \Theta \vdash v : \tau_2 \mid \Xi}{\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi}\ (\text{T-PInt}) \qquad \frac{\Theta \mid e : \theta_1 \vdash \Xi \qquad \Theta \mid e : \theta_2 \vdash \Xi}{\Theta \mid e : \theta_1 \vee \theta_2 \vdash \Xi}\ (\text{T-NUni})$$

$$\frac{v :: P}{\Theta \vdash v : \top_P \mid \Xi}\ (\text{T-PTop}) \qquad \frac{e :: N}{\Theta \mid e : \bot_N \vdash \Xi}\ (\text{T-NBot})$$

$$\frac{\Theta \vdash v : \tau \mid \Xi \quad \tau \preceq_P \tau'}{\Theta \vdash v : \tau' \mid \Xi}\ (\text{T-PSub}) \qquad \frac{\Theta \mid e : \theta \vdash \Xi \quad \theta' \preceq_N \theta}{\Theta \mid e : \theta' \vdash \Xi}\ (\text{T-NSub})$$

$$\frac{\Theta \mid e : \uparrow\tau \vdash \Xi \qquad \Theta \vdash v : \tau \mid \Xi}{\langle v \mathbin{\Rsh} e \rangle : (\Theta \vdash \Xi)}\ (\text{T-PCmd}) \qquad \frac{\Theta \vdash v : \downarrow\theta \mid \Xi \qquad \Theta \mid e : \theta \vdash \Xi}{\langle v \mathbin{\Lsh} e \rangle : (\Theta \vdash \Xi)}\ (\text{T-NCmd})$$

Fig. 5. Typing rules

$x : \tau \in \Theta$ for a positive variable x of sort P, then we assume that $\tau :: P$. It might be better to write $x : \tau :: P$ to make the sort explicit, but we do not choose this heavy notation.

There are three kinds of judgements, similar to the sort system.

Typing rules The complete list of the typing rules are in Fig. 5.

A.4 Untyped $\bar{\lambda}\mu\tilde{\mu}$

The syntax of *terms*, *commands* and *co-terms* is given by:

$$v ::= x \mid \lambda x.v \mid \mu\alpha.c \qquad c ::= \langle v \mid e \rangle \qquad e ::= \alpha \mid v \cdot e \mid \tilde{\mu}x.c,$$

where x and α are variables of different kinds. *Values* and *co-values* are distinguished terms and co-terms, respectively, defined by $V ::= x \mid \lambda x.v$ and $E ::= \alpha \mid v \cdot e$. There are three reduction rules:

$$\langle \lambda x.v \mid v' \cdot e \rangle \longrightarrow \langle v' \mid \tilde{\mu} x.\langle v \mid e \rangle \rangle \quad \langle \mu\alpha.c \mid e \rangle \longrightarrow c[e/\alpha] \quad \langle v \mid \tilde{\mu} x.c \rangle \longrightarrow c[v/x].$$

The calculus has a critical pair $\langle \mu\alpha.c \mid \tilde{\mu} x.c' \rangle$, to which both of the last two rules are applicable, and is not confluent. The conflict is resolved by an evaluation strategy. The former rule has the priority in the call-by-value calculus, and the latter has in call-by-name. Hence, in the call-by-value calculus, the third rule is applicable only if $v \neq \mu\alpha.c'$, and thus changed to $\langle V \mid \tilde{\mu} x.c \rangle \longrightarrow c[V/x]$. Similarly, in call-by-name, the second rule is changed to $\langle \mu\alpha.c \mid E \rangle \longrightarrow c[E/\alpha]$.

We write $=_\beta^{\mathrm{cbv}}$ for the equivalence relation on expressions induced by the call-by-value reduction relation, and $=_\beta^{\mathrm{cbn}}$ for call-by-name.

A.5 Intersection and union type system for call-by-value $\bar{\lambda}\mu\tilde{\mu}$

Let $\mathtt{V}$ be a positive sort variable with the equation $\mathtt{V} = \downarrow(\mathtt{V} \to (\uparrow\mathtt{V}))$.

Let $\mathcal{X}$ be non-empty set of atomic types. Let $\mathcal{X}_-$ be the family of atomic types defined by:

$$\begin{aligned} \mathcal{X}_{\mathtt{V}\to(\uparrow\mathtt{V})} &:= \mathcal{X} \\ \mathcal{X}_{\mathtt{V}} &:= \emptyset \\ \mathcal{X}_{\uparrow\mathtt{V}} &:= \emptyset \\ \mathcal{X}_{\downarrow\uparrow\mathtt{V}} &:= \emptyset \end{aligned}$$

and, for other cases, $\mathcal{X}_P := \{P\}$ and $\mathcal{X}_N := \{N\}$. Then every sort has a $\{\top, \bot\}$-free refinement type.

Types The syntax of types is given by the following grammar:

$$\begin{aligned} \psi &::= a \mid \nu \to \varphi \mid \psi \vee \psi \mid \bot_{\mathtt{V}\to(\uparrow\mathtt{V})} & \varphi &::= \uparrow\nu \mid \varphi \vee \varphi \mid \bot_{\uparrow\mathtt{V}} \\ \nu &::= \downarrow\psi \mid \nu \wedge \nu \mid \top_{\mathtt{V}} & \varrho &::= \downarrow\varphi \mid \rho \wedge \rho \mid \top_{\downarrow\uparrow\mathtt{V}}. \end{aligned}$$

It is easy to see that the above syntax properly defines refinement types of $\mathtt{V} \to (\uparrow\mathtt{V})$, $\mathtt{V}$, $\uparrow\mathtt{V}$, and $\downarrow\uparrow\mathtt{V}$, respectively.

Subtyping The subtyping relation is inherited from the refinement type system for $\bar{\lambda}\mu\tilde{\mu}_P$.

Type environments and judgements A positive type environment Θ is a finite set of type binding of the form $x : \nu$. A negative type environment Ξ is a set of type binding of the form $\alpha : \varphi$. Note that they can be considered as type environments of the refinement type system for $\bar{\lambda}\mu\tilde{\mu}_P$.

There are four kinds of type judgements:

- $\Theta \vdash^{\mathrm{cbv}} V : \nu \Rsh \Xi$,
- $\Theta \vdash^{\mathrm{cbv}} v : \varrho \mid \Xi$,
- $\Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi$, and
- $c : (\Theta \vdash^{\mathrm{cbv}} \Xi)$.

Typing rules The complete list of typing rules is found in Fig. 6 and 7.

A.6 $\lambda^{\to\times}$ and intersection types

Here we introduce a calculus $\lambda^{\to\times}$ and its intersection type system. We do not formally define a simply-typed version of $\lambda^{\to\times}$ with recursive sorts, since it is obvious.

Terms The syntax of terms is given as follows.

$$M ::= x \mid \lambda x.M \mid \lambda(x,y).M \mid M\,M \mid (M,M).$$

The meaning of the term constructors is obvious.

The base rules of the reduction relation are listed below:

$$(\lambda x.M)\,N \longrightarrow M[N/x]$$
$$(\lambda(x_1,x_2).M)\,(N_1,N_2) \longrightarrow M[N_1/x_1, N_2/x_2].$$

Reduction is full, i.e. a redex of any position can be reduced.

We write π_1 for $\lambda(x,y).x$ and π_2 for $\lambda(x,y).y$. Then $\pi_i(M_1,M_2) \longrightarrow M_i$.

Types The syntax of types is given as follows.

$$A ::= a \mid A \to A \mid A \times A \mid A \wedge A.$$

A type is *simple* is it does not contain $\wedge$.

Subyping Figure 8 defines the subtyping relation.

For a technical convenience, we require the side condition on the last two rules. It is easy to see that this side condition is harmless, i.e. the set of derivable subtyping relation does not increase when we employ the unrestricted rules.

Typing rules The typing rules are in Figure 9. These are the straightforward extension of the standard intersection type system to a calculus with pairs.

B Supplementary Materials for Section 2

B.1 Proof of Confluence (Proposition 1)

It is straightforward to prove Proposition 1 by using the notions of parallel reduction and complete development. Note that we cannot apply Newman's lemma to the calculus with recursive sorts, although weak Church-Rosser can be proved easily.

$$\frac{}{\Theta, x : \nu \vdash^{\mathrm{cbv}} x : \nu \between \Xi} \quad \text{(CH-VVAR)}$$

$$\frac{\Theta, x : \nu \vdash^{\mathrm{cbv}} v : \downarrow\varphi \mid \Xi}{\Theta \vdash^{\mathrm{cbv}} \lambda x.v : \downarrow(\nu \to \varphi) \between \Xi} \quad \text{(CH-VABS)}$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} V : \nu \between \Xi}{\Theta \vdash^{\mathrm{cbv}} V : \downarrow\uparrow\nu \mid \Xi} \quad \text{(CH-VL)}$$

$$\frac{c : (\Theta \vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)}{\Theta \vdash^{\mathrm{cbv}} \mu\alpha.c : \downarrow\varphi \mid \Xi} \quad \text{(CH-TMU)}$$

$$\frac{}{\Theta \mid \alpha : \varphi \vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi} \quad \text{(CH-CVAR)}$$

$$\frac{c : (\Theta, x : \nu \vdash^{\mathrm{cbv}} \Xi)}{\Theta \mid \tilde{\mu}x.c : \uparrow\nu \vdash^{\mathrm{cbv}} \Xi} \quad \text{(CH-CMU)}$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} v : \downarrow(\bigvee_{i\in I}(\uparrow\nu_i)) \mid \Xi \qquad \Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi}{\Theta \mid v \cdot e : \uparrow(\bigwedge_{i\in I}\downarrow(\nu_i \to \varphi)) \vdash^{\mathrm{cbv}} \Xi} \quad \text{(CH-CAPP)}$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} v : \downarrow\varphi \mid \Xi \qquad \Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi}{\langle v \mid e\rangle : (\Theta \vdash^{\mathrm{cbv}} \Xi)} \quad \text{(CH-COM)}$$

Fig. 6. Typing rules for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$ (Part 1: computational rules)

$$\frac{\Theta \vdash^{\mathrm{cbv}} V : \nu_1 \fatsemi \Xi \qquad \Theta \vdash^{\mathrm{cbv}} V : \nu_2 \fatsemi \Xi}{\Theta \vdash^{\mathrm{cbv}} V : \nu_1 \wedge \nu_2 \fatsemi \Xi} \quad (\text{CH-VInt})$$

$$\frac{}{\Theta \vdash^{\mathrm{cbv}} V : \top_{\mathtt{v}} \fatsemi \Xi} \quad (\text{CH-VTop})$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi \qquad \nu \preceq_{\mathtt{v}} \nu'}{\Theta \vdash^{\mathrm{cbv}} V : \nu' \fatsemi \Xi} \quad (\text{CH-VSub})$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} v : \varrho_1 \mid \Xi \qquad \Theta \vdash^{\mathrm{cbv}} v : \varrho_2 \mid \Xi}{\Theta \vdash^{\mathrm{cbv}} v : \varrho_1 \wedge \varrho_2 \mid \Xi} \quad (\text{CH-TInt})$$

$$\frac{}{\Theta \vdash^{\mathrm{cbv}} v : \top_{\downarrow\uparrow\mathtt{v}} \mid \Xi} \quad (\text{CH-TTop})$$

$$\frac{\Theta \vdash^{\mathrm{cbv}} v : \varrho \mid \Xi \qquad \varrho \preceq_{\downarrow\uparrow\mathtt{v}} \varrho'}{\Theta \vdash^{\mathrm{cbv}} v : \varrho' \mid \Xi} \quad (\text{CH-TSub})$$

$$\frac{\Theta \mid e : \varphi_1 \vdash^{\mathrm{cbv}} \Xi \qquad \Theta \mid e : \varphi_2 \vdash^{\mathrm{cbv}} \Xi}{\Theta \mid e : \varphi_1 \vee \varphi_2 \vdash^{\mathrm{cbv}} \Xi} \quad (\text{CH-CUni})$$

$$\frac{}{\Theta \mid e : \bot_{\uparrow\mathtt{v}} \vdash^{\mathrm{cbv}} \Xi} \quad (\text{CH-CBot})$$

$$\frac{\Theta \mid e : \varphi \vdash^{\mathrm{cbv}} \Xi \qquad \varphi' \preceq_{\uparrow\mathtt{v}} \varphi}{\Theta \mid e : \varphi' \vdash^{\mathrm{cbv}} \Xi} \quad (\text{CH-CSub})$$

Fig. 7. Typing rules for the call-by-value $\bar{\lambda}\mu\tilde{\mu}$ (Part 2: intersection, union and subtyping rules)

$$\frac{}{a \leq a}$$

$$\frac{A \geq A' \qquad B \leq B'}{A \to B \leq A' \to B'}$$

$$\frac{A \leq A' \qquad B \leq B'}{A \times B \leq A' \times B'}$$

$$\frac{A \leq B_1 \qquad A \leq B_2}{A \leq B_1 \wedge B_2}$$

$$\frac{A_1 \leq B \qquad B \neq B_1 \wedge B_2}{A_1 \wedge A_2 \leq B}$$

$$\frac{A_2 \leq B \qquad B \neq B_1 \wedge B_2}{A_1 \times A_2 \leq B}$$

Fig. 8. Subtyping rules of intersection types for $\lambda^{\to\times}$

B.2 Proof of Type Preservation (Proposition 2)

This subsection gives a routine proof of Subject Reduction of the sort system for $\bar{\lambda}\mu\tilde{\mu}_P$ with recursive sort. As usual, we appeal to Substitution Lemma and Inversion.

We define a map *top* by:

$$\begin{aligned}
top(p) &:= p \\
top(\mathtt{P}) &:= top(P) && (\text{where } \mathtt{P} = P \in \mathcal{E}) \\
top(P \leftarrow N) &:= \leftarrow \\
top(\downarrow N) &:= \downarrow \\
top(n) &:= n \\
top(\mathtt{N}) &:= top(N) && (\text{where } \mathtt{N} = N \in \mathcal{E}) \\
top(P \to N) &:= \to \\
top(\uparrow P) &:= \uparrow.
\end{aligned}$$

Recall that, if $\mathtt{P} = P \in \mathcal{E}$, then P is not a sort variable. Hence *top* is well-defined.

Lemma 9. *If $P \sim Q$, then $top(P) = top(Q)$. If $N \sim M$, then $top(N) = top(N)$.*

Proof. By induction on the derivation of $P \sim Q$ and $N \sim M$. □

Hence, for example, $\downarrow N \not\sim (Q \leftarrow M)$.

$$\frac{A \leq A' \qquad A' \neq B_1 \wedge B_2}{\Gamma, x : A \vdash x : A'}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

$$\frac{\Gamma, x : A, y : B \vdash M : C}{\Gamma \vdash \lambda(x, y).M : A \times B \to C}$$

$$\frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\, N : B}$$

$$\frac{\Gamma \vdash M_1 : A_1 \qquad \Gamma \vdash M_2 : A_2}{\Gamma \vdash (M_1, M_2) : A_1 \times A_2}$$

$$\frac{\Gamma \vdash M : A_1 \qquad \Gamma \vdash M : A_2}{\Gamma \vdash M : A_1 \wedge A_2}$$

$$\frac{\Gamma \vdash M : A \qquad A \leq B}{\Gamma \vdash M : B}$$

Fig. 9. Typing rules of the intersection type system for $\lambda^{\to\times}$

Lemma 10.

- $(P \to N) \sim (P' \to N')$ *implies* $P \sim P'$ *and* $N \sim N'$.
- $\downarrow N \sim \downarrow N'$ *implies* $N \sim N'$.
- $(P \leftarrow N) \sim (P' \leftarrow N')$ *implies* $N \sim N'$ *and* $P \sim P'$.
- $\uparrow P \sim \uparrow P'$ *implies* $P \sim P'$.

Proof. We prove stronger results. For example, for the first claim, we prove the following result.

> If $(P \to N) \sim M$, then either $M = (P' \to N')$ or M is a negative sort variable, say $\mathtt{M}$, and $(\mathtt{M} = (P' \to N')) \in \mathcal{E}$. Furthermore $P \sim P'$ and $N \sim N'$. A similar statement holds for $M \sim (P \to N)$.

This claim can be easily proved by induction on the structure of the derivation of $(P \to N) \sim M$ (or $M \sim (P \to N)$), using Lemma 9. □

Lemma 11 (Substitution). *Assume* $\Gamma \vdash v : P \mid \Delta$. *Then:*

$$\begin{array}{rcl} \Gamma, x : P \vdash v' : Q \mid \Delta & \textit{implies} & \Gamma \vdash v'[v/x] : Q \mid \Delta \\ \Gamma, x : P \mid e : N \vdash \Delta & \textit{implies} & \Gamma \mid e[v/x] : N \vdash \Delta \\ c : (\Gamma, x : P \vdash \Delta) & \textit{implies} & c[v/x] : (\Gamma \vdash \Delta). \end{array}$$

Assume $\Gamma \mid e : N \vdash \Delta$. *Then:*

$$\begin{array}{rcl} \Gamma \vdash v : P \mid \alpha : N, \Delta & \textit{implies} & \Gamma \vdash v[e/\alpha] : P \mid \Delta \\ \Gamma \mid e' : M \vdash \alpha : N, \Delta & \textit{implies} & \Gamma \mid e'[e/\alpha] : M \vdash \Delta \\ c : (\Gamma \vdash \alpha : N, \Delta) & \textit{implies} & c[e/\alpha] : (\Gamma \vdash \Delta). \end{array}$$

Proof. By induction on derivations. □

Lemma 12 (Inversion).

- *If* $\Gamma \vdash x :: P \mid \Delta$, *then* $x :: Q \in \Gamma$ *and* $P \sim Q$ *for some* Q.
- *If* $\Gamma \vdash \lambda x.v :: P \mid \Delta$, *then* $P \sim \downarrow(Q \to N)$ *and* $\Gamma, x :: Q \vdash v :: \downarrow N \mid \Delta$ *for some* Q *and* N.
- *If* $\Gamma \vdash \mu\alpha.c :: P \mid \Delta$, *then* $P \sim \downarrow N$ *and* $c :: (\Gamma \vdash \alpha :: N, \Delta)$ *for some* N.
- *If* $\Gamma \vdash e \triangleleft v :: P \mid \Delta$, *then* $P \sim Q \leftarrow N$ *and* $\Gamma \vdash v :: Q \mid \Delta$ *and* $\Gamma \mid e :: N \vdash \Delta$ *for some* N *and* Q.
- *If* $\Gamma \mid \alpha :: N \mid \Delta$, *then* $\alpha :: M \in \Delta$ *and* $N \sim M$ *for some* M.
- *If* $\Gamma \mid \lambda\alpha.e :: N \vdash \Delta$, *then* $N \sim \uparrow(P \leftarrow M)$ *and* $\Gamma \mid e :: \uparrow P \vdash \alpha :: M, \Delta$ *for some* M *and* P.
- *If* $\Gamma \mid \mu x.c :: N \vdash \Delta$, *then* $N \sim \uparrow P$ *and* $c :: (\Gamma, x :: P \vdash \Delta)$ *for some* P.
- *If* $\Gamma \mid e \triangleright v :: N \vdash \Delta$, *then* $N \sim P \leftarrow M$ *and* $\Gamma \vdash v :: P \mid \Delta$ *and* $\Gamma \mid e :: M \vdash \Delta$ *for some* M *and* P.

Proof. By induction on the structure of derivations. □

We prove Proposition 2 by induction on the structure of the derivation of $t \longrightarrow t'$.

We first prove the base cases.

Suppose that $t = \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ and $t' = \langle (v_1[v_2/x]) \downarrow e \rangle$. By the assumption $t :: (\Gamma \vdash \Delta)$. Then we have

$$\Gamma \vdash \lambda x.v_1 :: \downarrow N \mid \Delta$$

and

$$\Gamma \mid v_2 \triangleright e :: N \vdash \Delta$$

for some N. By Inversion (Lemma 12), the former implies $N \sim (Q \to M)$ and

$$\Gamma, x :: Q \vdash v_1 :: \downarrow M \mid \Delta$$

for some Q and M. Again, by Inversion, the latter implies $N \sim (Q' \to M')$ and

$$\Gamma \vdash v_2 :: Q' \mid \Delta$$

and

$$\Gamma \mid e :: M' \vdash \Delta$$

for some Q' and M'. By Lemma 10, $Q \sim Q'$ and $M \sim M'$. Hence $\Gamma \vdash v_2 :: Q$ and $\Gamma \mid e :: M \vdash \Delta$. By Substitution Lemma (Lemma 11), we have

$$\Gamma \vdash v_1[v_2/x] :: \downarrow M \mid \Delta.$$

Hence

$$\langle v_1[v_2/x] \downarrow e \rangle :: (\Gamma \vdash \Delta).$$

Suppose that $t = \langle (\mu\alpha.c) \downarrow e \rangle$ and $t' = c[e/\alpha]$. By the assumption, one has $t :: (\Gamma \vdash \Delta)$. Then we have

$$\Gamma \vdash \mu\alpha.c :: \downarrow N \mid \Delta$$

and

$$\Gamma \mid e :: N \vdash \Delta.$$

By Inversion (Lemma 12), there exists N' such that $c :: (\Gamma \vdash \alpha :: N', \Delta)$ and $N \sim N'$. Then one has $\Gamma \mid e :: N' \vdash \Delta$. By Substitution Lemma (Lemma 11), we conclude that

$$c[v/x] : (\Gamma \vdash \Delta).$$

Suppose that $t = v \triangleleft e$ and $t' = v' \triangleleft e$ with $v \longrightarrow v'$. By the assumption, one has $\Gamma \vdash v \triangleleft e :: P \mid \Delta$ for some P. By Inversion (Lemma 12), $P \sim Q \leftarrow N$ and $\Gamma \vdash v :: Q \mid \Delta$ and $\Gamma \mid e :: N \vdash \Delta$ for some N and Q. By the induction hypothesis, $\Gamma \vdash v' :: Q \mid \Delta$. Hence $\Gamma \vdash v \triangleleft e :: Q \leftarrow N \mid \Delta$. By applying (S-P$\sim$), we obtain $\Gamma \vdash v \triangleleft e :: P \mid \Delta$.

Other cases are similar.

Remark 3. Term-level distinction of negative and positive cuts plays an important role in the proof. For example, if

$$\langle e \downarrow v \rangle :: (\Gamma \vdash \Delta),$$

then we have

$$\Gamma \vdash v :: N \mid \Delta \quad \text{and} \quad \Gamma \mid e :: \downarrow N \vdash \Delta$$

without confusing the case that

$$\Gamma \vdash v :: \uparrow P \mid \Delta \quad \text{and} \quad \Gamma \mid e :: P \vdash \Delta$$

because the latter derives a different judgement, $\langle e \Uparrow v \rangle :: (\Gamma \vdash \Delta)$.

B.3 Proof of Lemma 1

Lemma 13. $(t[v/x])^* = t^*[v^*/x]$. *Similarly* $(t[e/\alpha])^* = t^*[e^*/\alpha]$.

Proof. By induction on the structure of t. □

The first claim By induction on the structure of the derivation of the reduction relation $t \longrightarrow u$.

We first prove the base cases.

Suppose that $t = \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ and $u = \langle (v_1[v_2/x]) \downarrow e \rangle$. Then

$$\begin{aligned}
t^* &= (\lambda x.v_1)^* \, (v_2 \triangleright e)^* \\
&= (\lambda(x,\beta).(v_1^* \, \beta)) \, (v_2^*, e^*) && \text{(where } \beta \text{ is fresh)} \\
&\longrightarrow v_1^*[v_2^*/x] \, e^* \\
&= (v_1[v_2/x])^* \, e^* && \text{(by Lemma 13)} \\
&= \langle v_1[v_2/x] \downarrow e \rangle^* \\
&= u^*.
\end{aligned}$$

Suppose that $t = \langle (\mu\alpha.c) \downarrow e \rangle$ and $u = c[e/\alpha]$. Then

$$\begin{aligned}
t^* &= (\mu\alpha.c)^* \, e^* \\
&= (\lambda\alpha.c^*) \, e^* \\
&\longrightarrow c^*[e^*/\alpha] \\
&= (c[e/\alpha])^* && \text{(by Lemma 13)} \\
&= u^*.
\end{aligned}$$

The other base cases are the dual of the above.

The congruence cases are easy.

The second claim By induction on the structure of the derivation of $t^* \longrightarrow u$.

We first prove the base cases.

Suppose that $t^* = (\lambda x.t_1)\, t_2$ and $u = t_1[t_2/x]$, where t_1 and t_2 are terms of $\lambda^{\to\times}$. Then either $t = \langle v \downarrow e\rangle$ or $t = \langle v \uparrow e\rangle$.

– Suppose that $t = \langle v \downarrow e\rangle$. Then $v^* = \lambda x.t_1$ and $t_2 = e^*$. Hence $v = \mu x.c$ and $c^* = t_1$, where x is a *negative* variable in $\bar{\lambda}\mu\tilde{\mu}_P$. Then we have

$$\begin{aligned} t &= \langle v \downarrow e\rangle \\ &= \langle \mu x.c \downarrow e\rangle \\ &\longrightarrow c[e/x]. \end{aligned}$$

By Lemma 13, one has $(c[e/x])^* = c^*[e^*/x]$ as desired.

– Suppose that $t = \langle v \uparrow e\rangle$. Then $e^* = \lambda x.t_1$ and $t_2 = v^*$. Hence $e = \mu x.c$ and $c^* = t_1$. Then we have

$$\begin{aligned} t &= \langle v \uparrow e\rangle \\ &= \langle v \uparrow \mu x.c\rangle \\ &\longrightarrow c[v/x]. \end{aligned}$$

By Lemma 13, one has $(c[v/x])^* = c^*[v^*/x]$ as desired.

Suppose that $t^* = (\lambda(x_1, x_2).t_1)\,(t_{21}, t_{22})$ and $u = t_1[t_{21}/x_1, t_{22}/x_2]$, where t_1, t_{21} and t_{22} are terms of $\lambda^{\to\times}$. Then either $t = \langle v \downarrow e\rangle$ or $\langle v \uparrow e\rangle$.

– Suppose that $t = \langle v \downarrow e\rangle$. Then $v^* = \lambda(x_1, x_2).t_1$ and $e^* = (t_{21}, t_{22})$. Hence $v = \lambda x_1.v_1$ and $t_1 = v_1^*\, x_2$, where x_2 is a *negative* variable and does not appear in v_1. We also have $e = v_2 \triangleright e_2$ and $(v_2)^* = t_{21}$ and $e_2^* = t_{22}$. Then we have

$$\begin{aligned} t &= \langle v \downarrow e\rangle \\ &= \langle \lambda x_1.v_1 \downarrow v_2 \triangleright e_2\rangle \\ &\longrightarrow \langle v_1[v_2/x_1] \downarrow e_2\rangle. \end{aligned}$$

Now

$$\begin{aligned} &\langle v_1[v_2/x_1] \downarrow e_2\rangle^* = && (v_1[v_2/x_1])^*\, e_2^* \\ =&v_1^*[v_2^*/x_1]\, e_2^* && \text{(by Lemma 13)} \\ =&(v_1^*[v_2^*/x_1, e_2^*/x_2])\,(x_2[v_2^*/x_1, e_2^*/x_2]) && \text{(since } x_2 \text{ does not appear in } v_1\text{)} \\ =&(v_1^*\, x_2)[v_2^*/x_1, e_2^*/x_2] \\ =&t_1[t_{21}/x_1, t_{22}/x_2] \\ =&u. \end{aligned}$$

– Suppose that $t = \langle v \uparrow e\rangle$. This case is similar to the above case.

The congruence cases are easy.

C Supplementary Materials for Section 3

C.1 Proof of Substitution Lemma 3

By induction on the structure of derivations of judgements having x in their environment.

- Case (T-PVar) and $v' = x$: Then $\sigma = \tau$ and thus $\Theta \vdash v : \sigma \mid \Xi$.
- Case (T-PVar) and $v' = y \neq x$: Then $v'[v/x] = y$. Hence $\Theta \vdash v'[v/x] : \sigma \mid \Xi$.
- Case (T-PAbs): Then $v' = \lambda y.v''$ and $\sigma = \downarrow(\sigma' \to \theta')$. We can assume without loss of generality that y is not occur in v and we get $\Theta, y : \sigma', x : \tau \vdash v'' : \downarrow\theta' \mid \Xi$. By the induction hypothesis, we have $\Theta, y : \sigma' \vdash v''[v/x] : \downarrow\theta' \mid \Xi$. So $\Theta \vdash \lambda y.(v''[v/x]) : \downarrow(\sigma' \to \theta') \mid \Xi$.
- Case (T-PMu): Then $v' = \mu\alpha.c$ and $\sigma = \downarrow\theta$ and $c : (\Theta, x : \tau \vdash \alpha : \theta, \Xi)$. By the induction hypothesis, $c[v/x] : (\Theta \vdash \alpha : \theta, \Xi)$ and thus $\Theta \vdash \mu\alpha.c : \downarrow\theta \mid \Xi$.
- Case (T-PApp): Then $v' = v'' \triangleleft e$ and $\sigma = \theta' \leftarrow \sigma'$. We have $\Theta, x : \tau \vdash v'' : \sigma' \mid \Xi$ and $\Theta, x : \tau \mid e : \theta' \vdash \Xi$. By the induction hypothesis, $\Theta \vdash v''[v/x] : \sigma' \mid \Xi$ and $\Theta \mid e[v/x] : \theta' \vdash \Xi$, which implies $\Theta \vdash (v'' \triangleleft e)[v/x] : \theta' \leftarrow \sigma' \mid \Xi$.
- Case (T-PCmd): Then $c = \langle v'' \Rsh e \rangle$ and $\Theta, x : \tau \vdash v'' : \sigma \mid \Xi$ and $\Theta, x : \tau \mid e : \uparrow\sigma \vdash \Xi$. By the induction hypothesis, $\Theta \vdash v''[v/x] : \sigma \mid \Xi$ and $\Theta \mid e[v/x] : \uparrow\sigma \vdash \Xi$. Hence $(\langle v'' \Rsh e \rangle[v/x]) : (\Theta \vdash \Xi)$.
- Case (T-PInt): Then $\sigma = \sigma_1 \wedge \sigma_2$ and $\Theta, x : \tau \vdash v' : \sigma_i \mid \Xi$ for $i \in \{1, 2\}$. By the induction hypothesis, $\Theta \vdash v'[v/x] : \sigma_i \mid \Xi$ for $i \in \{1, 2\}$. By (T-PInt), we have $\Theta \vdash v'[v/x] : \sigma_1 \wedge \sigma_2 \mid \Xi$.
- Case (T-PTop): Then $\sigma = \top$ and $\Theta \vdash v'[v/x] : \top \mid \Xi$ trivially holds.
- Case (T-PSub): Then $\Theta, x : \tau \vdash v' : \sigma' \mid \Xi$ for some $\sigma' \leq \sigma$. By the induction hypothesis, $\Theta \vdash v'[v/x] : \sigma' \mid \Xi$. By the subtyping rule, $\Theta \vdash v'[v/x] : \sigma \mid \Xi$.

Other cases are the dual of the above.

The claim about substitution of co-terms can be proved by a similar way.

C.2 Proof of Inversion Lemma (Lemma 4)

We prove a stronger version of the lemma, what we call General Inversion here. The statement of Lemma 4 is a weaker version that suffices to prove Subject Reduction (Proposition 4).

We define

$$\bigwedge_{1 \leq i \leq k} \tau_i \quad := \quad (((\tau_1 \wedge \tau_2) \wedge \cdots) \wedge \tau_{k-1}) \wedge \tau_k$$

and

$$\bigvee_{1 \leq i \leq k} \theta_i \quad := \quad (((\theta_1 \vee \theta_2) \vee \cdots) \vee \theta_{k-1}) \vee \theta_k.$$

Lemma 14 (General Inversion).

1. *If* $\Theta \vdash x : \tau \mid \Xi$, *there exists* τ' *such that*
   - $\tau' \preceq \tau$, *and*
   - $x : \tau' \in \Theta$.
2. *If* $\Theta \vdash \lambda x.v : \tau \mid \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \downarrow(\sigma_i \to \theta_i) \}_{1 \le i \le k}$ *such that*
   - $\bigwedge_{1 \le i \le k} (\downarrow(\sigma_i \to \theta_i)) \preceq \tau$, *and*
   - $\Theta, x : \sigma_i \vdash v : \downarrow\theta_k \mid \Xi$ *for every* i $(1 \le i \le k)$.
3. *If* $\Theta \vdash \mu\alpha.c : \tau \mid \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \downarrow\theta_i \}_{1 \le i \le k}$ *such that*
   - $\bigwedge_{1 \le i \le k} (\downarrow\theta_i) \preceq \tau$, *and*
   - $c : (\Theta \vdash \alpha : \theta_i, \Xi)$ *for every* i $(1 \le i \le k)$.
4. *If* $\Theta \vdash e \triangleleft v : \tau \mid \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \sigma_i \leftarrow \theta_i \}_{1 \le i \le k}$ *such that*
   - $\bigwedge_{1 \le i \le k} (\sigma_i \leftarrow \theta_i) \preceq \tau$, *and*
   - $\Theta \vdash v : \sigma_i \mid \Xi$ *and* $\Theta \mid e : \theta_i \vdash \Xi$ *for every* i $(1 \le i \le k)$.
5. *If* $\Theta \mid \alpha : \theta \vdash \Xi$, *there exists* θ' *such that*
   - $\theta \preceq \theta'$, *and*
   - $\alpha : \theta' \in \Xi$.
6. *If* $\Theta \mid \lambda\alpha.e : \theta \vdash \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \uparrow(\tau_i \leftarrow \delta_i) \}_{1 \le i \le k}$ *such that*
   - $\theta \preceq \bigvee_{1 \le i \le k} (\uparrow(\tau_i \leftarrow \delta_i))$, *and*
   - $\Theta \mid e : \uparrow\tau_i \vdash \alpha : \delta_i, \Xi$ *for every* i $(1 \le i \le k)$.
7. *If* $\Theta \mid \mu x.c : \theta \vdash \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \uparrow\tau_i \}_{1 \le i \le k}$ *such that*
   - $\theta \preceq \bigvee_{1 \le i \le k} (\uparrow\tau_i)$, *and*
   - $c : (\Theta, x : \tau_i \vdash \Xi)$ *for every* i $(1 \le i \le k)$.
8. *If* $\Theta \mid e \triangleright v : \theta \vdash \Xi$, *there exists a (possibly empty) finite collection of types* $\{ \tau_i \to \delta_i \}_{1 \le i \le k}$ *such that*
   - $\theta \preceq \bigvee_{1 \le i \le k} (\tau_i \to \delta_i)$, *and*
   - $\Theta \vdash v : \tau_i \mid \Xi$ *and* $\Theta \mid e : \delta_i \vdash \Xi$ *for every* i $(1 \le i \le k)$.
9. *If* $\langle v \downarrow e \rangle : (\Theta \vdash \Xi)$, *there exists* θ *such that*
   - $\Theta \vdash v : \downarrow\theta \mid \Xi$ *and* $\Theta \mid e : \theta \vdash \Xi$.
10. *If* $\langle e \uparrow v \rangle : (\Theta \vdash \Xi)$, *there exists* τ *such that*
    - $\Theta \vdash v : \tau \mid \Xi$ *and* $\Theta \mid e : \uparrow\tau \vdash \Xi$.

Proof. By induction on the structure of the derivations. We prove (2) for example.

Suppose that $\Theta \vdash \lambda x.v : \tau \mid \Xi$. There are four rules that can derive this judgement:

- (T-PAbs): Then $\tau = \downarrow(\sigma \to \theta)$ and $\Theta, x : \sigma \vdash v : \downarrow\theta \mid \Xi$.
- (T-PInt): Then $\tau = \tau_1 \wedge \tau_2$ and $\Theta \vdash v : \tau_1 \mid \Xi$ and $\Theta \vdash v : \tau_2 \mid \Xi$. By the induction hypothesis, each judgement has a finite collection of types that satisfies the requirements. Then the union of the collection satisfies the requirements for $\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi$.
- (T-PTop): Then $\tau = \top$ and the empty collection satisfies the requirement.

– (T-PSub): Then $\Theta \vdash \lambda x.v : \tau' \mid \Xi$ and $\tau' \preceq \tau$. By the induction hypothesis, we have a finite collection $\{ \downarrow(\sigma_i \to \theta_i) \}_{1 \le i \le k}$ that satisfies the requirements. Since $\bigwedge_{1 \le i \le k}(\downarrow(\sigma_i \to \theta_i)) \preceq \tau' \preceq \tau$, this collection is what we need.

□

Basically Inversion Lemma (Lemma 4) is a consequence of the above general lemma. We prove the first claim of Lemma 4 for example.

Suppose that $\Theta \vdash \lambda x.v : \downarrow\theta \mid \Xi$. By Lemma 14, there is a finite collection of types $\{ \downarrow(\tau_i \to \delta_i) \}_{1 \le i \le k}$ such that

$$\bigwedge_{1 \le i \le k} (\downarrow(\tau_i \to \delta_i)) \preceq \downarrow\theta$$

and $\Theta, x : \tau_i \vdash v : \downarrow\delta_i \mid \Xi$ for every i $(1 \le i \le k)$. We claim that there exists j such that $\tau_j \to \delta_j \preceq \theta$. We prove this claim by induction on the structure of the derivation of $\bigwedge_{1 \le i \le k}(\downarrow(\tau_i \to \delta_i)) \preceq \downarrow\theta$. If the last rule is (Sub-PShift), then $k = 1$ and $\tau_1 \to \sigma_1 \preceq \theta$ as desired. If the last rule is (Sub-PIntL), then we have either $\bigwedge_{1 \le i \le k-1}(\downarrow(\tau_i \to \delta_i)) \preceq \downarrow\theta$ or $\downarrow(\tau_k \to \delta_k) \preceq \downarrow\theta$. For the former case, we appeal to the induction hypothesis. For the latter case, let $j = k$.

C.3 Subject Reduction (Proposition 4)

We prove the base cases.

– Case $c = \langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ and $c' = \langle (v_1[v_2/x]) \downarrow e \rangle$:
Assume that $\langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle : (\Theta \vdash \Xi)$. Then $\Theta \vdash \lambda x.v_1 : \downarrow\theta \mid \Xi$ and $\Theta \mid v_2 \triangleright e : \theta \vdash \Xi$. By Lemma 4, one has $\Theta, x : \sigma \vdash v_1 : \downarrow\delta \mid \Xi$ and $\sigma \to \delta \preceq \theta$ for some σ and δ. By the subtyping rule, we have $\Theta \mid v_2 \triangleright e : \sigma \to \delta \vdash \Xi$. Again, by Lemma 4, $\Theta \vdash v_2 : \sigma \mid \Xi$ and $\Theta \mid e : \delta \vdash \Xi$. By Substitution Lemma (Lemma 3), $\Theta \vdash v_1[v_2/x] : \downarrow\delta \mid \Xi$. Hence $\langle v_1[v_2/x] \downarrow e \rangle : (\Theta \vdash \Xi)$.
– Case $c = \langle (\mu\alpha.c_0) \downarrow e \rangle$ and $c' = c_0[e/\alpha]$:
Assume that $\langle (\mu\alpha.c_0) \downarrow e \rangle : (\Theta \vdash \Xi)$. Then $\Theta \vdash \mu\alpha.c_0 : \downarrow\theta \mid \Xi$ and $\Theta \mid e : \theta \vdash \Xi$ for some θ. By Lemma 4, one has $c_0 : (\Theta \vdash \alpha : \theta, \Xi)$. By Substitution Lemma (Lemma 3), we have $c_0[e/\alpha] : (\Theta \vdash \Xi)$.

The other cases are easy.

C.4 De-substitution (Lemma 5)

We first prove the case in which v' is a variable.

– Case $v' = x$: Let $\tau = \sigma$.
– Case $v' = y \neq x$: Let $\tau = \top$.

We prove the other cases by induction on the structure of derivations. Since v' is not a variable, the top-level structure of $v'[v/x]$ is that of v'.

- Case (T-PAbs): Then $v' = \lambda y.v''$ and $\sigma = \downarrow(\sigma' \to \theta')$. We can assume without loss of generality that y is not occur in v and we get $\Theta, y : \sigma', \vdash v''[v/x] : \downarrow\theta' \mid \Xi$. By the induction hypothesis, we have $\Theta, y : \sigma', x : \tau \vdash v'' : \downarrow\theta' \mid \Xi$ and $\Theta, y : \sigma' \vdash v : \tau \mid \Xi$ for some τ. Since y does not appear in v, the latter can be refined as $\Theta \vdash v : \tau \mid \Xi$. By applying (T-PAbs), we have $\Theta, x : \tau \vdash v'' : \downarrow(\sigma' \to \theta') \mid \Xi$ as desired.
- Case (T-PMu): Then $v' = \mu\alpha.c$ and $\sigma = \downarrow\theta$ and $c[v/x] : (\Theta \vdash \alpha : \theta, \Xi)$. We can assume without loss of generality that α does not appear in v. By the induction hypothesis, there exists θ such that $c : (\Theta, x : \tau \vdash \alpha : \theta, \Xi)$ and $\Theta \vdash v : \tau \mid \alpha : \theta, \Xi$, which can be strengthened as $\Theta \vdash v : \tau \mid \Xi$. The former judgement leads to $\Theta \vdash \mu\alpha.c : \downarrow\theta \mid \Xi$.
- Case (T-PApp): Then $v' = v'' \triangleleft e$ and $\sigma = \sigma' \leftarrow \theta'$. We have $\Theta \vdash v''[v/x] : \sigma' \mid \Xi$ and $\Theta \mid e[v/x] : \theta' \vdash \Xi$. By the induction hypothesis, there exist τ_1 and τ_2 such that
  - $\Theta, x : \tau_1 \vdash v'' : \sigma \mid \Xi$,
  - $\Theta \vdash v : \tau_1 \mid \Xi$,
  - $\Theta, x : \tau_2 \mid e : \theta' \vdash \Xi$, and
  - $\Theta \vdash v : \tau_2 \mid \Xi$.

  Then we have $\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi$. Since $\tau_1 \wedge \tau_2 \leq \tau_1$, by Lemma 2, one has $\Theta, x : \tau_1 \wedge \tau_2 \vdash v'' : \sigma \mid \Xi$. Similarly $\Theta, x : \tau_1 \wedge \tau_2 \mid e : \theta' \vdash \Xi$. Hence $\Theta, x : \tau_1 \wedge \tau_2 \vdash v'' \triangleleft e : \sigma' \leftarrow \theta' \mid \Xi$.
- Case (T-PCmd): Then $c = \langle v'' \Rsh e \rangle$ and $\Theta \vdash v''[v/x] : \sigma \mid \Xi$ and $\Theta \mid e[v/x] : \uparrow\sigma \vdash \Xi$ for some σ. By the induction hypothesis for the former judgement, we have $\Theta, x : \tau_1 \vdash v'' : \sigma \mid \Xi$ and $\Theta \vdash v : \tau_1 \mid \Xi$. By the induction hypothesis for the latter judgement, we have $\Theta, x : \tau_2 \mid e : \uparrow\sigma \vdash \Xi$ and $\Theta \vdash v : \tau_2 \mid \Xi$. By the same argument as in the above case, one has $\Theta, x : \tau_1 \wedge \tau_2 \vdash v'' : \sigma \mid \Xi$ and $\Theta, x : \tau_1 \wedge \tau_2 \vdash v'' : \uparrow\sigma \vdash \Xi$, which implies $\langle v'' \Rsh e \rangle : (\Theta, x : \tau_1 \wedge \tau_2 \vdash \Xi)$. We have $\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi$ as required.
- Case (T-PInt): Then $\sigma = \sigma_1 \wedge \sigma_2$ and $\Theta \vdash v'[v/x] : \sigma_i \mid \Xi$ for $i \in \{1, 2\}$. By the induction hypothesis, $\Theta, x : \tau_i \vdash v' : \sigma_i \mid \Xi$ and $\Theta \vdash v : \tau_i \mid \Xi$ for some τ_i for $i \in \{1, 2\}$. By the same argument as in the above case, we have $\Theta, x : \tau_1 \wedge \tau_2 \vdash v' : \sigma_i \mid \Xi$ for $i \in \{1, 2\}$ and $\Theta \vdash v : \tau_1 \wedge \tau_2 \mid \Xi$. By (P-Int-I), we have $\Theta, x : \tau_1 \wedge \tau_2 \vdash v' : \sigma_1 \wedge \sigma_2 \mid \Xi$.
- Case (P-$\top$): Let $\tau = \top$. Then $\Theta \vdash v' : \top \mid \Xi$ and $\Theta \vdash v : \top \mid \Xi$ trivially hold.
- Case (P-Sub): Then $\Theta \vdash v'[v/x] : \sigma' \mid \Xi$ for some $\sigma' \leq \sigma$. By the induction hypothesis, $\Theta, x : \tau \vdash v' : \sigma' \mid \Xi$ and $\Theta \vdash v : \tau \mid \Xi$ for some τ. By the subtyping rule, $\Theta, x : \tau \vdash v' : \sigma \mid \Xi$.

The other cases are the dual of the above.

C.5 Subject Expansion (Proposition 5)

We prove the base cases.

- Case $c = \langle (\lambda x.v_1) \Lsh (v_2 \triangleright e) \rangle$ and $c' = \langle (v_1[v_2/x]) \Lsh e \rangle$: Assume that $\langle (v_1[v_2/x]) \Lsh e \rangle : (\Theta \vdash \Xi)$. Then $\Theta \vdash v_1[v_2/x] : \downarrow\theta \mid \Xi$ and $\Theta \mid e : \theta \vdash \Xi$. By

De-substitution Lemma (Lemma 5), we have σ such that $\Theta, x : \sigma \vdash v_1 : \downarrow\theta \mid \Xi$ and $\Theta \vdash v_2 : \sigma \mid \Xi$ for some σ. Hence we have $\Theta \vdash \lambda x.v_1 : \downarrow(\sigma \to \theta) \mid \Xi$ and $\Theta \mid v_2 \triangleright e : \sigma \to \theta \vdash \Xi$. Now $\langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle : (\Theta \vdash \Xi)$ as desired.

– Case $c = \langle (\mu\alpha.c_0) \downarrow e \rangle$ and $c' = c_0[e/\alpha]$: Assume that $c_0[e/\alpha] : (\Theta \vdash \Xi)$. By De-substitution Lemma (Lemma 5), we have θ such that $c_0 : (\Theta \vdash \alpha : \theta, \Xi)$ and $\Theta \mid e : \theta \vdash \Xi$. The former implies $\Theta \vdash \mu\alpha.c_0 : \downarrow\theta \mid \Xi$. Hence $\langle (\mu\alpha.c_0) \downarrow e \rangle : (\Theta \vdash \Xi)$.

The other cases are similar.

C.6 Relationship to the intersection type system for $\lambda^{\to\times}$ (Lemma 6)

Fix an atomic type $o :: O$. Recall that the translation of types is given by:

$$(\tau \leftarrow \theta)^* := \tau^* \times \theta^* \qquad (\tau \to \theta)^* := \tau^* \times \theta^*$$
$$(\downarrow\theta)^* := \theta^* \to o \qquad (\uparrow\tau)^* := \tau^* \to o$$
$$(\tau_1 \wedge \tau_2)^* := \tau_1^* \wedge \tau_2^* \qquad (\theta_1 \vee \theta_2)^* := \theta_1^* \wedge \theta_2^*,$$

where we assume a fixed translation of atomic sorts. Note that $\top_P$ and $\bot_P$ does not appear since we focus on the $\{\top, \bot\}$-free subsystem.[5]

We first prove that the translation of types preserves the subtyping relation. We note here that subtyping $\theta \preceq_N \delta$ of negative types is translated to subtyping $\theta^* \geq \delta^*$ in the opposite direction.

Lemma 15. *If $\tau \preceq_P \sigma$, then $\tau^* \leq \sigma^*$. If $\theta \preceq_N \delta$, then $\theta^* \geq \delta^*$.*

Proof. By induction on the structure of the subtyping derivation.

– (Sub-PRefl) and (Sub-NRefl): Obvious.
– (Sub-PTo): Suppose that $\tau \preceq_P \sigma$ and $\theta \succeq_N \delta$. By the induction hypothesis, $\tau^* \leq \sigma^*$ and $\theta^* \leq \delta^*$. Hence

$$(\tau \leftarrow \theta)^* \;=\; \tau^* \times \theta^* \;\leq\; \sigma^* \times \delta^* \;=\; (\sigma \leftarrow \delta)^*.$$

– (Sub-NTo): Suppose that $\tau \succeq_P \sigma$ and $\theta \preceq_N \delta$. By the induction hypethesis, $\tau^* \geq \sigma^*$ and $\theta^* \geq \delta^*$. Hence

$$(\tau \to \theta)^* \;=\; \tau^* \times \theta^* \;\geq\; \sigma^* \times \delta^* \;=\; (\sigma \to \delta)^*.$$

– (Sub-PShift): Suppose that $\theta \preceq_N \delta$. By the induction hypothesis, $\theta^* \geq \delta^*$. Hence

$$(\downarrow\theta)^* \;=\; \theta^* \to o \;\leq\; \delta^* \to o \;=\; (\downarrow\delta)^*.$$

[5] $\top_P$ and $\bot_N$ can be translated to the type for all terms, which is usually written as ω, although our intersection type system for $\lambda^{\to\times}$ does not have such a constant.

– (Sub-NShift): Suppose that $\tau \preceq_P \sigma$. By the induction hypothesis, $\tau^* \leq \sigma^*$. Hence

$$(\uparrow\tau)^* \;=\; \tau^* \to o \;\geq\; \sigma^* \to o \;=\; (\uparrow\sigma)^*.$$

– (Sub-PInt-R): Suppose that $\tau \preceq_P \sigma_1$ and $\tau \preceq_P \sigma_2$. By the induction hypothesis, $\tau^* \leq \sigma_1^*$ and $\tau \leq \sigma_2^*$. Hence

$$\tau^* \;\leq\; \sigma_1^* \wedge \sigma_2^* \;=\; (\sigma_1 \wedge \sigma_2)^*.$$

– (Sub-NUni-L): Suppose that $\theta_1 \preceq_N \delta$ and $\theta_2 \preceq_N \delta$. By the induction hypothesis, $\theta_1^* \geq \delta^*$ and $\theta_2^* \geq \delta^*$. Hence

$$(\theta_1 \vee \theta_2)^* \;=\; \theta_1^* \wedge \theta_2^* \;\geq\; \delta^*.$$

– (Sub-PInt-L): Suppose (w.o.l.g.) that $\tau_1 \preceq_P \sigma$ (and $\tau_1 \wedge \tau_2 :: P$). By the induction hypothesis, $\tau_1^* \leq \sigma^*$. Hence,

$$(\tau_1 \wedge \tau_2)^* \;=\; \tau_1^* \wedge \tau_2^* \;\leq\; \sigma^*.$$

– (Sub-NUni-R): Suppose (w.o.l.g.) that $\theta \preceq_N \delta_1$ (and $\delta_1 \vee \delta_2 :: N$). By the induction hypothesis, $\theta^* \geq \delta_1^*$. Hence,

$$\theta^* \;\geq\; \delta_1^* \wedge \delta_2^* \;=\; (\delta_1 \vee \delta_2)^*.$$

□

The proof of Lemma 6 is easy induction using the above lemma.

C.7 Typability of Normal Forms in the Subsystem (Lemma 7)

Definition 1 (Canonical Form). *An expression in* canonical form *is defined by the following grammar.*

$$\check{v} ::= x \mid \lambda x.\check{v} \mid \mu\alpha.\check{c} \mid \check{v} \triangleleft \check{e}$$
$$\check{c} ::= \langle x \downarrow \check{e}\rangle \mid \langle \lambda x.\check{v} \downarrow \alpha\rangle \mid \langle \check{v} \Rsh \alpha\rangle \mid \langle x \Rsh \lambda\alpha.\check{e}\rangle$$
$$\check{e} ::= \alpha \mid \lambda\alpha.\check{e} \mid \mu x.\check{c} \mid \check{v} \triangleright \check{e}$$

Lemma 16. *A well-sorted expression in normal form is in canonical form.*

Proof. By induction on the structure of expressions.

– $c = \langle v \downarrow e\rangle$ with $c :: (\Gamma \vdash \Delta)$: Then one has $\Gamma \vdash v :: \downarrow N \mid \Delta$ and $\Gamma \mid e : N \vdash \Delta$. There are four subcases:
  • $v = x$: Then $c = \langle x \downarrow e\rangle$. By the induction hypothesis, e is in canonical form, i.e., $e = \check{e}$. Hence $c = \langle x \downarrow \check{e}\rangle$ is in canonical form.

- $v = \lambda x.v'$: By Inversion Lemma for the sort system (Lemma 12), we have $\downarrow N \sim \downarrow(Q \to M)$ and $\Gamma, x :: Q \vdash v :: \downarrow M \mid \Delta$ for some Q and M. By Lemma 10, $N \sim Q \to M$ and thus $\Gamma \mid e : Q \to M \vdash \Delta$. If $e = \lambda\alpha.e'$, then by Inversion (Lemma 12), one has $N \sim \uparrow(Q' \leftarrow M')$, which contradicts to Lemma 10. By the same reason, $e \neq \mu x.c'$. If $e = v'' \triangleright e'$, then $c = \langle \lambda x.v' \downarrow v'' \triangleright e' \rangle$ is reducible, which contradicts to the assumption. Hence $e = \alpha$. By the induction hypothesis, $v = \lambda x.v'$ is in canonical form, i.e. $v = \check{v}$ for some $\check{v}$, and thus $c = \langle \check{v} \downarrow \alpha \rangle$ is in canonical form.
- $v = \mu\alpha.c$: Then $c = \langle \mu\alpha.c \downarrow e \rangle$ is reducible, a contradiction.
- $v = v' \triangleleft e'$: By Inversion (Lemma 12), $\downarrow N \sim (Q \leftarrow M)$, which contradicts to Lemma 10.

Other cases are easy. □

We prove a bit stronger result.

Lemma 17. *A well-sorted term v in normal form satisfies $\Theta \Vdash v : \tau \mid \Xi$ for some τ, Θ and Ξ. Furthermore τ can be chosen in such a way that τ is not atomic unless the sort for v is atomic. The similar statement holds for co-terms and commands.*

Proof. By Lemma 16, we can assume without loss of generality that expressions are in canonical form. We prove this claim by induction on the structure of expressions in the canonical form.

We first prove the claim for terms (using the induction hypothesis for terms as well as co-terms and commands).

- $\check{v} = x$: Let P be the sort of x. By the assumption ($\star$) (see Section 3.6), there exists a $\{\top, \bot\}$-free type $\tau :: P$. It is easy to see that, if P is not an atomic sort, then there exists non-atomic τ that refines P. For example, if $P \sim \downarrow N$ for some N, let $\tau = \downarrow\theta$, where θ is a $\{\top, \bot\}$-free type $\theta :: N$. We have $x : \tau \Vdash x : \tau \parallel \cdot$.
- $\check{v} = \lambda x.\check{v}'$: By the induction hypothesis, one has $\Theta \Vdash \check{v}' : \tau \parallel \Xi$ for some Θ, τ, and Ξ. Since the sort of $\check{v}$ is $\downarrow N$ for some N by Inversion (Lemma 12). Hence τ is not atomic. We can assume without loss of generality that τ is not an intersection (if $\tau = \tau_1 \wedge \tau_2$, then $\Gamma \Vdash \check{v}' : \tau_1 \parallel \Xi$ by the subtyping rule). Hence $\tau = \uparrow\theta$ for some $\theta :: N$.
  If $x : \sigma \in \Theta$, i.e. $\Theta = \Theta', x : \sigma$, then $\Theta' \Vdash \lambda x.\check{v}' : \downarrow(\tau \to \theta) \parallel \Xi$. Suppose that x is not in Θ. Let σ be a $\{\top, \bot\}$-free refinement type of the sort of x. Then $\Theta, x : \sigma \Vdash \check{v}' : \downarrow\theta \parallel \Xi$, and thus $\Theta \Vdash \lambda x.\check{v}' : \downarrow(\tau \to \theta) \parallel \Xi$.
- $\check{v} = \mu\alpha.\check{c}$: By the induction hypothesis, $\check{c} : (\Theta \Vdash \Xi)$ for some Θ and Ξ. If $\alpha : \theta \in \Xi$, then $\Xi = \Xi', \alpha : \theta$ and $\Theta \Vdash \mu\alpha.c : \downarrow\theta \mid \Xi'$. Suppose that α is not in Ξ. Let θ be a $\{\top, \bot\}$-free refinement type of the sort of α. Then $\check{c} : (\Theta \Vdash \alpha : \theta, \Xi)$ and thus $\Theta \Vdash \mu\alpha.c : \downarrow\theta \parallel \Xi$.
- $\check{v} = \check{v}' \triangleleft \check{e}'$: By the induction hypothesis, we have $\Theta_1 \Vdash \check{v}' : \tau \parallel \Xi_1$ and $\Theta_2 \parallel e : \theta \Vdash \Xi_2$. Then $\Theta_1 \wedge \Theta_2 \Vdash \check{v}' \triangleleft \check{e}' : \tau \leftarrow \theta \mid \Xi_1 \vee \Xi_2$, where $\Theta_1 \wedge \Theta_2$ and $\Xi_1 \vee \Xi_2$ is defined as the point-wise operations.

The proof for co-terms is similar to the above. We prove the claim for commands.

- $\check{c} = \langle x \downarrow \check{e} \rangle$: By the induction hypothesis, one has $\Theta \parallel e : \theta \Vdash \Xi$. We have $x : \downarrow\theta \Vdash x : \downarrow\theta \parallel \cdot$. Hence $\langle x \downarrow \check{e} \rangle : (\Theta \wedge (x : \downarrow\theta) \Vdash \Xi)$.
- $\check{c} = \langle \lambda x.\check{v} \downarrow \alpha \rangle$: By the induction hypothesis, one has $\Theta \Vdash \lambda x.\check{v} : \tau \parallel \Xi$. Since the sort of $\lambda x.\check{v}$ is $\downarrow N$ for some N by Inversion (Lemma 12), we can assume that τ is not atomic. Because $\tau :: \downarrow N$, we have $\tau = \downarrow\theta$ for some $\theta :: N$. Then we have $\cdot \parallel \alpha : \theta \Vdash \alpha : \theta$ and thus $\langle \lambda x.\check{v} \downarrow \alpha \rangle : (\Theta \Vdash \Xi \vee (\alpha : \theta))$.

Other cases are similar to the above. □

C.8 Completeness for Strong Normalisation (Theorem 2)

We first prove De-substitution Lemma for the $\{\top, \bot\}$-free subsystem. Recall that derivablity in the subsystem is written as $\Theta \Vdash v : \tau \parallel \Xi$.

Lemma 18 (De-substitution). *Suppose that $\Gamma \vdash v :: P \mid \Delta$. Assume $\{\top, \bot\}$-free type environments $\Theta :: \Gamma$ and $\Xi :: \Delta$. Let x be a variable of sort P.*

- *If $\Theta \Vdash v'[v/x] : \sigma \parallel \Xi$ and x has a free occurrence in v', then $\Theta, x : \tau \Vdash v' : \sigma \parallel \Xi$ and $\Theta \Vdash v : \tau \parallel \Xi$ for some $\tau :: P$.*
- *If $\Theta \parallel e[v/x] : \theta \Vdash \Xi$ and x has a free occurrence in e, then $\Theta, x : \tau \parallel e : \theta \Vdash \Xi$ and $\Theta \Vdash v : \tau \parallel \Xi$ for some $\tau :: P$.*
- *If $(c[v/x]) : (\Theta \Vdash \Xi)$ and x has a free occurrence in c, then $c : (\Theta, x : \tau \Vdash \Xi)$ and $\Theta \Vdash v : \tau \parallel \Xi$ for some $\tau :: P$.*

The similar statements hold for substitution of co-terms.

Proof. The proof is basically the same as the proof of De-substitution Lemma for the full system (Lemma 5). We prove the claim by induction on the structure of the derivations.

We first prove the case in which v' is a variable.

- Case $v' = x$: Let $\tau = \sigma$.
- Case $v' = y \neq x$: This case never happens since x has a free occurrence.

We prove the other cases by induction on the structure of derivations. Since v' is not a variable, the top-level structure of $v'[v/x]$ is that of v'.

- Case (T-PAbs): Then $v' = \lambda y.v''$ and $\sigma = \downarrow(\sigma' \to \theta')$. We can assume without loss of generality that y is not occur in v and we get $\Theta, y : \sigma' \Vdash v''[v/x] : \downarrow\theta' \parallel \Xi$. Note that x has a free occurrence in v''. By the induction hypothesis, we have $\Theta, y : \sigma', x : \tau \Vdash v'' : \downarrow\theta' \parallel \Xi$ and $\Theta, y : \sigma' \Vdash v : \tau \parallel \Xi$ for some τ. Since y does not appear in v, the latter can be refined as $\Theta \Vdash v : \tau \parallel \Xi$. By applying (T-PAbs), we have $\Theta, x : \tau \Vdash v'' : \downarrow(\sigma' \to \theta') \parallel \Xi$ as desired.

- Case (T-PMu): Then $v' = \mu\alpha.c$ and $\sigma = \downarrow\theta$ and $c[v/x] : (\Theta \Vdash \alpha : \theta, \Xi)$. We can assume without loss of generality that α does not appear in v. Note that x has a free occurrence in c. By the induction hypothesis, there exists θ such that $c : (\Theta, x : \tau \Vdash \alpha : \theta, \Xi)$ and $\Theta \Vdash v : \tau \parallel \alpha : \theta, \Xi$, which can be strengthened as $\Theta \Vdash v : \tau \parallel \Xi$ (since α does not occur in v). The former judgement leads to $\Theta \Vdash \mu\alpha.c : \downarrow\theta \parallel \Xi$.
- Case (T-PApp): Then $v' = v'' \triangleleft e$ and $\sigma = \sigma' \leftarrow \theta'$. We have $\Theta \Vdash v''[v/x] : \sigma' \parallel \Xi$ and $\Theta \parallel e[v/x] : \theta' \Vdash \Xi$. There are four subcases:
  - Both v'' and e has a free occurrence of x: Then, by the induction hypothesis, there exist τ_1 and τ_2 such that
    - $\Theta, x : \tau_1 \Vdash v'' : \sigma \parallel \Xi$,
    - $\Theta \Vdash v : \tau_1 \parallel \Xi$,
    - $\Theta, x : \tau_2 \parallel e : \theta' \Vdash \Xi$, and
    - $\Theta \Vdash v : \tau_2 \parallel \Xi$.

    Then we have $\Theta \Vdash v : \tau_1 \wedge \tau_2 \parallel \Xi$, $\Theta, x : \tau_1 \wedge \tau_2 \Vdash v'' : \sigma \parallel \Xi$, and $\Theta, x : \tau_1 \wedge \tau_2 \parallel e : \theta' \Vdash \Xi$. Hence $\Theta, x : \tau_1 \wedge \tau_2 \Vdash v'' \triangleleft e : \sigma' \leftarrow \theta' \parallel \Xi$.
  - v'' has a free occurrence of x but e does not: Then, by the induction hypothesis, one has
    - $\Theta, x : \tau \Vdash v'' : \sigma \parallel \Xi$, and
    - $\Theta \Vdash v : \tau \parallel \Xi$.

    Since x does not appear in e, we have $e[v/x] = e$ and thus $\Theta \parallel e : \theta' \Vdash \Xi$. Because the weakening rule is admissible, $\Theta, x : \tau \parallel e : \theta' \Vdash \Xi$. Hence $\Theta, x : \tau \Vdash v'' \triangleleft e : \sigma' \leftarrow \theta' \parallel \Xi$ and $\Theta \Vdash v : \tau \parallel \Xi$.
  - e has a free occurrence of x but v'' does not: Similar to the above subcase.
  - Neither v'' nor e does not have a free occurrence of x: This never happens since $v'' \triangleleft e$ has a free occurrence by the assumption.
- Case (T-PCmd): Similar to the above case.
- Case (T-PInt): Then $\sigma = \sigma_1 \wedge \sigma_2$ and $\Theta \Vdash v'[v/x] : \sigma_i \parallel \Xi$ for $i \in \{1, 2\}$. By the induction hypothesis, $\Theta, x : \tau_i \Vdash v' : \sigma_i \parallel \Xi$ and $\Theta \Vdash v : \tau_i \parallel \Xi$ for some τ_i for $i \in \{1, 2\}$. By the same argument as in the above case, we have $\Theta, x : \tau_1 \wedge \tau_2 \Vdash v' : \sigma_i \parallel \Xi$ for $i \in \{1, 2\}$ and $\Theta \Vdash v : \tau_1 \wedge \tau_2 \parallel \Xi$. By (T-PInt), we have $\Theta, x : \tau_1 \wedge \tau_2 \Vdash v' : \sigma_1 \wedge \sigma_2 \parallel \Xi$.
- Case (P-Sub): Then $\Theta \Vdash v'[v/x] : \sigma' \parallel \Xi$ for some $\sigma' \preceq_P \sigma$. By the induction hypothesis, $\Theta, x : \tau \Vdash v' : \sigma' \parallel \Xi$ and $\Theta \Vdash v : \tau \parallel \Xi$ for some τ. By the subtyping rule, $\Theta, x : \tau \Vdash v' : \sigma \parallel \Xi$.

The other cases are the dual of the above. □

We prove Theorem 2 by induction on the length of the longest reduction sequence starting from the expression.

If the expression is in normal form, we use Lemma 7.

For reducible expressions, we prove the claim by induction on the structure of the derivation of the reduction relations. For example, consider a base case $\langle(\lambda x.v_1) \downarrow (v_2 \triangleright e)\rangle \longrightarrow \langle(v_1[v_2/x]) \downarrow e\rangle$. By the induction hypothesis, $\langle(v_1[v_2/x]) \downarrow e\rangle$ is typable in the $\{\top, \bot\}$-free subsystem. Assume that $\langle(v_1[v_2/x]) \downarrow e\rangle : (\Theta \Vdash \Xi)$, and hence $\Theta \Vdash v_1[v_2/x] : \downarrow\theta \parallel \Xi$ and $\Theta \parallel e : \theta \Vdash \Xi$ for some θ. There are two cases. If x occurs freely in v_1, then one can prove De-substitution

Lemma for the subsystem, which completes the proof for this case. If x does not occur in v_1, the naïve adaptation of De-substitution Lemma does not work, since the judgement obtained by the lemma is $\Theta \vdash v_2 : \top_P \mid \Xi$, which is not derivable in the subsystem. However, in this case, $\Theta, x : \tau \Vdash v_1 : \theta \parallel \Xi$ for every $\{\top, \bot\}$-free type τ (which refines the sort of x). By the induction hypothesis, one has $\Theta' \Vdash v_2 : \tau \parallel \Xi'$ in the $\{\top, \bot\}$-free subsystem for some Θ', Ξ', and τ'. Then we have $\Theta' \wedge \Theta \Vdash \lambda x.v_1 : \downarrow(\tau \to \theta) \parallel \Xi \vee \Xi'$ and $\Theta' \wedge \Theta \parallel v_2 \triangleright e : \tau \to \theta \Vdash \Xi \vee \Xi'$, and thus $\langle (\lambda x.v_1) \downarrow (v_2 \triangleright e) \rangle$ is typable in the subsystem.

D Supplementary Materials for Section 4

D.1 Proof of Proposition 6

Lemma 19. *Let t be an expression, V be a value and e be a co-term of $\bar{\lambda}\mu\tilde{\mu}$. Then $t^v[\Phi(V)/x] = (t[V/x])^v$ and $t^v[e^v/\alpha] = (t[e/\alpha])^v$.*

Proof. By induction on the structure of t □

Now we have the following reduction sequences:

$$\begin{aligned}\langle V \mid \tilde{\mu}x.c\rangle^v &= \langle \Downarrow \Phi(V) \downarrow \mu x.c^v\rangle \\ &\longrightarrow \langle \Phi(V) \Uparrow \mu x.c^v\rangle \\ &\longrightarrow c^v[\Phi(V)/x] \\ &= (c[V/x])^v.\end{aligned}$$

$$\begin{aligned}\langle \mu\alpha.c \mid e\rangle^v &= \langle \mu\alpha.c^v \downarrow e^v\rangle \\ &\longrightarrow c^v[e^v/\alpha] \\ &= (c[e/\alpha])^v.\end{aligned}$$

$$\begin{aligned}\langle \lambda x.v_1 \mid v_2 \cdot e\rangle^v &= \langle \Downarrow(\lambda x.v_1^v) \downarrow \mu f.\langle v_2^v \downarrow \mu y.\langle f \downarrow y \triangleright e^v\rangle\rangle\rangle \\ &\longrightarrow \langle \lambda x.v_1^v \Uparrow \mu f.\langle v_2^v \downarrow \mu y.\langle f \downarrow y \triangleright e^v\rangle\rangle\rangle \\ &\longrightarrow \langle v_2^v \downarrow \mu y.\langle \lambda x.v_1^v \downarrow y \triangleright e^v\rangle\rangle \\ &\longrightarrow \langle v_2^v \downarrow \mu x.\langle v_1^v \downarrow e^v\rangle\rangle \\ &= \langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e\rangle\rangle^v.\end{aligned}$$

D.2 Proof of Proposition 7

Lemma 20. *Let t be an expression, v be a term and E be a co-value of $\bar{\lambda}\mu\tilde{\mu}$. Then $t^n[v^n/x] = (t[v/x])^n$ and $t^n[\Psi(E)/\alpha] = (t[E/\alpha])^n$.*

Proof. By induction on the structure of t. □

Now we have the following reduction sequences.

$$\begin{aligned}\langle v \mid \tilde{\mu}x.c\rangle^n &= \langle v^n \Rsh \mu x.c^n\rangle \\ &\longrightarrow c^n[v^n/x] \\ &= (c[v/x])^n.\end{aligned}$$

$$\begin{aligned}\langle \mu\alpha.c \mid E\rangle^n &= \langle \mu\alpha.c^n \Rsh \Downarrow\Psi(E)\rangle \\ &\longrightarrow \langle \mu\alpha.c^n \Ldsh \Psi(E)\rangle \\ &\longrightarrow c^n[\Psi(E)/\alpha] \\ &= (c[E/\alpha])^n.\end{aligned}$$

$$\begin{aligned}\langle \lambda x.v_1 \mid v_2 \cdot e\rangle^n &= \langle (\lambda x.(\Downarrow v_1^v)) \Rsh \Downarrow\Psi(v_2 \cdot e)\rangle \\ &\longrightarrow \langle \lambda x.(\Downarrow v_1^n) \Ldsh \Psi(v_2 \cdot e)\rangle \\ &= \langle \lambda x.(\Downarrow v_1^n) \Ldsh v_2^n \rhd e^n\rangle \\ &\longrightarrow \langle \Downarrow v_1^n[v_2^n/x] \Ldsh e^n\rangle \\ &\longrightarrow \langle v_1^n[v_2^n/x] \Rsh e^n\rangle \\ &\longleftarrow \langle v_2^n \Rsh \mu x.\langle v_1^n \Rsh e^n\rangle\rangle \\ &= \langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e\rangle\,\rangle^n.\end{aligned}$$

They show that the translation preserves β-equivalences.

The latter claim is proved by contraposition. Suppose that t is not strongly normalising. Then one has an infinite reduction sequence

$$t = t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow \cdots.$$

We construct an infinite reduction sequence starting from t^n.

The crucial observation is as follows. Let us write $\longrightarrow_\lambda$ (resp. $\longrightarrow_\mu$, $\longrightarrow_{\tilde{\mu}}$) for reduction of λ-redex (resp. μ-redex, $\tilde{\mu}$-redex). For every context C, if

$$t = C[\langle \lambda x.v_1 \mid v_2 \cdot e\rangle] \longrightarrow_\lambda C[\langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e\rangle\,\rangle] \longrightarrow_{\tilde{\mu}} C[\langle v_1[v_2/x] \mid e\rangle] = t',$$

we write $t \longrightarrow_{\lambda'} t'$.

Lemma 21. *Assume an infinite reduction sequence starting from t. Then there exists an infinite reduction sequence*

$$t = t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow \cdots$$

such that either

- $t_0 \longrightarrow_\mu t_1$ *or* $t_0 \longrightarrow_{\tilde{\mu}}$, *or*
- $t_0 \longrightarrow_{\lambda'} t_2$.

Proof. We first note that every infinite sequence has infinite μ or $\tilde{\mu}$ reduction (since reducing λ-redex decreases the number of λ in the expression). Let us focus on the first such μ or $\tilde{\mu}$ reduction in the given infinite reduction sequence, say $t_i \longrightarrow t_{i+1}$. One can interchange the focused reduction with the previous λ reduction $t_{i-1} \longrightarrow_\lambda t_i$ in the following sense.

- If $t_{i-1} \longrightarrow_\lambda t_i \longrightarrow_\mu t_{i+1}$, then $t_{i-1} \longrightarrow_\mu t'_i \longrightarrow^*_\lambda t_{i+1}$.
- If $t_{i-1} \longrightarrow_\lambda t_i \longrightarrow_{\tilde{\mu}} t_{i+1}$ and $t_{i-1} \not\longrightarrow_{\lambda'} t_{i+1}$, then $t_{i-1} \longrightarrow_{\tilde{\mu}} t'_i \longrightarrow^*_\lambda t_{i+1}$.
- If $t_{i-2} \longrightarrow_\lambda t_{i-1} \longrightarrow_\lambda t_i \longrightarrow_{\tilde{\mu}} t_{i+1}$ with $t_{i-1} \longrightarrow_{\lambda'} t_{i+1}$, then $t_{i-2} \longrightarrow_{\lambda'} t'_i \longrightarrow^*_\lambda t_{i+1}$.

By applying this argument as much as required, we will have an infinite reduction sequence starting from $\longrightarrow_\mu$, $\longrightarrow_{\tilde{\mu}}$ or $\longrightarrow_{\lambda'}$.

Formally we appeal to induction on the position of the first μ or $\tilde{\mu}$ reduction. □

As a consequence of the previous lemma, we can assume without loss of generality that we have an infinite reduction sequence starting from t with respect to $(\xrightarrow{\text{cbn}'}) := (\longrightarrow_\mu) \cup (\longrightarrow_{\tilde{\mu}}) \cup (\longrightarrow_{\lambda'})$. Since $u_1 \xrightarrow{\text{cbn}'} u_2$ implies $u_1^n \longrightarrow^+ u_2^n$, we obtain an infinite reduction sequence starting from t^n.

E Supplementary Materials for Section 5

E.1 Proof of Lemma 8

The left-to-right direction is proved by induction on the structure of the derivation. Most cases are easily proved. For example, let us consider the case in which the last is (CH-VL). Then we have $\Theta \vdash^{\text{cbv}} V : \nu \between \Xi$. By the induction hypothesis, $\Theta \vdash \Phi(V) : \nu \mid \Xi$. Hence $\Theta \vdash \Phi(V) : \downarrow\uparrow\nu \mid \Xi$ is required.

The only nontrivial case is (CH-CApp). Suppose that $\Theta \vdash^{\text{cbv}} v : \downarrow\bigvee_{i\in I}(\uparrow\nu_i) \mid \Xi$ and $\Theta \mid e : \varphi \vdash^{\text{cbv}} \Xi$. By the induction hypothesis, we have

$$\Theta \vdash v^v : \downarrow(\bigvee_{i\in I}(\uparrow\nu_i)) \mid \Xi$$

and

$$\Theta \mid e^v : \varphi \vdash^{\text{cbv}} \Xi.$$

Then, for every $j \in I$, one has

$$\frac{\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi)), y : \nu_j \vdash y : \nu_j \mid \Xi \qquad \Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi)), y : \nu_j \mid e^v : \varphi \mid \Xi}{\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi)), y : \nu_j \mid y \triangleright e^v : \nu_j \to \varphi \vdash \Xi}.$$

Because $\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi)), y : \nu_j \vdash f : \downarrow(\nu_j \to \varphi) \mid \Xi$ for every $j \in I$ by the subtyping rule, one has

$$\langle y \triangleright e^v \Vdash f \rangle : (\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi)), y : \nu_j \vdash \Xi)$$

which implies

$$\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_j \to \varphi)) \mid \mu y.\langle y \triangleright e^v \Rsh f\rangle : \uparrow\nu_j \vdash \Xi.$$

Since this judgement is derivable for every $j \in I$, we have

$$\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_j \to \varphi)) \mid \mu y.\langle y \triangleright e^v \Rsh f\rangle : \bigvee_{i\in I}(\uparrow\nu_j) \vdash \Xi.$$

Hence

$$\langle v^v \Lsh \mu y.\langle y \triangleright e^v \Rsh f\rangle\rangle : (\Theta, f : \bigwedge_{i\in I}(\downarrow(\nu_j \to \varphi)) \vdash \Xi).$$

Thus

$$\Theta \mid \mu f.\langle v^v \Lsh \mu y.\langle y \triangleright e^v \Rsh f\rangle\rangle : \uparrow(\bigwedge_{i\in I}(\downarrow(\nu_j \to \varphi))) \vdash \Xi.$$

The right-to-left direction is proved by induction on the structure of the expression t (where we regard that the subject V for a value judgement $\Theta \vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$ is smaller than the same subject V for a term judgement $\Theta \vdash^{\mathrm{cbv}} V : \varrho \mid \Xi$).

Most rules are easily provable by using General Inversion Lemma (Lemma 14). For example, we show that $\Theta \vdash V^v : \varrho \mid \Xi$ implies $\Theta \vdash^{\mathrm{cbv}} V : \varrho \mid \Xi$. By definition of V^v, we have $\Theta \vdash \mu\alpha.\langle \Phi(V) \Rsh \alpha\rangle : \varrho \mid \Xi$. By Lemma 14, one has a finite collection of types $\{\downarrow\varphi_i\}_{i\in I}$ such that $\bigwedge_{i\in I}(\downarrow\varphi_i) \preceq \varrho$ and, for every $i \in I$, $\langle \Phi(V) \Rsh \alpha\rangle : (\Theta \vdash \alpha : \varphi_i, \Xi)$. By the same lemma, for every $i \in I$, there exists ν_i such that $\Theta \mid \alpha : \uparrow\nu_i \vdash \alpha : \varphi_i, \Xi$ and $\Theta \vdash \Phi(V) : \nu_i \mid \alpha : \varphi_i, \Xi$. The latter can be strengthened to $\Theta \vdash \Phi(V) : \nu_i \mid \Xi$ since V has no free occurrence of α. Again, by General Inversion (Lemma 14) and $\Theta \mid \alpha : \uparrow\nu_i \vdash \alpha : \varphi_i, \Xi$, one has $\uparrow\nu_i \preceq \varphi_i$ for every $i \in I$. By the induction hypothesis, $\Theta \vdash^{\mathrm{cbv}} V : \nu_i \fatsemi \Xi$ for every $i \in I$. Hence $\Theta \vdash^{\mathrm{cbv}} V : \downarrow\uparrow\nu_i \mid \Xi$ for every $i \in I$. Now we have $\Theta \vdash^{\mathrm{cbv}} V : \bigwedge_{i\in I}(\downarrow\uparrow\nu_i) \mid \Xi$. Since

$$\bigwedge_{i\in I}(\downarrow\uparrow\nu_i) \;\preceq\; \bigwedge_{i\in I}(\downarrow\varphi_i) \;\preceq\; \varrho,$$

we obtain $\Theta \vdash^{\mathrm{cbv}} V : \varrho \mid \Xi$ as desired.

The most complex case is $v \cdot e$. Suppose that $\Theta \mid (v \cdot e)^v : \varphi \vdash \Xi$. By definition,

$$\Theta \mid \mu f.\langle v^v \Lsh \mu y.\langle f \Lsh y \triangleright e^v\rangle\rangle : \varphi \vdash \Xi.$$

By General Inversion (Lemma 14), there is a finite collection of types $\{\uparrow\nu_i\}_{i\in I}$ such that $\varphi \preceq \bigvee_{i\in I}\uparrow\nu_i$ and $\langle v^v \Lsh \mu y.\langle f \Lsh y \triangleright e^v\rangle\rangle : (\Theta, f : \nu_i \vdash \Xi)$ for every $i \in I$. Again, by General Inversion (Lemma 14), for every $i \in I$, there exists φ_i' such that $\Theta, f : \nu_i \vdash v^v : \downarrow\varphi_i' \mid \Xi$ and $\Theta, f : \nu_i \mid \mu y.\langle f \Lsh y \triangleright e^v\rangle : \varphi_i' \vdash \Xi$.

The latter can be strengthened to $\Theta \vdash v^v : \downarrow\varphi'_i \mid \Xi$ since v does not have a free occurrence of f. Then, by the induction hypothesis, for every $i \in I$,

$$\Theta \vdash^{\mathrm{cbv}} v : \downarrow\varphi'_i \mid \Xi.$$

Because $\Theta, f : \nu_i \mid \mu y.\langle f \downarrow y \triangleright e^v\rangle : \varphi'_i \vdash \Xi$, by General Inversion (Lemma 14), there exists a finite family $\{\nu'_{i,j}\}_{j\in J_i}$ such that $\varphi'_i \preceq \bigvee_{j\in J_i} \uparrow\nu'_{i,j}$ and $\langle f \downarrow y \triangleright e^v\rangle : (\Theta, f : \nu_i, y : \nu'_{i,j} \vdash \Xi)$ for every $i \in I$ and $j \in J_i$. Again, by General Inversion (Lemma 14), for every i and $j \in J_i$, there exists $\psi_{i,j}$ such that $\Theta, f : \nu_i, y : \nu'_{i,j} \vdash f : \downarrow\psi_{i,j} \mid \Xi$ and $\Theta, f : \nu_j, y : \nu'_{i,j} \mid y \triangleright e^v : \psi_{i,j} \vdash \Xi$. By General Inversion (Lemma 14) for the judgements, for every $i \in I$ and $j \in J_i$, we have

- $\nu_i \preceq \downarrow\psi_{i,j}$, and
- there exists a family $\{\nu''_{i,j,k} \to \varphi'_{i,j,k}\}_{k\in K_{i,j}}$ such that
  - $\psi_{i,j} \preceq \bigvee_{k\in K_{i,j}} (\nu''_{i,j,k} \to \varphi'_{i,j,k})$,
  - $\Theta, f : \nu_j, y : \nu'_{i,j} \vdash y : \nu''_{i,j,k} \mid \Xi$ for every $k \in K_{i,j}$, and
  - $\Theta, f : \nu_j, y : \nu'_{i,j} \mid e^v : \varphi'_{i,j,k} \vdash \Xi$ for every $k \in K_{i,j}$.

Since f nor y dose not have a free occurrence in e, we have $\Theta \mid e^v : \varphi'_{i,j,k} \vdash \Xi$ for every $i \in I$, $j \in J_i$ and $k \in K_{i,j}$. By the induction hypothesis,

$$\Theta \mid e : \varphi'_{i,j,k} \vdash^{\mathrm{cbv}} \Xi.$$

Let $\varphi'' := \bigvee_{i\in I, j\in J_i, k\in K_{i,j}} \varphi'_{i,j,k}$. Then

$$\Theta \mid e : \varphi'' \vdash^{\mathrm{cbv}} \Xi.$$

Since $\Theta, f : \nu_j, y : \nu'_{i,j} \vdash y : \nu''_{i,j,k} \mid \Xi$ for every $k \in K_{i,j}$, we know that $\nu'_{i,j} \preceq \nu''_{i,j,k}$ for every $k \in K_{i,j}$. Recall that $\Theta \vdash^{\mathrm{cbv}} v : \downarrow\varphi'_i \mid \Xi$ for every $i \in I$ and $\varphi'_i \preceq \bigvee_{j\in J_i} (\uparrow\nu'_{i,j})$. Hence

$$\Theta \vdash^{\mathrm{cbv}} v : \downarrow\Big(\bigvee_{j\in J_i} (\uparrow\nu'_{i,j})\Big) \mid \Xi.$$

So by (CH-CApp), for every $i \in I$,

$$\Theta \mid v \cdot e : \uparrow\Big(\bigwedge_{j\in J_i} (\downarrow(\nu'_{i,j} \to \varphi''))\Big) \vdash^{\mathrm{cbv}} \Xi$$

and thus

$$\Theta \mid v \cdot e : \bigvee_{i\in I} \Big(\uparrow\Big(\bigwedge_{j\in J_i} (\downarrow(\nu'_{i,j} \to \varphi''))\Big)\Big) \vdash^{\mathrm{cbv}} \Xi.$$

It suffices to prove that $\varphi \preceq \bigvee_{i\in I} \big(\uparrow\big(\bigwedge_{j\in J_i} (\downarrow(\nu'_{i,j} \to \varphi''))\big)\big)$. We have

$$\varphi \preceq \bigvee_{i\in I} \uparrow\nu_i \preceq \bigvee_{i\in I} \uparrow\Big(\bigwedge_{j\in J_i} \downarrow\psi_{i,j}\Big) \preceq \bigvee_{i\in I} \uparrow\Big(\bigwedge_{j\in J_i} \downarrow\Big(\bigvee_{k\in K_{i,j}} (\nu''_{i,j,k} \to \varphi'_{i,j,k})\Big)\Big).$$

Recall that, for every $i \in I$, $j \in J_i$ and $k \in K_{i,j}$, one has $\nu'_{i,j} \preceq \nu''_{i,j,k}$ and $\varphi'_{i,j,k} \preceq \varphi''$. Hence

$$\bigvee_{k \in K_{i,j}} (\nu''_{i,j,k} \to \varphi'_{i,j,k}) \preceq \nu'_{i,j} \to \varphi''.$$

So we have

$$\varphi \preceq \bigvee_{i \in I} \uparrow\Big(\bigwedge_{j \in J_i} \downarrow\Big(\bigvee_{k \in K_{i,j}} (\nu''_{i,j,k} \to \varphi'_{i,j,k})\Big)\Big). \preceq \bigvee_{i \in I} \uparrow\Big(\bigwedge_{j \in J_i} \downarrow(\nu'_{i,j} \to \varphi'')\Big).$$

E.2 Proof of Theorem 4

As discussed is Section 5, Soundness follows from Lemma 8 and Corollary 3. Here we prove completeness.

We use $\Vdash^{\mathrm{cbv}}$ for judgements in the $\{\top, \bot\}$-free subsystem for call-by-value $\bar{\lambda}\mu\tilde{\mu}$.

Definition 2 (Canonical Form). *$\bar{\lambda}\mu\tilde{\mu}$-expressions in canonical form is defined by the following grammar:*

$$\check{V} ::= x \mid \lambda x.\check{v}$$
$$\check{v} ::= \check{V} \mid \mu\alpha.\check{c}$$
$$\check{c} ::= \langle x \mid \check{v} \cdot \check{e}\rangle \mid \langle \check{V} \mid \alpha\rangle$$
$$\check{e} ::= \alpha \mid \check{v} \cdot \check{e} \mid \tilde{\mu}x.\check{c}.$$

Lemma 22. *A $\bar{\lambda}\mu\tilde{\mu}$-expression that is normal with respect to the call-by-value reduction is in canonical form.*

Proof. By induction on the structure of the expression. We show the claim for commands. Other cases are easy.

Consider a command $\langle v \mid e\rangle$. Since it is in normal form, so are both v and e. By the induction hypothesis, $v = \check{v}$ and $e = \check{e}$.

- Case $\check{v} = x$:
  - Subcase $\check{e} = \alpha$: Then $\langle \check{v} \mid \check{e}\rangle = \langle x \mid \alpha\rangle$, which is in canonical form.
  - Subcase $\check{e} = \check{v}' \cdot \check{e}'$: Then $\langle \check{v} \mid \check{e}\rangle = \langle x \mid \check{v}' \cdot \check{e}'\rangle$, which is in canonical form.
  - Subcase $\check{e} = \tilde{\mu}y.\check{c}$: Then $\langle \check{v} \mid \check{e}\rangle = \langle x \mid \tilde{\mu}y.\check{c}\rangle \longrightarrow \check{c}[x/y]$, a contradiction.
- Case $\check{v} = \lambda x.\check{v}'$:
  - Subcase $\check{e} = \alpha$: Then $\langle \check{v} \mid \check{e}\rangle = \langle \lambda x.\check{v}' \mid \alpha\rangle$, which is in canonical form.
  - Subcase $\check{e} = \check{v}'' \cdot \check{e}'$: Then $\langle \check{v} \mid \check{e}\rangle = \langle \lambda x.\check{v}' \mid \check{v}'' \cdot \check{e}'\rangle \longrightarrow \langle \check{v}'' \mid \tilde{\mu}x.\langle \check{v}' \mid \check{e}'\rangle\rangle$, a contradiction.
  - Subcase $\check{e} = \tilde{\mu}y.\check{c}$: Then $\langle \check{v} \mid \check{e}\rangle = \langle \lambda x.\check{v}' \mid \tilde{\mu}y.\check{c}\rangle \longrightarrow \check{c}[\lambda x.\check{v}'/y]$, a contradiction.
- Case $\check{v} = \mu\alpha.\check{c}$: Then $\langle \check{v} \mid \check{e}\rangle = \langle \mu\alpha.\check{c} \mid \check{e}\rangle \longrightarrow \check{c}[\check{e}/\alpha]$, a contradiction.

□

We show that every expression in normal form is typable.

Lemma 23. *Every $\bar{\lambda}\mu\tilde{\mu}$-expression in normal form with respect to call-by-value reduction is typable in the $\{\top, \bot\}$-free subsystem.*

- *For every $\check{V}$, there exist ν, Θ and Ξ such that $\Theta \Vdash^{\mathrm{cbv}} \check{V} : \nu \fatsemi \Xi$.*
- *For every $\check{v}$, there exist φ, Θ and Ξ such that $\Theta \Vdash^{\mathrm{cbv}} \check{v} : \downarrow\varphi \parallel \Xi$.*
- *For every $\check{c}$, there exist Θ and Ξ such that $\check{c} : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$.*
- *For every $\check{e}$, there exist φ, Θ and Ξ such that $\Theta \parallel \check{e} : \varphi \Vdash^{\mathrm{cbv}} \Xi$.*

Proof. By induction on the structure of the expression.

- $\Theta \Vdash^{\mathrm{cbv}} \check{V} : \nu \fatsemi \Xi$
  - Case $\check{V} = x$: Then $x : \downarrow a \Vdash^{\mathrm{cbv}} x : \downarrow a \parallel \cdot$.
  - Case $\check{V} = \lambda x.\check{v}$: By the induction hypothesis, $\Theta \Vdash^{\mathrm{cbv}} \check{v} : \downarrow\varphi \parallel \Xi$ for some Θ and Ξ. We can assume without loss of generality that $\Theta = (\Theta', x : \nu)$ (otherwise, consider $\Theta, x : \downarrow a$). We have $\Theta' \Vdash^{\mathrm{cbv}} \lambda x.\check{v}\downarrow(\nu \to \varphi) \fatsemi \Xi$.
- $\Theta \Vdash^{\mathrm{cbv}} \check{v} : \downarrow\varphi \parallel \Xi$
  - Case $\check{v} = \check{V}$: By the induction hypothesis, $\Theta \Vdash^{\mathrm{cbv}} \check{V} : \nu \fatsemi \Xi$ for some Θ and Ξ. We have $\Theta \Vdash^{\mathrm{cbv}} \check{V} : \downarrow\uparrow\nu \parallel \Xi$.
  - Case $\check{v} = \mu\alpha.\check{c}$: By the induction hypothesis, $\check{c} : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$ for some Θ and Ξ. We can assume without loss of generality that $\Xi = (\Xi', \alpha : \varphi)$ (otherwise consider $\Xi, \alpha : \uparrow\downarrow a$). Then $\Theta \Vdash^{\mathrm{cbv}} \mu\alpha.\check{c} : \downarrow\varphi \parallel \Xi'$.
- $\check{c} : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$
  - Case $\check{c} = \langle x \mid \check{v}' \cdot \check{e}' \rangle$: By the induction hypothesis, $\Theta_1 \Vdash^{\mathrm{cbv}} \check{v}' : \downarrow\varphi_1 \parallel \Xi_1$ and $\Theta_2 \parallel \check{e}' : \varphi_2 \Vdash^{\mathrm{cbv}} \Xi_2$ for some Θ_1, Θ_2, φ_1, φ_2, Ξ_1, and Ξ_2. Since φ_1 is $\{\top, \bot\}$-free, it must be the case that $\varphi_1 = \bigvee_{i \in I} \uparrow\nu_i$ (up to the equivalence induced by the subtyping relation). Then we have $\Theta_1 \wedge \Theta_2 \parallel \check{v}' \cdot \check{e}' \mid \uparrow(\bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2))) \Vdash^{\mathrm{cbv}} \Xi_1 \vee \Xi_2$. Now
  $$\langle x \mid \check{v}' \cdot \check{e}' \rangle : (\Theta_1 \wedge \Theta_2 \wedge (x : \bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2))) \Vdash^{\mathrm{cbv}} \Xi_1 \vee \Xi_2)$$
  because $\Theta_1 \wedge \Theta_2 \wedge (x : \bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2))) \Vdash^{\mathrm{cbv}} x : \bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2)) \fatsemi \Xi_1 \vee \Xi_2$ and thus $\Theta_1 \wedge \Theta_2 \wedge (x : \bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2))) \Vdash^{\mathrm{cbv}} x : \downarrow\uparrow(\bigwedge_{i \in I}(\downarrow(\nu_i \to \varphi_2))) \parallel \Xi_1 \vee \Xi_2$.
  - Case $\check{c} = \langle \check{V} \mid \alpha \rangle$: By the induction hypothesis, $\Theta \Vdash^{\mathrm{cbv}} \check{V} : \downarrow\varphi \parallel \Xi$. Hence $\langle \check{V} \mid \alpha \rangle : (\Theta \Vdash^{\mathrm{cbv}} \Xi \vee (\alpha : \varphi))$.
- $\Theta \parallel \check{e} : \varphi \Vdash^{\mathrm{cbv}} \Xi$
  - Case $\check{e} = \alpha$: Similar to the case $\check{V} = x$.
  - Case $\check{e} = \check{v}' \cdot \check{e}'$: Similar to the argument in the case $\check{c} = \langle x \mid \check{v} \cdot \check{e} \rangle$.
  - Case $\check{e} = \tilde{\mu}.\check{c}$: Similar to the case $\check{v} = \mu\alpha.\check{c}$.

□

The $\{\top, \bot\}$-free subsystem also enjoys De-substitution if the substituted variable has a free occurrence.

Lemma 24. *For substitution of values, we have the following (where we assume that x is a term variable not appearing in Θ).*

- *If* $\Theta \Vdash^{\mathrm{cbv}} V'[V/x] : \nu' \fatsemi \Xi$ *and* V' *has a free occurrence of* x*, there exists* ν *such that* $\Theta, x : \nu \Vdash^{\mathrm{cbv}} V' : \nu' \fatsemi \Xi$ *and* $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$*.*
- *If* $\Theta \Vdash^{\mathrm{cbv}} v[V/x] : \varrho \parallel \Xi$ *and* v *has a free occurrence of* x*, there exists* ν *such that* $\Theta, x : \nu \Vdash^{\mathrm{cbv}} v : \varrho \parallel \Xi$ *and* $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$*.*
- *If* $c[V/x] : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$ *and* c *has a free occurrence of* x*, there exists* ν *such that* $c : (\Theta, x : \nu \Vdash^{\mathrm{cbv}} \Xi)$ *and* $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$*.*
- *If* $\Theta \parallel e[V/x] : \varphi \Vdash^{\mathrm{cbv}} \Xi$*, there exists* ν *such that* $\Theta, x : \nu \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$ *and* $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$*.*

For substitution of co-terms, we have the following (where we assume that α *is a co-term variable not appearing in* Ξ*).*

- *If* $\Theta \Vdash^{\mathrm{cbv}} V[e/\alpha] : \nu \fatsemi \Xi$ *and* V *has a free occurrence of* α*, there exists* φ *such that* $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \alpha : \varphi, \Xi$ *and* $\Theta \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$*.*
- *If* $\Theta \Vdash^{\mathrm{cbv}} v[e/\alpha] : \varrho \parallel \Xi$ *and* v *has a free occurrence of* α*, there exists* φ *such that* $\Theta \Vdash^{\mathrm{cbv}} v : \varrho \parallel \alpha : \varphi, \Xi$ *and* $\Theta \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$*.*
- *If* $c[e/\alpha] : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$ *and* c *has a free occurrence of* α*, there exists* φ *such that* $c : (\Theta \Vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)$ *and* $\Theta \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$*.*
- *If* $\Theta \parallel e'[e/\alpha] : \varphi' \Vdash^{\mathrm{cbv}} \Xi$ *and* e' *has a free occurrence of* α*, there exists* φ *such that* $\Theta \parallel e' : \varphi' \Vdash^{\mathrm{cbv}} \Xi$ *and* $\Theta \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$*.*

Proof. By induction on the structure of derivations. We prove the claims for substitution of values. The claims about Substitution of co-terms can be proved similarly.

We first prove the case in which V' is a variable.

- Case $V' = x$: Let $\nu = \nu'$.
- Case $V' = y \neq x$: This contradicts to the assumption that x has a free occurrence.

For other cases, we do case analysis on the last rule used in the derivation.

- Case (CH-VAbs): Then $\nu' = \downarrow(\nu'' \to \varphi)$, $V' = \lambda y.v'$ (where y is a fresh variable not appearing in V), and $\Theta, y : \nu'' \Vdash^{\mathrm{cbv}} v' : \downarrow\varphi \parallel \Xi$. By the induction hypothesis, we have ν such that $\Theta, y : \nu'', x : \nu \Vdash^{\mathrm{cbv}} v' : \downarrow\varphi \parallel \Xi$ and $\Theta, y : \nu'' \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$. The latter can be strengthened to $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$ since y is fresh. We have $\Theta, x : \nu \Vdash^{\mathrm{cbv}} \lambda y.v' : \downarrow(\nu'' \to \varphi) \fatsemi \Xi$ as desired.
- Case (CH-VL): Then $\varrho = \downarrow\uparrow\nu'$ and $\Theta \Vdash^{\mathrm{cbv}} V'[V/x] : \nu' \fatsemi \Xi$. By the induction hypothesis, there exists ν such that $\Theta, x : \nu \Vdash^{\mathrm{cbv}} V' : \nu' \fatsemi \Xi$ and $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$. We obtain $\Theta, x : \nu \Vdash^{\mathrm{cbv}} V' : \downarrow\uparrow\nu' \parallel \Xi$ from the former judgement.
- (CH-TMu): Then $\varrho = \downarrow\varphi$, $v = \mu\alpha.c$ (where α is a fresh variable not appearing in V), and $c[V/x] : (\Theta \Vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)$. By the induction hypothesis, we have ν such that $c : (\Theta, x : \nu \Vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)$ and $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \alpha : \varphi, \Xi$. The latter can be strengthened to $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$ since α is fresh. We have $\Theta, x : \nu \Vdash^{\mathrm{cbv}} \mu\alpha.c : \downarrow\varphi \parallel \Xi$ as desired.
- (CH-CVar): This contradicts to the assumption that x has a free occurrence.

– (CH-CMu): Similar to the case (CH-TMu).
– (CH-CApp): Then $\varphi = \uparrow(\bigvee_{i\in I}(\downarrow(\nu'_i \to \varphi')))$, $e = v' \cdot e'$, $\Theta \Vdash^{\mathrm{cbv}} v' : \downarrow(\bigvee_{i\in I}(\uparrow\nu'_i)) \parallel \Xi$, and $\Theta \parallel e' : \varphi' \Vdash^{\mathrm{cbv}} \Xi$.
  - When both v' and e' have a free occurrence of x: By the induction hypothesis, we have ν_1 and ν_2 such that
    * $\Theta, x : \nu_1 \Vdash^{\mathrm{cbv}} v' : \downarrow(\bigvee_{i\in I}(\uparrow\nu'_i)) \parallel \Xi$,
    * $\Theta \Vdash^{\mathrm{cbv}} V : \nu_1 \fatsemi \Xi$,
    * $\Theta, x : \nu_2 \parallel e' : \varphi' \Vdash^{\mathrm{cbv}} \Xi$, and
    * $\Theta \Vdash^{\mathrm{cbv}} V : \nu_2 \fatsemi \Xi$.

    Then $\Theta, x : \nu_1 \wedge \nu_2 \parallel v' \cdot e' : \uparrow(\bigvee_{i\in I}(\downarrow(\nu'_i \to \varphi'))) \Vdash^{\mathrm{cbv}} \Xi$ and $\Theta \Vdash^{\mathrm{cbv}} V : \nu_1 \wedge \nu_2 \fatsemi \Xi$.
  - When v' has a free occurrence of x but e' does not: By the induction hypothesis, there exists ν such that $\Theta, x : \nu \Vdash^{\mathrm{cbv}} v' : \downarrow(\bigvee_{i\in I}(\uparrow\nu'_i)) \parallel \Xi$ and $\Theta \Vdash^{\mathrm{cbv}} V : \nu \fatsemi \Xi$. Since $e'[V/x] = e'$, we have $\Theta, x : \nu \parallel e' : \varphi' \Vdash^{\mathrm{cbv}} \Xi$. Hence $\Theta, x : \nu \parallel v' \cdot e' : \uparrow(\bigvee_{i\in I}(\downarrow(\nu'_i \to \varphi'))) \Vdash^{\mathrm{cbv}} \Xi$.
  - When e' has a free occurrence of x but v' does not: Similar to the above subcase.
  - When neither v' nor e' does not have a free occurrence of x: This contradicts to the assumption that x has a free occurrence in $v' \cdot e'$.
– (CH-COM): Similar to the above case.

□

We prove the completeness side of Theorem 4 by induction on the length of the maximum reduction sequences.

If the expression is in normal form, then it is in a canonical form by Lemma 22 and typable in the $\{\top, \bot\}$-free subsystem by Lemma 23.

Suppose that the expression is reducible. We use induction on the structure of the derivation of the reduction relation. We prove the base cases. The other cases are trivial.

– Case $\langle \mu\alpha.c \mid e\rangle \longrightarrow c[e/\alpha]$: By the induction hypothesis, $c[e/\alpha]$ is typable in the $\{\top, \bot\}$-free subsystem. Suppose that $c[e\alpha] : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$.
  Suppose that c has a free occurrence of α. Then by Lemma 24, there exists φ such that $c : (\Theta \Vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)$ and $\Theta \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi$. Then $\Theta \Vdash^{\mathrm{cbv}} \mu\alpha.c : \downarrow\varphi \parallel \Xi$. Hence $\langle \mu\alpha.c \mid e\rangle : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$.
  Suppose that c does not have a free occurrence of α. Since $\langle \mu\alpha.c \mid e\rangle$ is strongly normalising, so is e. If $e \longrightarrow^* e'$ is the longest sequence from e, then $\langle \mu\alpha.c \mid e\rangle \longrightarrow^* \langle \mu\alpha.c \mid e'\rangle \longrightarrow c$ is a longer reduction sequence from $\langle \mu\alpha.c \mid\rangle$. By the induction hypothesis, e is typable in the subsystem, say $\Theta' \parallel e : \varphi \Vdash^{\mathrm{cbv}} \Xi'$. Then $c : (\Theta \Vdash^{\mathrm{cbv}} \alpha : \varphi, \Xi)$ since α is not free in c, and thus $\Theta \Vdash^{\mathrm{cbv}} \mu\alpha.c : \downarrow\varphi \parallel \Xi$. Then we have $\langle \mu\alpha.c \mid e\rangle : (\Theta \wedge \Theta' \Vdash^{\mathrm{cbv}} \Xi \vee \Xi')$.
– Case $\langle V \mid \tilde{\mu}x.c\rangle \longrightarrow c[V/x]$: Similar to the above case.
– Case $\langle \lambda x.v \mid v' \cdot e\rangle \longrightarrow \langle v' \mid \tilde{\mu}x.\langle v \mid e\rangle\rangle$ (where x is not free in e): By the induction hypothesis, $\langle v' \mid \tilde{\mu}x.\langle v \mid e\rangle\rangle$ is typable in the $\{\top, \bot\}$-free subsystem. Suppose that $\langle v' \mid \tilde{\mu}x.\langle v \mid e\rangle\rangle : (\Theta \Vdash^{\mathrm{cbv}} \Xi)$. Then $\Theta \Vdash^{\mathrm{cbv}} v' : \downarrow\varphi \parallel \Xi$ and $\Theta \parallel \tilde{\mu}x.\langle v \mid e\rangle : \varphi \Vdash^{\mathrm{cbv}} \Xi$. We appeal to the inversion of this judgement described as follows:

If $\Theta \parallel \tilde{x}.c' : \varphi \Vdash^{\mathrm{cbv}} \Xi$, there exists a finite family of types $\{\uparrow\nu_i\}_{i\in I}$ such that $\varphi \preceq \bigvee_{i\in I} \uparrow\nu_i$ and $c' : (\Theta, x : \nu_i \Vdash^{\mathrm{cbv}} \Xi)$ for every $i \in I$.

This claim can be proved by induction on the structure of the derivation, similar to the proof of Lemma 14. By this claim, we have a finite family of types $\{\uparrow\nu_i\}_{i\in I}$ such that $\varphi \preceq \bigvee_{i\in I} \uparrow\nu_i$ and $\langle v \mid e\rangle : (\Theta, x : \nu_i \Vdash^{\mathrm{cbv}} \Xi)$. Hence, for every $i \in I$, there exists φ'_i such that $\Theta, x : \nu_i \Vdash^{\mathrm{cbv}} v : \downarrow\varphi'_i \parallel \Xi$ and $\Theta, x : \nu_i \parallel e : \varphi'_i \Vdash^{\mathrm{cbv}} \Xi$, which can be strengthened to $\Theta \parallel e : \varphi'_i \Vdash^{\mathrm{cbv}} \Xi$ since x is not free in e. Let $\varphi'' = \bigvee_{i\in I} \varphi'_i$. Then $\Theta, x : \nu_i \Vdash^{\mathrm{cbv}} v : \downarrow\varphi'' \parallel \Xi$ for every $i \in I$ and

$$\Theta \parallel e : \varphi'' \Vdash^{\mathrm{cbv}} \Xi.$$

From the former judgement, $\Theta \Vdash^{\mathrm{cbv}} \lambda x.v : \downarrow(\nu_i \to \varphi'') \fatsemi \Xi$ for every $i \in I$, and thus $\Theta \Vdash^{\mathrm{cbv}} \lambda x.v : \bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi'')) \fatsemi \Xi$. So we have

$$\Theta \Vdash^{\mathrm{cbv}} \lambda x.v : \downarrow\uparrow(\bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi''))) \mid \Xi.$$

Since $\varphi \preceq \bigvee_{i\in I} \uparrow\nu_i$ and $\Theta \Vdash^{\mathrm{cbv}} v' : \downarrow\varphi \parallel \Xi$, we have

$$\Theta \Vdash^{\mathrm{cbv}} v' : \downarrow(\bigvee_{i\in I}(\uparrow\nu_i)) \parallel \Xi.$$

By (CH-CApp), we obtain

$$\Theta \parallel v' \cdot e : \uparrow(\bigwedge_{i\in I}(\downarrow(\nu_i \to \varphi''))) \Vdash^{\mathrm{cbv}} \Xi.$$

So

$$\langle \lambda x.v \mid v' \cdot e\rangle : (\Theta \Vdash^{\mathrm{cbv}} \Xi).$$