Completeness and Partial Soundness Results for Intersection & Union Typing for $\overline{\lambda}\mu\tilde{\mu}$*

(Annals of Pure and Applied Logic 161, pp 1400-1430, 2010)

Steffen van Bakel

Department of Computing, Imperial College London,
180 Queen's Gate, London SW7 2BZ, UK
svb@doc.ic.ac.uk

Abstract

This paper studies intersection and union type assignment for the calculus $\overline{\lambda}\mu\tilde{\mu}$ [16], a proof-term syntax for Gentzen's classical sequent calculus, with the aim of defining a type-based semantics, via setting up a system that is closed under conversion.
We will start by investigating what the minimal requirements are for a system for $\overline{\lambda}\mu\tilde{\mu}$ to be complete (closed under redex expansion); this coincides with System $\mathcal{M}^{\cap\cup}$, the notion defined in [18]; however, we show that this system is not sound (closed under subject reduction), so our goal cannot be achieved. We will then show that System $\mathcal{M}^{\cap\cup}$ is also not complete, but can recover from this by presenting System $\mathcal{M}^{c}$ as an extension of $\mathcal{M}^{\cap\cup}$ (by adding typing rules) and showing that it satisfies completeness; it still lacks soundness. We show how to restrict $\mathcal{M}^{\cap\cup}$ so that it satisfies soundness as well by limiting the applicability of certain type assignment rules, but only when limiting reduction to (confluent) call-by-name or call-by-value reduction; in restricting the system this way, we sacrifice completeness. These results combined show that, with respect to full reduction, it is not possible to define a sound and complete intersection and union type assignment system for $\overline{\lambda}\mu\tilde{\mu}$.

keywords: classical logic, sequent calculus, intersection and union types, soundness, completeness.

Introduction

The role of and the attention to Classical Logic in computer science are changing drastically over the last few years. Given the direct relation between the (typed) λ-calculus [14, 10] and intuitionistic logic, for many years it was believed that only the *constructive* logics had any real computational content, and only after Griffin's discovery of the relation between double-negation elimination [25] and Felleisen's control operators [22] did the research community become aware of the computational advantages of Classical Logic.

There are two main directions in doing proof theory: *sequent calculi* and *natural deduction systems*, both introduced by Gentzen in [23, 24]. On the one hand, the *Sequent Calculus* LK is a logical system in which the rules only introduce connectives (but on either side of a sequent); on the other hand, *Natural Deduction* uses rules that introduce or eliminate connectives in the logical formulae. Natural deduction normally derives statements with a single conclusion, whereas LK allows for multiple conclusions, deriving sequents of the form $A_1, \ldots, A_n \vdash B_1, \ldots, B_m$, where $A_1, \ldots, A_n$ is to be understood as $A_1 \wedge \ldots \wedge A_n$ and $B_1, \ldots, B_m$ is to be understood as $B_1 \vee \ldots \vee B_m$.

* Small mistakes have been corrected here, as well as the proof of Theorem 9.8.

Exploring Classical Logic for Computation, research has focussed on different calculi, trying to exploit the Curry-Howard isomorphism (correspondence) for various classical logics, both in sequent style and in natural deduction. In this paper we contribute to that line of research by studying Curien and Herbelin's calculus $\overline{\lambda}\mu\tilde{\mu}$-calculus [16], which enjoys a Curry-Howard isomorphism for a variant (with *focus*, or *active formulae à la* Parigot's $\lambda\mu$ [38, 39]) of the implicative variant of Kleene's G_3 [33], itself a variant of Gentzen's LK.

The research that forms the context of this paper is vast; we mention the $\lambda\mu$-calculus introduced by Parigot, a confluent, natural deduction-based system; to achieve confluence, it has a limited use of negation and does not allow all possible reductions. $\lambda\mu$ has been studied in depth by various authors, like Ong & Stewart who studied call-by-value reduction [37], Bierman who defined an abstract machine for $\lambda\mu$ [12], de Groote who studied how to express control structures in $\lambda\mu$ [26], just to mention a few. On the other hand, more recently attention has moved to sequent calculi. Using $\lambda\mu$'s notion of *active* formula, Herbelin [28] and Curien [16, 29] developed the sequent calculus $\overline{\lambda}\mu\tilde{\mu}$; Wadler defined the Dual Calculus [46]. Urban defined a calculus that manipulates proofs in Gentzen's LK [44, 45], using a notion of *activated cut* to accomplish strong normalisation; this later led to the definition of the calculus $\mathcal{X}$ by Lengrand [35], and van Bakel and Lescanne [7, 8]; principal types for implicative $\mathcal{X}$ and its relation with an ML-like polymorphic extension of the λ-calculus are studied in [42, 43].

One of the great advantages of using classical logic is the capture of not only parameter passing, but also context management: for programs based on full classical logic, a parameter call (fetching an operand for a procedure) is truly dual to a context call (exiting from / aborting a computation), and a program and its context are treated on equal footing. This gives rise to far richer fully typed programming paradigms than the normal functional one, which need to be fully investigated and exploited. This paper tries to contribute to this line of research, by attempting to set up filter semantics through the definition a notion of intersection-union typing for $\overline{\lambda}\mu\tilde{\mu}$ and to study its properties. We will see that this attempt fails: our system does not fulfil the semantic requirement, in that we cannot define a notion that is both *sound* (*i.e.* closed under reduction) *and complete* (*i.e.* closed under redex expansion, the reverse of reduction).

Filter semantics and a filter model were first defined for the λ-calculus by Barendregt, Coppo and Dezani-Ciancaglini in [11]; they defined a notion of intersection type assignment (introduced by Coppo and Dezani-Ciancaglini [15] and Sallé [41]), and showed that a model can be created through the interpretation of terms by their assignable types. Intersection type assignment for the λ-calculus adds a new type constructor $\cap$ and a type constant $\top$[1]; using intersection it is possible to express that a term can have a number of different (perhaps even incompatible, non-unifiable) types, and $\top$ is the universal type, *i.e.* all terms have type $\top$. Although this extension is conceptually simple, it is, in fact, a very powerful characterisation and semantic tool, since all the following properties have been shown to hold for a number of different systems:

- If $\Gamma \vdash M : A$ and $M =_\beta N$, then $\Gamma \vdash N : A$.
- $\Gamma \vdash M : A$ and $A \neq \top$, iff M has a head-normal form.
- $\Gamma \vdash M : A$ and $\top$ does not occur in Γ and A, iff M has a normal form.
- $\Gamma \vdash M : A$ and $\top$ is not used at all in this derivation, iff M is strongly normalising.
- Intersection type systems have the principal type property.
- $[\![M]\!] = \{A \mid \exists \Gamma \ [\Gamma \vdash M : A]\}$ gives a (filter) λ-model.
- If $\Gamma \vdash M : A$ and $M \to_\eta N$, then $\Gamma \vdash N : A$ (this property needs a contra-variant $\leq$-relation,

[1] Normally called ω; here we reserve Greek characters for context variables.

which is not present in all systems).

A natural question to ask now is: "Can we achieve the same for $\overline{\lambda}\mu\tilde{\mu}$?"; to answer this question, this paper studies the addition of intersection types to the system for $\overline{\lambda}\mu\tilde{\mu}$; union types are added for reasons of symmetry.

The system we define in this paper is set up to be a conservative extension of Krivine's *Système* Dω of intersection type assignment for the λ-calculus [34], in that λ-terms typeable in that system translate to $\overline{\lambda}\mu\tilde{\mu}$-terms, while preserving the type. There are many different notions of intersection type assignment in existence (see also [11, 1, 2, 5]), that, in the context of the λ-calculus more or less coincide; the most important difference is normally the language of types (full BCD [11, 34], or strict types [1, 2]) and the availability of a contra-variant $\leq$-relation (as in [11, 2], or not [34, 1]). Surprisingly, this is no longer true when bringing intersection types (and union) to the context of sequent calculi; BCD-types are needed, as will be shown in this paper.

Perhaps the strongest of the above results is the characterisation of strong normalisation, which states that, in a system without the type constant $\top$, the typeable terms are exactly the strongly normalisable ones [40, 1]. This has since then been achieved in many ways for different calculi, and in order to come to a similar characterisation for the (untyped) sequent calculus $\overline{\lambda}\mu\tilde{\mu}$, Dougherty, Ghilezan and Lescanne presented System $\mathcal{M}^{\cap\cup}$ [18], that defines a notion of intersection and union typing for that calculus. With our eye on the definition of semantics, in this paper we revisit System $\mathcal{M}^{\cap\cup}$, adding $\top$ as the maximal and $\bot$ as the minimal type, and extending the set of derivation rules for the purpose of *completeness* (the property that types are preserved also going backwards with respect to reduction).

The notion of typing (*i.e.* environment assignment) we present here will be shown to be the natural system, in that intersection, $\top$, union, and $\bot$ play their expected roles for completeness, our first step towards the construction of a filter model. However, we will show that completeness does not hold directly for $\mathcal{M}^{\cap\cup}$, and that the system needs to be generalised first. As was already mentioned in [19], also soundness does not hold for $\mathcal{M}^{\cap\cup}$, and we will argue that this is mainly caused by the non-logical foundation (*i.e.* typeable terms no longer correspond to proofs) of both intersection and union; this was also observed for intersection type assignment for the λ-calculus by Hindley [31]. This failure was the motivation for the restriction made to come to System $\mathcal{M}^{\cap}$ as presented in [19]; as we will show in Section 8, this was not enough; also, the completeness problem was not picked up on, as we will show as well. Since $\mathcal{M}^{\cap\cup}$ is built using the minimal requirements for completeness, this implies that a sound and complete system with respect to $\overline{\lambda}\mu\tilde{\mu}$'s full reduction cannot be defined.

We will show that we can partially recover from these failures by restricting to either call-by-name or call-by-value reduction, but that we also need to restrict the applicability of either union or intersection assignment. Since these restrictions regard System $\mathcal{M}^{\cap\cup}$, which does not satisfy the completeness result, the resulting systems are not suitable to define semantics.

In this paper we will show that the –at the time– surprising loss of soundness for the system with intersection and union types for the λ-calculus in [9] is, in fact, natural and inevitable. Also, working with intersection and union in the context of these highly symmetric sequent calculi makes clear that these are truly dual; we will show that it is not union alone that causes problems, but that the problem is much more profound, and also arises when dealing with intersection. Although the idea behind both intersection and union might be (the logical) *and* and *or*, the fact that they are both *not* logical destroys the soundness, both for a system based on intersection, as for a system based on union. This also explains why, for ML with side-effects [27, 47, 36], quantification is no longer sound: also the $(\forall I)$ and $(\forall E)$ rules of ML are not logical (see Example 10.1). This problem also appears when using intersection and union

types in an operational setting [17, 21].

A number of variants exist for Gentzen's calculus for classical logic LK; the variant of LK we will consider in this paper, and that lies at the basis of the calculus $\overline{\lambda}\mu\tilde{\mu}$, is the system known as Kleene's G_3, which is defined by:

$$(Ax): \frac{}{\Gamma, A \vdash A, \Delta} \qquad (cut): \frac{\Gamma \vdash A, \Delta \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}$$

$$(\to R): \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta} \qquad (\to L): \frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}$$

It has no structural rules, *i.e.* has implicit weakening and contraction. The rules only introduce connectives (but on both sides of a sequent), in contrast to natural deduction which uses introduction and elimination rules. The only way to eliminate a connective is to eliminate the whole formula in which it appears, via an application of the (cut)-rule.

In this paper we will treat $\overline{\lambda}\mu\tilde{\mu}$ as a pure, untyped calculus, and ignore its origin as a proof calculus in that we study various notions of sequent-style intersection-union typing ($\mathcal{M}^{\cap\cup}$, $\mathcal{M}^{\mathrm{c}}$, $\mathcal{M}^{\mathrm{N}}$, and $\mathcal{M}^{\mathrm{v}}$) for it, that are natural extensions of the system considered for $\overline{\lambda}\mu\tilde{\mu}$ in [16], *i.e.* the basic implicative system for Classical Logic.

The main result of this paper is that it presents a notion of type assignment $\mathcal{M}^{\mathrm{c}}$ that is closed under redex expansion, and two restrictions or sub-sytems, $\mathcal{M}^{\mathrm{N}}$ and $\mathcal{M}^{\mathrm{v}}$, that are closed under either CBN or CBV-reduction; for CBV we limit $(\cap R)$ to *values*, and for CBN limit $(\cup L)$ to *slots* (see Definition 1.4). Our solutions are crucially different from any other published in the past (in, for example, [32, 9, 17]), in that we do not limit the syntax of types at all (as in [20, 19]), and pose completely different side-conditions on rules (with respect to [29]) in fact generalising the there claimed result. As far as we know, these are the first correct presentations of sound restrictions of a fully expressive system with intersection and union types for $\overline{\lambda}\mu\tilde{\mu}$.

However, our findings show that it is not possible to define a notion of typing that is closed under *conversion*; we have not reached a conclusion yet for the restriction to CBV or CBN. So we cannot construct a filter model using types for full $\overline{\lambda}\mu\tilde{\mu}$ using this system.

Outline of this paper

This paper is constructed as follows: in Section 1 we present the $\overline{\lambda}\mu\tilde{\mu}$-calculus, and encode the λ-calculus into it. In Section 3, we will see what are the minimal requirements for a complete extension of the basic type assignment system for $\overline{\lambda}\mu\tilde{\mu}$, and see in Section 4 that it coincides with System $\mathcal{M}$, a slight variant of the intersection-union typing system $\mathcal{M}^{\cap\cup}$ of [18]. We show that this is not a conservative extension of Krivine's *Système Dω* for the λ-calculus that we present in Section 2. In Section 5, for System $\mathcal{M}$, we show soundness for reduction rule , and will show that contraction via rules $\to_\mu$ and $\to_{\tilde{\mu}}$ is only sound in certain circumstances. In Section 6 we will give a counter example for completeness, and formulate the missing rules to fix this problem. In Section 7 we will show that also soundness fails in general; we will show that these failures are fundamental, and cannot be resolved by adding rules, but only by restricting rules: since the failure of soundness means that reduction brings us from a typeable term to an untypeable one, we fix this by forcing untypeability also for the first term. This will be followed in Section 8 by a discussion of System $\mathcal{M}^{\cap}$ of [19]; we will show also here both soundness and completeness fail, albeit for different reasons.

We will present a modified system $\mathcal{M}^{\mathrm{c}}$ – an extension of $\mathcal{M}^{\cap\cup}$ – for which we can show completeness in Section 9, and show also that types assignable in $D\omega$ are preserved by the interpretation of λ-terms; because $\mathcal{M}^{\mathrm{c}}$ is an extension of $\mathcal{M}^{\cap\cup}$, soundness will still fail. In

Section 10, we will define two restrictions of $\mathcal{M}^{\cap\cup}$, being $\mathcal{M}^{\mathrm{N}}$ and $\mathcal{M}^{\mathrm{V}}$, and will show that call-by-name and call-by-value, respectively, are sound for these notions; completeness will fail.

1 The calculus $\overline{\lambda}\mu\tilde{\mu}$

In its typed version, $\overline{\lambda}\mu\tilde{\mu}$ is a proof-term syntax for a classical sequent calculus. As in $\lambda\mu$, for $\overline{\lambda}\mu\tilde{\mu}$ there are two sets of variables: x,y,z, *etc.*, label the types of the hypotheses and α,β,γ, *etc.*, label the types of the conclusions. Moreover, the syntax of $\overline{\lambda}\mu\tilde{\mu}$ has three different categories: commands, terms, and contexts (or co-terms); we use *expressions* as a generic term for these three categories. Correspondingly, they are typed by three kinds of sequents: the usual sequents $\Gamma \vdash \Delta$ type commands, while the sequents typing terms (resp. contexts) are of the form $\Gamma \vdash A \mid \Delta$ (resp. $\Gamma \mid A \vdash \Delta$), marking the conclusion (resp. hypothesis) A as *active*.

Definition 1.1 (Commands, Terms, and Contexts [16]) There are three categories of expressions in $\overline{\lambda}\mu\tilde{\mu}$, defined by:

$$\begin{array}{lll} c & ::= & \langle v|e\rangle \qquad\qquad (\textit{commands}) \\ v & ::= & x \mid \lambda x.v \mid \mu\beta.c \qquad (\textit{terms}) \\ e & ::= & \alpha \mid v\cdot e \mid \tilde{\mu}x.c \qquad (\textit{contexts}) \end{array}$$

Here λ, μ, and $\tilde{\mu}$are binders, and the notion of free or bound term and context variables is defined as usual.

With conventional notations about contexts (*i.e.* seeing contexts as terms with a hole), $v\cdot e$ can be thought of as $e[[\,]\,v]$, and the context $v_1\cdot(\cdots(v_n\cdot\alpha)\cdots)$ (we can omit these brackets and write $v_1\cdots v_n\cdot\alpha$) as a *stack* (see Example 2.5); $\mu\alpha.c$ is inherited from $\lambda\mu$, as is $\langle v|\alpha\rangle$ which corresponds to $\lambda\mu$'s *naming* construct $[\alpha]v$, giving name α to the implicit output name of v; the construct $\tilde{\mu}x.c$ can be thought of as $\mathsf{let}\ x = [\,]\ \mathsf{in}\ c$.

Commands can be computed (thus eliminating the cut in the corresponding proof):

Definition 1.2 (Reduction in $\overline{\lambda}\mu\tilde{\mu}$ [16, 29]) Let $c\{e/\beta\}$ stand for the implicit substitution of the free occurrences of the context variable β by the context e, and $c\{v/x\}$ for that of x by the term v. The reduction rules are defined by:

	logical rules				*extensional rules*		
$(\to)$:	$\langle\lambda x.v_1\|v_2\cdot e\rangle$	$\to$	$\langle v_2\,\|\,\tilde{\mu}x.\langle v_1\|e\rangle\rangle$	(η):	$\lambda x.\mu\beta.\langle v\|x\cdot\beta\rangle$	$\to$ v	$x,\beta\notin fv(v)$
(μ):	$\langle\mu\beta.c\|e\rangle$	$\to$	$c\{e/\beta\}$	$(\eta\mu)$:	$\mu\alpha.\langle v\|\alpha\rangle$	$\to$ v	$\alpha\notin fv(v)$
$(\tilde{\mu})$:	$\langle v\|\tilde{\mu}x.c\rangle$	$\to$	$c\{v/x\}$	$(\eta\tilde{\mu})$:	$\tilde{\mu}x.\langle x\|e\rangle$	$\to$ e	$x\notin fv(e)$

Notice that rules $(\to)$, (μ), and $(\tilde{\mu})$ reduce commands to commands, rules (η) and $(\eta\mu)$ reduce a term to a term, and rule $(\eta\tilde{\mu})$ reduces a context to a context. Apart from Lemma 2.3, the extensional rules play no role in this paper. Not all commands can be reduced: *e.g.*, $\langle x|\alpha\rangle$, $\langle\lambda x.v|\alpha\rangle$ and $\langle x|v\cdot e\rangle$ are irreducible cuts; this is one of the differences between lk and $\overline{\lambda}\mu\tilde{\mu}$.

(Implicative) Typing for $\overline{\lambda}\mu\tilde{\mu}$ is defined by:

Definition 1.3 (Typing for $\overline{\lambda}\mu\tilde{\mu}$ [16]) Let $\mathcal{V}$ be a countable (infinite) set of type-variables, ranged over by φ. Types are defined by the grammar

$$A,B \ ::= \ \varphi \mid (A\to B)$$

As usual, we omit right-most, outer-most parentheses.

Type assignment is defined via the rules:

$$(cut): \frac{\Gamma \vdash t : A \mid \Delta \quad \Gamma \mid e : A \vdash \Delta}{\langle t|e\rangle : \Gamma \vdash \Delta}$$

$$(Ax\text{-}R): \frac{}{\Gamma, x{:}A \vdash x : A \mid \Delta} \qquad (Ax\text{-}L): \frac{}{\Gamma \mid \alpha : A \vdash \alpha{:}A, \Delta}$$

$$(\to R): \frac{\Gamma, x{:}A \vdash t : B \mid \Delta}{\Gamma \vdash \Lambda x.t : A \to B \mid \Delta} \qquad (\to L): \frac{\Gamma \vdash t : A \mid \Delta \quad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid t{\cdot}e : A \to B \vdash \Delta}$$

$$(\mu): \frac{c : \Gamma \vdash \alpha{:}A, \Delta}{\Gamma \vdash \mu\alpha.c : A \mid \Delta} \qquad (\tilde{\mu}): \frac{c : \Gamma, x{:}A \vdash \Delta}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

We write $c : \Gamma \vdash_{\overline{\lambda}} \Delta$ ($\Gamma \vdash_{\overline{\lambda}} v : A \mid \Delta$, or $\Gamma \mid e : A \vdash_{\overline{\lambda}} \Delta$) if there exists a derivation built using these rules that has this judgement in the bottom line, and write $\mathcal{D} :: c : \Gamma \vdash_{\overline{\lambda}} \Delta$ *etc.* if we want to name the derivation.

We will write, for example, $\Gamma \vdash_{\overline{\lambda}} v : A \mid$ for $\Gamma \vdash_{\overline{\lambda}} v : A \mid \emptyset$.

$\overline{\lambda}\mu\tilde{\mu}$ has a critical pair in the command $\langle \mu\alpha.c_1 \,|\, \tilde{\mu}x.c_2\rangle$, which reduces to both $c_1\{\tilde{\mu}x.c_2/\alpha\}$ and $c_2\{\mu\alpha.c_1/x\}$; since *cut*-elimination of the classical sequent calculus G_3 is not confluent, neither is reduction in $\overline{\lambda}\mu\tilde{\mu}$. For example, in LK the proof (where (W) is the admissible weakening rule)

$$\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}(W)} \quad \dfrac{\boxed{\mathcal{D}_2}}{\dfrac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}(W)}}{\Gamma \vdash \Delta}(cut)$$

reduces to both $\mathcal{D}_1$ and $\mathcal{D}_2$, different proofs, albeit for the same sequence; likewise, in $\vdash_{\overline{\lambda}}$ we can derive (where α does not appear in c_1, and x does not appear in c_2):

$$\dfrac{\dfrac{\dfrac{\boxed{}}{\dfrac{c_1 : \Gamma \vdash \Delta}{c_1 : \Gamma \vdash \alpha{:}A, \Delta}(W)}}{\Gamma \vdash \mu\alpha.c_1 : A \mid \Delta}(\mu) \quad \dfrac{\dfrac{\boxed{}}{\dfrac{c_2 : \Gamma \vdash \Delta}{c_2 : \Gamma, x{:}A \vdash \Delta}(W)}}{\Gamma \mid \tilde{\mu}x.c_2 : A \vdash \Delta}(\tilde{\mu})}{\langle \mu\alpha.c_1 \,|\, \tilde{\mu}x.c_2\rangle : \Gamma \vdash \Delta}(cut)$$

and $\langle \mu\alpha.c_1 \,|\, \tilde{\mu}x.c_2\rangle$ reduces to both c_1 and c_2.

Notice that, although $\overline{\lambda}\mu\tilde{\mu}$ has abstraction, it does not have application, which is natural since LK lacks *elimination rules*. In fact, abstraction's counterpart is that of *context construction* $v{\cdot}e$, where a term with a hole is built, offering the operand v and the continuation e. The main operators are μ and $\tilde{\mu}$abstraction, which, in a sense, respectively, correspond to (delayed) substitution (parameter call) and to context call.

Notice that $\overline{\lambda}\mu\tilde{\mu}$ has both *explicit* and *implicit* variables: the implicit variables are for example in $v{\cdot}e$, where the hole ($\cdot$, which acts as input) does not have an identity, and in $\lambda x.v$ where the context (output) is anonymous. We can make these variables explicit by *naming*, respectively, $\tilde{\mu}y.\langle y|v{\cdot}e\rangle$ and $\mu\alpha.\langle\lambda x.v|\alpha\rangle$; in case the variable y (α) does not occur in $v{\cdot}e$ ($\lambda x.v$), these terms are η redexes, but, in general, the implicit variable can be made to correspond to one that already occurs. The type assignment system below (Definition 4.5) is designed to be indifferent to these steps.

Herbelin's $\overline{\lambda}\mu\tilde{\mu}$-calculus expresses the duality of LK's left and right introduction in a very symmetric syntax. But the duality goes beyond that: for instance, the symmetry of the reduction rules display syntactically the duality between the call-by-value (CBV) and call-by-name

(CBN) evaluations (see also [46]). Indeed, the CBV reduction $\to_{\mathrm{v}}$ is obtained by forbidding a $\tilde{\mu}$-reduction when the redex is also a μ-redex, whereas the CBN reduction $\to_{\mathrm{N}}$ forbids a μ-reduction when the redex is also a $\tilde{\mu}$-redex.

Definition 1.4 (CBV AND CBN [16, 29]) *i*) *Values* V are defined by $V ::= x \mid \lambda x.v$, and *slots*[2] E are defined by $E ::= \alpha \mid v{\cdot}e$.

ii) CBV-reduction is defined by replacing rule $(\tilde{\mu})$ by: $(\tilde{\mu}_{\mathrm{v}}) : \langle V|\tilde{\mu}x.c\rangle \ \to\ c\{V/x\}$;

iii) CBN-reduction is defined by replacing rule (μ) by: $(\mu_{\mathrm{N}}) : \langle \mu\beta.c|E\rangle \ \to\ c\{E/\beta\}$.

2 *Système* $D\omega$ of Intersection Type Assignment for the λ-calculus

The remainder of this paper will be dedicated to the study of a notion of intersection typing on $\overline{\lambda}\mu\tilde{\mu}$. This will be defined as a natural extension of Krivine's *Système* $D\omega$ [34] of intersection type assignment for the λ-calculus.

Definition 2.1 (LAMBDA TERMS AND β-CONTRACTION [10]) *i*) The set Λ of *λ-terms* is defined by the syntax:

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

ii) The reduction relation $\to_\beta$ is defined as the contextual closure of the rule:

$$(\lambda x.M)N \ \to_\beta \ M\{N/x\}$$

$\twoheadrightarrow_\beta$ is the reflexive and transitive closure of $\to_\beta$, and $=_\beta$ is the equivalence relation generated by $\twoheadrightarrow_\beta$.

iii) Call-by-value reduction is defined by limiting the reduction rule $\to_\beta$ to contract only if the right-hand term is a *value*, *i.e.* is either a variable or an abstraction.

$$(\lambda x.M)V \ \to_\beta \ M\{V/x\}$$

Essentially following [16], an interpretation $[\![\cdot]\!]^{\mathrm{n}}_{\mu}$ of the λ-calculus into $\overline{\lambda}\mu\tilde{\mu}$ can be defined as follows:

Definition 2.2 Interpretation of the λ-calculus into $\overline{\lambda}\mu\tilde{\mu}$ via $[\![\cdot]\!]^{\mathrm{n}}_{\mu}$:

$$\begin{array}{rcl} [\![x]\!]^{\mathrm{n}}_{\mu} & \stackrel{\Delta}{=} & x \\ [\![\lambda x.M]\!]^{\mathrm{n}}_{\mu} & \stackrel{\Delta}{=} & \lambda x.\,[\![M]\!]^{\mathrm{n}}_{\mu} \\ [\![MN]\!]^{\mathrm{n}}_{\mu} & \stackrel{\Delta}{=} & \mu\alpha.\langle\, [\![M]\!]^{\mathrm{n}}_{\mu} \mid [\![N]\!]^{\mathrm{n}}_{\mu}{\cdot}\alpha\rangle \end{array}$$

We can even represent substitution explicitly (so interpret Bloo and Rose's $\lambda\mathbf{x}$ [13]), by adding

$$[\![M\langle x:=N\rangle]\!]^{\mathrm{n}}_{\mu} \ = \ \mu\alpha.\langle\, [\![N]\!]^{\mathrm{n}}_{\mu} \mid \tilde{\mu}x.\langle\, [\![M]\!]^{\mathrm{n}}_{\mu} \mid\alpha\rangle\rangle$$

Notice that λ-values are interpreted by $\overline{\lambda}\mu\tilde{\mu}$-values.

Correctness of this encoding is easy to prove:

Lemma 2.3 $[\![(\lambda x.M)N]\!]^{\mathrm{n}}_{\mu} \ \to \ [\![M]\!]^{\mathrm{n}}_{\mu}\,\{[\![N]\!]^{\mathrm{n}}_{\mu}/x\}$.

[2] In [29], slots are called *linear evaluation contexts*.

Proof: $\begin{array}{lcll} [\![(\lambda x.M)N]\!]_{\tilde{\mu}} & = & \mu\alpha.\langle\, [\![\lambda x.M]\!]_{\tilde{\mu}} \mid [\![N]\!]_{\tilde{\mu}} \cdot \alpha\rangle & = \quad \mu\alpha.\langle \lambda x.\, [\![M]\!]_{\tilde{\mu}} \mid [\![N]\!]_{\tilde{\mu}} \cdot \alpha\rangle \\ & \to & \mu\alpha.\langle\, [\![N]\!]_{\tilde{\mu}} \mid \tilde{\mu}x.\langle\, [\![M]\!]_{\tilde{\mu}} \mid \alpha\rangle\rangle & \to_{\tilde{\mu}} \ \mu\alpha.\langle\, [\![M]\!]_{\tilde{\mu}}\{\, [\![N]\!]_{\tilde{\mu}}/x\} \mid \alpha\rangle \\ & \to_{\eta\mu} & [\![M]\!]_{\tilde{\mu}}\{\, [\![N]\!]_{\tilde{\mu}}/x\}. & \end{array}$

Notice that we need $\eta\mu$-reduction to achieve this, and that $[\![(\lambda x.M)N]\!]_{\tilde{\mu}}$ runs past the term $[\![M\langle x := N\rangle]\!]_{\tilde{\mu}}$. □

Using this lemma, we can prove the following relation between the λ-calculus and $\overline{\lambda}\mu\tilde{\mu}$:

Theorem 2.4 *i) If $M \to_\beta N$, then $[\![M]\!]_{\tilde{\mu}} \to [\![N]\!]_{\tilde{\mu}}$.*
ii) If $M \to_{\mathrm{v}} N$, then $[\![M]\!]_{\tilde{\mu}} \to_{\mathrm{v}} [\![N]\!]_{\tilde{\mu}}$.

Proof: Both parts follow by induction on the definition of $\to_\beta$, using Lemma 2.3. For part *(ii)*, we also need to check that $[\![(\lambda x.M)N]\!]_{\tilde{\mu}} \to_{\mathrm{v}} [\![M]\!]_{\tilde{\mu}}\{\, [\![N]\!]_{\tilde{\mu}}/x\}$ only if N is a value. Well, then either $N \equiv x$ or $N \equiv \lambda y.N'$, and for both cases $[\![N]\!]_{\tilde{\mu}}$ is a value. Then the $\tilde{\mu}$-reduction in the proof of Lemma 2.3 is permitted, making the reduction CBV. □

Example 2.5 In $\overline{\lambda}\mu\tilde{\mu}$ we express the interaction between a program (term) and its context via commands. Although there is no notion of application, $\overline{\lambda}\mu\tilde{\mu}$ sees $[\![MN_1\cdots N_n]\!]_{\tilde{\mu}}$ as running $[\![M]\!]_{\tilde{\mu}}$ in the context that offers the terms $[\![N_1]\!]_{\tilde{\mu}}, \ldots, [\![N_n]\!]_{\tilde{\mu}}$ in sequence. To understand this, first notice that

$$\begin{array}{lcll} [\![MN_1N_2]\!]_{\tilde{\mu}} & = & \mu\alpha.\langle\, [\![MN_1]\!]_{\tilde{\mu}} \mid [\![N_2]\!]_{\tilde{\mu}} \cdot \alpha\rangle & = \\ & & \mu\alpha.\langle \mu\beta.\langle\, [\![M]\!]_{\tilde{\mu}} \mid [\![N_1]\!]_{\tilde{\mu}} \cdot \beta\rangle \mid [\![N_2]\!]_{\tilde{\mu}} \cdot \alpha\rangle & \to_\mu \ \mu\alpha.\langle\, [\![M]\!]_{\tilde{\mu}} \mid [\![N_1]\!]_{\tilde{\mu}} \cdot [\![N_2]\!]_{\tilde{\mu}} \cdot \alpha\rangle \end{array}$$

so it is easy to verify that

$$[\![MN_1\cdots N_n]\!]_{\tilde{\mu}} \ \to^*_\mu \ \mu\alpha.\langle\, [\![M]\!]_{\tilde{\mu}} \mid [\![N_1]\!]_{\tilde{\mu}} \cdot \ldots \cdot [\![N_n]\!]_{\tilde{\mu}} \cdot \alpha\rangle$$

which puts into evidence that, for λ-terms, the only contexts that are needed are *stacks*. Notice that the context $[\![N_1]\!]_{\tilde{\mu}} \cdot (\, [\![N_1]\!]_{\tilde{\mu}} \cdot \ldots \cdot [\![N_n]\!]_{\tilde{\mu}} \cdot \alpha)$ represents the λ-context for M, so stands for $\mathrm{C}[[\,]N_1]$, where $\mathrm{C}[\,] = [\,]N_2\cdots N_n$.

Since we can add the rule

$$\langle \lambda x.v_1 \mid v_2 \cdot \gamma\rangle \ \to_\beta \ \langle v_1\{v_2/x\} \mid \gamma\rangle$$

it is fair to say that it is the presence of the construct $\tilde{\mu}x.c$ that makes $\overline{\lambda}\mu\tilde{\mu}$ suitable for representing Classical Logic (see also [30]).

The standard reference for intersection type assignment for the λ-calculus is [11], which presented what has become known as the BCD-system. This in itself is based on earlier notions of intersection type assignment (for an overview, see [2, 5]), that all add the intersection type constructor '$\cap$' next to the standard type constructor '$\to$'. The BCD system differs from others in that it treats these two type constructors the same, allowing, in particular, intersection to occur at the right of arrow types; this general treatment is not necessary within the context of the λ-calculus (see [2]), but for $\overline{\lambda}\mu\tilde{\mu}$, as we will see in Example 7.1, to type all normal forms it is natural to have an intersection type occur on the right-hand side of an arrow type. We will, in this section, not consider a relation on types that is contra-variant on the arrow, since we are not now interested in modelling extensionality.

One of the many notions of intersection type assignment, and the one that is at the basis of the notion of intersection-union type assignment for $\overline{\lambda}\mu\tilde{\mu}$ as defined in the next section, is Krivine's *Système Dω*, which constitutes a restricted version of the BCD-system, *i.e.* not closed under η-reduction. We will show that our notion is a conservative extension of *Système Dω* in

that we can translate λ-terms typeable in that system to $\overline{\lambda}\mu\tilde{\mu}$-terms whilst preserving types. We will, however, only achieve that after changing the system to $\mathcal{M}^{c}$, allowing all types in both left and right contexts, in Section 9; we cannot prove this for the original system defined $\mathcal{M}^{\cap\cup}$ in Section 4, as first presented in [18] (see Example 5.6), and neither can it be achieved for the system presented in Section 8, as first presented in [19].

We first define *Système Dω*; from hereon, we will write $\underline{n}$ for the set $\{1,\ldots,n\}$.

Definition 2.6 *i*) The set $\mathcal{T}$ of *intersection types*, ranged over by $A, B\ldots$, is defined through the grammar:

$$A,B \ ::= \ \varphi \mid \top \mid (A\to B) \mid (A\cap B)$$

ii) A *statement* is an expression of the form $M:A$, with $M\in\Lambda$ and $A\in\mathcal{T}$. M is the *subject* and A the *predicate* of $M:A$.

iii) A *type-environment* Γ is a partial mapping from term variables to intersection types, and we write $x{:}A\in\Gamma$ if $\Gamma(x)=A$. We will write $x\notin\Gamma$ if Γ is not defined on x, and $\Gamma\backslash x$ when we remove x from the domain of Γ.

In the notation of types, as above, right-most outer-most parentheses in arrow types will be omitted, and we assume $\cap$ to bind stronger than $\to$. We will consider a pre-order on types which takes into account the idem-potence, commutativity and associativity of the intersection type constructor, and defines $\top$ to be the maximal element.

Definition 2.7 The relation $\leq$ is defined as the least pre-order (*i.e.* reflexive and transitive relation) on $\mathcal{T}$ such that:

$$A\cap B\leq A \quad A\cap B\leq B \quad A\leq\top \quad C\leq A\wedge C\leq B \ \Rightarrow \ C\leq A\cap B$$

and the relation $\sim$ is defined by:

$$A\leq B\leq A \ \Rightarrow \ A\sim B \qquad A\sim C\wedge B\sim D \ \Rightarrow \ A\to B\sim C\to D$$

$\mathcal{T}$ can be considered modulo $\sim$; then $\leq$ becomes a partial order. It is easy to show that both $(A\cap B)\cap C\sim B\cap(A\cap C)$ and $A\cap B\sim B\cap A$, so the type constructor $\cap$ is associative and commutative, and we will write $\cap_{\underline{n}}A_i$ for $A_1\cap\cdots\cap A_n$, and consider $\top$ to be the empty intersection: $\top=\cap_{\underline{0}}A_i$. Moreover, we will assume, unless stated explicitly otherwise, that in $\cap_{\underline{n}}A_i$ each A_i is not an intersection type.

The inspiration for the rules that define how to assign intersection types come directly from the way the logical *and* ($\wedge$) is treated in a natural deduction system.

$$(\wedge I):\ \frac{\Gamma\vdash A \quad \Gamma\vdash B}{\Gamma\vdash A\wedge B} \qquad (\wedge E):\ \frac{\Gamma\vdash A\wedge B}{\Gamma\vdash A}$$

As type assignment rules, these become:

$$(\cap I):\ \frac{\Gamma\vdash_\lambda M:A \quad \Gamma\vdash_\lambda M:B}{\Gamma\vdash_\lambda M:A\cap B} \qquad (\cap E):\ \frac{\Gamma\vdash_\lambda M:A\cap B}{\Gamma\vdash_\lambda M:A}$$

(generalised to dealing with an arbitrary number of types below). Notice that the Curry-Howard relation between typeable terms and proofs is lost: the introduction of the intersection is not represented by syntax, and neither is its elimination. In fact, seen as al, rule, rule $(\cap I)$ states that $A\wedge B$ can only be proven if A and B are proven using two proofs with the **exact same** structure; since it does not represent a proof construction step, we call it *non-logical*.

Definition 2.8 *Intersection type assignment* and *derivations* are defined by the following natural deduction system.

$$(Ax):\ \frac{}{\Gamma,x{:}A\vdash x:A}\qquad (\to I):\ \frac{\Gamma,x{:}A\vdash M:B}{\Gamma\vdash \lambda x.M:A\to B}\qquad (\to E):\ \frac{\Gamma\vdash M:A\to B\quad \Gamma\vdash N:A}{\Gamma\vdash MN:B}$$

$$(\cap I):\ \frac{\Gamma\vdash M:A_i\quad (\forall i\in\underline{n})}{\Gamma\vdash M:\cap_{\underline{n}}A_i}\,(n\geq 0,n\neq 1)\qquad (\cap E):\ \frac{\Gamma\vdash M:\cap_{\underline{n}}A_i}{\Gamma\vdash M:A_j}\,(\forall j\in\underline{n},n\geq 2)$$

We will write $\Gamma\vdash_{\mathcal{D}} M:A$ for statements that are derived using these rules.

Notice that $\Gamma\vdash_{\mathcal{D}} M:\top$ for all Γ, M by rule $(\cap I)$.

3 Some initial observations

As mentioned above, our aim is to come to a notion of type assignment that is a natural extension of Curien and Herbelin's system but closed under conversion, *i.e.* closed both under redex contraction and redex expansion. Since we can map the λ-calculus into $\overline{\lambda}\mu\tilde{\mu}$, we also want this encoding to preserve the assignable intersection types. The evident approach for this is to add intersection and union types together with the appropriate rules. Before we look at the various systems that have appeared in the past, we will investigate the minimal requirements a system should satisfy by looking at completeness. We choose to be not too formal in this section, but focus on intuition; formal definitions will follow when we prove our results.

It is well known that it is possible to show that intersection type assignment for the λ-calculus is closed under $=_\beta$. We will not show the formal proof of this result here (it's out of scope and published in various papers), but rather give an informal reasoning, highlighting the precise role of the type constructor '$\cap$' and the type constant $\top$. First we look at reduction.

Example 3.1 Suppose first that $\Gamma\vdash_{\mathcal{D}}(\lambda x.M)N:A$ is derived by $(\to E)$, so there exists B such that $\Gamma\vdash_{\mathcal{D}}\lambda x.M:B\to A$ and $\Gamma\vdash_{\mathcal{D}} N:B$. If $(\to I)$ is the last step performed for the first result, also $\Gamma,x{:}B\vdash_{\mathcal{D}} M:A$ and $\Gamma\vdash_{\mathcal{D}} N:B$. Then a derivation for $\Gamma\vdash_{\mathcal{D}} M\{N/x\}:A$ can be obtained by replacing in the derivation for $\Gamma,x{:}B\vdash_{\mathcal{D}} M:A$, all occurrences of the sub-derivation $\Gamma,x{:}B\vdash_{\mathcal{D}} x:B$ by the derivation for $\Gamma\vdash_{\mathcal{D}} N:B$.

In fact, this reasoning is applicable to many notions of type assignment, and does not depend at all on the presence of either $\cap$ or $\top$.

The second problem to solve in a proof for closure under β-equality is that of β-expansion:

Example 3.2 In order to show 'if $\Gamma\vdash_{\mathcal{D}} M\{N/x\}:A$, then $\Gamma\vdash_{\mathcal{D}}(\lambda x.M)N:A$', we consider two cases:

- Assume that the term-variable x occurs in M and so the term N is a sub-term of $M\{N/x\}$; then N is typed in the derivation for $\mathcal{D}::\Gamma\vdash_{\mathcal{D}} M\{N/x\}:A$, probably with several different types $B_1,\ldots,B_n$.

$$\frac{\boxed{}}{\Gamma\vdash N:B_i\quad(\forall i\in\underline{n})}\qquad\boxed{\phantom{\Gamma\vdash M\{N/x\}:A}}$$
$$\Gamma\vdash M\{N/x\}:A$$

  A derivation for $\Gamma,x{:}\cap_{\underline{n}}B_i\vdash_{\mathcal{D}} M:A$ can be obtained by replacing, in $\mathcal{D}$, all sub-derivations

for $\Gamma \vdash_D N : B_j$ by the derivation for

$$\dfrac{\dfrac{}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : \cap_{\underline{n}} B_i}\,(Ax)}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : B_j}\,(\cap E)$$

Then, using $(\to I)$, we can derive $\Gamma \vdash_D \lambda x.M : (\cap_{\underline{n}} B_i) \to A$, and, using $(\cap I)$ on the collection of removed sub-derivations, derive $\Gamma \vdash_D N : \cap_{\underline{n}} B_i$. Then, using $(\to E)$, the redex can be typed; in short:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : \cap_{\underline{n}} B_i}\,(Ax)}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : B_j}\,(\cap E) \quad (\forall j \in \underline{n})}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash M : A}}{\Gamma \vdash \lambda x.M : (\cap_{\underline{n}} B_i) \to A}\,(\to I) \qquad \dfrac{\Gamma \vdash N : B_i \quad (\forall i \in \underline{n})}{\Gamma \vdash N : \cap_{\underline{n}} B_i}\,(\cap I)$$
$$\Gamma \vdash (\lambda x.M)N : A \qquad (\to E)$$

- When the term-variable x does not occur in M, the term N is a not a subterm of $M\{N/x\}$ and $\Gamma \vdash_D M\{N/x\} : A$ stands for $\Gamma \vdash_D M : A$. In this case, the type $\top$ is used: since x does not occur in M, by weakening $x{:}\top$ can be assumed to appear in Γ, and applying rule $(\to I)$ gives $\Gamma \vdash_D \lambda x.M : \top \to A$. By $(\cap I)$, $\Gamma \vdash_D N : \top$, so, using $(\to E)$, the redex can be typed.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash M : A}}{\Gamma, x{:}\top \vdash M : A}\,(Wk)}{\Gamma \vdash \lambda x.M : \top \to A}\,(\to I) \qquad \dfrac{}{\Gamma \vdash N : \top}\,(\cap I)}{\Gamma \vdash (\lambda x.M)N : A}\,(\to E)$$

So it is fair to say that intersection is mainly added for reasons of expansion.

We would like to come to the definition of a notion of type assignment for $\overline{\lambda}\mu\tilde{\mu}$ that is both a conservative extension of Herbelin and Curien's system $\vdash_{\overline{\lambda}}$, and (using the embedding $[\![\cdot]\!]^{\mathrm{T}}_{\tilde{\mu}}$) of $\vdash_D$, and could serve as a basis for semantics using types. A minimal requirement then is that, as above for $\vdash_D$, the system should be both sound and complete.

When considering reduction in $\overline{\lambda}\mu\tilde{\mu}$, we need to deal with the three reduction rules:

$$\begin{array}{rrcl} (\to): & \langle \lambda x.v_1 \,|\, v_2 \cdot e \rangle & \to & \langle v_2 \,|\, \tilde{\mu}x.\langle v_1 \,|\, e \rangle \rangle \\ (\mu): & \langle \mu\beta.c \,|\, e \rangle & \to & c\{e/\beta\} \\ (\tilde{\mu}): & \langle v \,|\, \tilde{\mu}x.c \rangle & \to & c\{v/x\} \end{array}$$

Let us look at the last two (substitution) rules; following the above line of thought, we can say:

Example 3.3 Let $\langle \mu\beta.c \,|\, e \rangle \to c\{e/\beta\}$, and assume the latter is typeable, so we have a derivation $\mathcal{D} :: c\{e/\beta\} : \Gamma \vdash \Delta$. Now, if e occurs more than once in $c\{e/\beta\}$, it might be typed with different types $B_1, \ldots, B_n$.

$$\dfrac{\dfrac{\boxed{\mathcal{D}_i}}{\Gamma \mid e : B_i \vdash \Delta} \quad (\forall i \in \underline{n})}{\boxed{\mathcal{D}}}$$
$$c\{e/\beta\} : \Gamma \vdash \Delta$$

Collecting the sub-derivations $\mathcal{D}_i :: \Gamma \mid e : B_i \vdash \Delta$, as above, we need to collect these types using union, say, into the type $B_1 \cup \cdots \cup B_n$, with which we can construct

$$\dfrac{\dfrac{\dfrac{}{\Gamma \mid \beta : \cup_{\underline{n}} B_i \vdash \beta{:}\cup_{\underline{n}} B_i, \Delta}}{\Gamma \mid \beta : B_i \vdash \beta{:}\cup_{\underline{n}} B_i, \Delta}(\cup v) \quad (\forall i \in \underline{n}) \quad \boxed{\mathcal{D}}}{\dfrac{c : \Gamma \vdash \beta{:}\cup_{\underline{n}} B_i, \Delta}{\Gamma \vdash \mu\beta.c : \cup_{\underline{n}} B_i \mid \Delta}(\mu) \qquad \dfrac{\boxed{\mathcal{D}_i} \quad \Gamma \mid e : B_i \vdash \Delta \quad (\forall i \in \underline{n})}{\Gamma \mid e : \cup_{\underline{n}} B_i \vdash \Delta}(\cup L)}(cut)}{\langle \mu\beta.c \mid e \rangle : \Gamma \vdash \Delta}$$

which implies that, at least, the right-hand environment contains union types and that we need the type assignment rules $(\cup v)$ for context variables and $(\cup L)$ for contexts; in case e does not occur in $c\{e/\beta\}$, we add the rule $(\bot)$.

$$\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}}}{c : \Gamma \vdash \Delta}(W)}{c : \Gamma \vdash \beta{:}\bot, \Delta}(\mu) \qquad \dfrac{}{\Gamma \mid e : \bot \vdash \Delta}(\bot)}{\langle \mu\beta.c \mid e \rangle : \Gamma \vdash \Delta}(cut)$$

Similarly, we can deduce:

Example 3.4 Let $\langle v \mid \tilde{\mu}x.c \rangle \to c\{v/x\}$, and assume the latter is typeable, then, reasoning as above, we need to collect these types using intersection into the type $B_1 \cap \cdots \cap B_n$, with which we can construct

$$\dfrac{\dfrac{\boxed{\mathcal{D}_i}}{\Gamma \vdash v : B_i \mid \Delta} \quad (\forall i \in \underline{n})}{\Gamma \vdash v : \cap_{\underline{n}} B_i \mid \Delta}(\cap R) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : \cap_{\underline{n}} B_i \mid \Delta}(\cap v)}{\Gamma, x{:}\cap_{\underline{n}} B_i \vdash x : B_i \mid \Delta} \quad (\forall i \in \underline{n}) \quad \boxed{\mathcal{D}}}{c : \Gamma, x{:}\cap_{\underline{n}} B_i \vdash \Delta}(\tilde{\mu})}{\langle v \mid \tilde{\mu}x.c \rangle : \Gamma \vdash \Delta}(cut)$$

so the left-hand environment should contain intersection types and we need the type assignment rules $(\cap v)$ for term variables and $(\cap R)$ for terms; in case v does not occur in $c\{v/x\}$, we add the rule $\top$.

$$\dfrac{\dfrac{}{\Gamma \vdash v : \top \mid \Delta}(\top) \qquad \dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}}}{c : \Gamma \vdash \Delta}(W)}{c : \Gamma, x{:}\top \vdash \Delta}(\tilde{\mu})}{\Gamma \vdash \tilde{\mu}x.c : \top \mid \Delta}}{\langle v \mid \tilde{\mu}x.c \rangle : \Gamma \vdash \Delta}(cut)$$

Combining these observations, we have some minimal requirements:

Example 3.5 (The minimal system) The set of types we consider for the intersection-union type assignment system for $\overline{\lambda}\mu\tilde{\mu}$ is:

$$A, B \ ::= \ \varphi \mid \top \mid \bot \mid (A \to B) \mid (A \cap B) \mid (A \cup B)$$

We allow intersection types in left-hand environments Γ, and union in right-hand environments Δ (see Definition 4.4), and add the derivation rules:

$$(cut): \frac{\Gamma \vdash v : A \mid \Delta \quad \Gamma \mid e : A \vdash \Delta}{\langle v|e\rangle : \Gamma \vdash \Delta}$$

$$(\cap v): \frac{}{\Gamma, x{:}\cap_{\underline{n}} A_i \vdash x : A_i \mid \Delta} \qquad (\cup v): \frac{}{\Gamma \mid \alpha : A_i \vdash \alpha{:}\cup_{\underline{n}} A_i, \Delta}$$

$$(\to R): \frac{\Gamma, x{:}A \vdash v : B \mid \Delta}{\Gamma \vdash \lambda x.v : A \to B \mid \Delta} \qquad (\to L): \frac{\Gamma \vdash v : A \mid \Delta \quad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid v \cdot e : A \to B \vdash \Delta}$$

$$(\mu): \frac{c : \Gamma \vdash \alpha{:}A, \Delta}{\Gamma \vdash \mu\alpha.c : A \mid \Delta} \qquad (\tilde{\mu}): \frac{c : \Gamma, x{:}A \vdash \Delta}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

$$(\cap R): \frac{\Gamma \vdash v : A_i \mid \Delta \quad (\forall i \in \underline{n})}{\Gamma \vdash v : \cap_{\underline{n}} A_i \mid \Delta} \qquad (\cup L): \frac{\Gamma \mid e : A_i \vdash \Delta \quad (\forall i \in \underline{n})}{\Gamma \mid e : \cup_{\underline{n}} A_i \vdash \Delta}$$

$$(\top): \frac{}{\Gamma \vdash v : \top \mid \Delta} \qquad (\bot): \frac{}{\Gamma \mid e : \bot \vdash \Delta}$$

Notice that the rules $(Ax\text{-}R)$ and $(Ax\text{-}L)$ of Definition 1.3 are special cases of the rules $(\cap v)$ and $(\cup v)$ we constructed above. Moreover, we can derive

$$\frac{\dfrac{}{\Gamma, x{:}\cap_{\underline{n}} A_i \vdash x : A_1 \mid \Delta}(\cap v) \quad \cdots \quad \dfrac{}{\Gamma, x{:}\cap_{\underline{n}} A_i \vdash x : A_n \mid \Delta}(\cap v)}{\Gamma, x{:}\cap_{\underline{n}} A_i \vdash x : \cap_{\underline{n}} A_i \mid \Delta}(\cap R)$$

and

$$\frac{\dfrac{}{\Gamma \mid \alpha : A_1 \vdash \alpha{:}\cup_{\underline{n}} A_i, \Delta}(\cup v) \quad \cdots \quad \dfrac{}{\Gamma \mid \alpha : A_n \vdash \alpha{:}\cup_{\underline{n}} A_i, \Delta}(\cup v)}{\Gamma \mid \alpha : \cup_{\underline{n}} A_i \vdash \alpha{:}\cup_{\underline{n}} A_i, \Delta}(\cup L)$$

so can model the rules $(Ax\text{-}R)$ and $(Ax\text{-}L)$ for intersection and union types.

We will, over the next sections, see a number of approaches towards defining a sound and complete notion of intersection-union type assignment for $\overline{\lambda}\mu\tilde{\mu}$ that have at least the rules of this minimal system. Unfortunately, as we will see in Section 7, soundness already fails for this system; since we reasoned above that this system is the minimal one to satisfy completeness, this puts into evidence that a sound and complete system for $\overline{\lambda}\mu\tilde{\mu}$ cannot be defined.

Notice that we have not considered rule $(\to)$ in our analysis above, but just looked at the substitution rules (μ) and $(\tilde{\mu})$; in fact, rule $(\to)$ in the context of union types forces some additional measures, as can be seen in Example 6.1. In Section 9 we will look at how to extend the above system and fix this flaw; see the proof of Theorem 9.9.

4 The system $\mathcal{M}$ of intersection and union typing for $\overline{\lambda}\mu\tilde{\mu}$

The first notion of intersection and union typing for $\overline{\lambda}\mu\tilde{\mu}$, System $\mathcal{M}^{\cap\cup}$, was presented in [18]. System $\mathcal{M}^{\cap\cup}$ is set up using the minimal rules we established above in Example 3.5; since the aim of [18] is to characterise strong normalisation, it does not consider $\top$ or $\bot$. This was followed by [19], which presented improvements/variants of the original system. In this paper, we will revisit these systems, in order, albeit in a presentation that is adapted to our approach, and discuss their results and shortcomings.

We start with the definition of intersection/union types.

Definition 4.1 (Intersection and Union Types) *i*) Let φ be a type variable, as before. The set $\mathcal{T}_{\cap\cup}$ of *intersection-union types* (we will write $\mathcal{T}$ for short), ranged over by $A, B, \ldots$ is inductively defined by:

$$A, B \ ::= \ \varphi \mid \top \mid \bot \mid (A \to B) \mid (A \cap B) \mid (A \cup B)$$

ii) $\mathcal{T}_p$ is the set of *proper types*, defined by: $\mathcal{T}_p ::= \varphi \mid (\mathcal{T} \to \mathcal{T})$.

iii) $\mathcal{T}_\cap$ is the set of *intersection types*, defined by: $\mathcal{T}_\cap ::= \mathcal{T}_p \mid \top \mid (\mathcal{T}_\cap \cap \mathcal{T}_\cap)$.

iv) $\mathcal{T}_\cup$ is the set of *union types*, defined by: $\mathcal{T}_\cup ::= \mathcal{T}_p \mid \bot \mid (\mathcal{T}_\cup \cup \mathcal{T}_\cup)$.

As above, we will omit unnecessary parentheses in types: the type constructors '$\cap$' and '$\cup$' will bind more strongly than '$\to$', so $A\cap B\to C\cap D$ is used for $((A\cap B)\to(C\cap D))$, $A\to B\cap C\to D$ for $(A\to((B\cap C)\to D))$, and $(A\to B)\cap C\to D$ for $(((A\to B)\cap C)\to D)$. We will sometimes write the omissible parentheses to enhance readability. We will consider pre-orders on types which take into account the idem-potence, commutativity and associativity of the intersection and union type constructors.

Definition 4.2 (Relations on types) *i*) The relation $\leq_\cap$ is defined as the least pre-order on $\mathcal{T}_\cap$ such that:

$$A \leq_\cap \top \quad A\cap B \leq_\cap A \quad A\cap B \leq_\cap B \quad A \leq_\cap C \wedge A \leq_\cap B \;\Rightarrow\; A \leq_\cap B\cap C$$

ii) The relation $\leq_\cup$ is defined as the least pre-order on $\mathcal{T}_\cup$ such that:

$$\bot \leq_\cup A \quad A \leq_\cup A\cup B \quad B \leq_\cup A\cup B \quad A \leq_\cup C \wedge B \leq_\cup C \;\Rightarrow\; A\cup B \leq_\cup C$$

iii) We define $A \leq B$ by: there exists a $C\in\mathcal{T}_p$ such that $A \leq_\cap C \leq_\cup B$.

iv) The equivalence relation $\sim_\cap$ on $\mathcal{T}_\cap$ is defined by:

$$A \leq_\cap B \leq_\cap A \;\Rightarrow\; A \sim_\cap B \quad A \sim_\cap C \;\wedge\; B \sim_\cap D \;\Rightarrow\; A\to B \sim_\cap C\to D$$

The relation $\sim_\cup$ on $\mathcal{T}_\cup$ is defined similarly.

The set $\mathcal{T}_\cap$ and $\mathcal{T}_\cup$ will be considered modulo their respective equivalence relations generated by the pre-orders. As before, intersection and union are associative and commutative, and we will write $\cap_{\underline{n}} A_i$ for the type $A_1\cap\cdots\cap A_n$, as well as $\cup_{\underline{n}} A_i$ for $A_1\cup\cdots\cup A_n$, and will only allow permutations of types within an intersection or within a union.

Remark 4.3 As was the case for intersection types for the Lambda Calculus, the inspiration for the rules that define below how to assign intersection and union come directly from the way *and* ($\wedge$) and *or* ($\vee$) are treated in LK.

$$(\wedge R):\ \frac{\Gamma\vdash A,\Delta \quad \Gamma\vdash B,\Delta}{\Gamma\vdash A\wedge B,\Delta} \qquad (\wedge L):\ \frac{\Gamma, A\vdash\Delta}{\Gamma, A\wedge B\vdash\Delta}$$

$$(\vee R):\ \frac{\Gamma\vdash A,\Delta}{\Gamma\vdash A\vee B,\Delta} \qquad (\vee L):\ \frac{\Gamma, A\vdash\Delta \quad \Gamma, B\vdash\Delta}{\Gamma, A\vee B\vdash\Delta}$$

If we view the explicitly mentioned formulae as under focus, this becomes:

$$(\wedge R):\ \frac{\Gamma\vdash A\mid\Delta \quad \Gamma\vdash B\mid\Delta}{\Gamma\vdash A\wedge B\mid\Delta} \qquad (\wedge L):\ \frac{\Gamma\mid A\vdash\Delta}{\Gamma\mid A\wedge B\vdash\Delta}$$

$$(\vee R):\ \frac{\Gamma\vdash A\mid\Delta}{\Gamma\vdash A\vee B\mid\Delta} \qquad (\vee L):\ \frac{\Gamma\mid A\vdash\Delta \quad \Gamma\mid B\vdash\Delta}{\Gamma\mid A\vee B\vdash\Delta}$$

This naturally leads to the definition of the following rules for intersection and union:

$$(\cap R):\ \frac{\Gamma\vdash v:A\mid\Delta \quad \Gamma\vdash v:B\mid\Delta}{\Gamma\vdash v:A\cap B\mid\Delta} \qquad (\cap L):\ \frac{\Gamma\mid e:A\vdash\Delta}{\Gamma\mid e:A\cap B\vdash\Delta}$$

$$(\cup R):\ \frac{\Gamma\vdash v:A\mid\Delta}{\Gamma\vdash v:A\cup B\mid\Delta} \qquad (\cup L):\ \frac{\Gamma\mid e:A\vdash\Delta \quad \Gamma\mid e:B\vdash\Delta}{\Gamma\mid e:A\cup B\vdash\Delta}$$

This is, essentially, the approach of [18, 29]; unfortunately, as we will show below, this approach does not lead to the correct system, in the sense that not all desired properties can be shown to hold. In fact, as we will argue below (Theorem 9.9), there is no reason to treat the formulae exclusively as activated; also the variants that view the explicitly mentioned formulae as just appearing in the environment, *i.e.* as when the sequent is witnessed by a command

$$(): \frac{c:\Gamma\vdash\alpha{:}A,\Delta \quad c:\Gamma\vdash\alpha{:}B,\Delta}{c:\Gamma\vdash\alpha{:}A\cap B,\Delta} \qquad (\cap L\text{-v}): \frac{c:\Gamma,x{:}A\vdash\Delta}{c:\Gamma,x{:}A\cap B\vdash\Delta}$$

$$(): \frac{c:\Gamma\vdash\alpha{:}A,\Delta}{c:\Gamma\vdash\alpha{:}A\cup B,\Delta} \qquad (): \frac{c:\Gamma,x{:}A\vdash\Delta \quad c:\Gamma,x{:}B\vdash\Delta}{c:\Gamma,x{:}A\cup B\vdash\Delta}$$

should be added; notice that now a left-hand environment can contain union types and a right-hand environment intersection types.

Definition 4.4 (Environments) *i*) A *left-hand environment* Γ is a partial mapping from term variables to intersection types, represented as a set of statements with only distinct variables as subjects.

ii) We write $\Gamma\cap x{:}A$ for the left-hand environment $\Gamma\cup\{x{:}A\}$ if x does not occur in Γ, and for $\Gamma\backslash x\cup\{x{:}A\cap B\}$ if $x{:}B\in\Gamma$, and will write $\Gamma,x{:}A$ for $\Gamma\cap x{:}A$ when $x\notin\Gamma$.

iii) We write $\Gamma_1\cap\Gamma_2$ for the left-hand environment Γ defined by: if $x{:}A\in\Gamma$ then either:

a) $x{:}A\in\Gamma_1$ and x does not occur in Γ_2, or

b) $x{:}A\in\Gamma_2$ and x does not occur in Γ_1, or

c) $x{:}B\in\Gamma_1$ and $x{:}C\in\Gamma_2$, and $A=B\cap C$,

and write $\cap_{\underline{n}}\Gamma_i$ for $\Gamma_1\cap\cdots\cap\Gamma_n$.

iv) A *right-hand environment* Δ is a partial mapping from environment variables to union types, represented as a set of statements with only distinct variables as subjects; the notions $\alpha{:}A,\Delta$, as well as $\Delta_1\cup\Delta_2$ and $\cup_{\underline{n}}\Delta_i$ are defined as above.

Notice that the restriction to intersection types for left-hand environments and union types for right-hand environments clearly is inspired by the logical reading of $A_1,\ldots,A_n\vdash B_1,\ldots,B_m$, where the left-hand formulae are joined by the logical *and*, and the right-hand formulae are joined by *or*. As was remarked in [18], left-hand environments are limited to intersection types (and right-hand environment to union types) for reasons of soundness, since otherwise the substitution lemmae (see Lemma 5.8 and 5.9) would no longer hold; we will see that this restriction is too strong in that it blocks completeness, and too weak in that it does not solve the soundness problem.

We will now present the notion of intersection-union typing (or better, environment assignment) for $\overline{\lambda}\mu\tilde{\mu}$, System $\mathcal{M}$, a slight variant of System $\mathcal{M}^{\cap\cup}$ as defined in [18] in a notation adapted to our purposes.

Definition 4.5 (System $\mathcal{M}$) System $\mathcal{M}$ is defined using the following sequent system:

$$(cut): \frac{\Gamma_1 \vdash v : A \mid \Delta_1 \quad \Gamma_2 \mid e : A \vdash \Delta_2}{\langle v|e\rangle : \Gamma_1 \cap \Gamma_2 \vdash \Delta_1 \cup \Delta_2}$$

$$(Ax\text{-}R): \frac{}{\Gamma, x{:}A \vdash x : B \mid \Delta}\,(A \leq_\cap B, B \in \mathcal{T}_p) \qquad (Ax\text{-}L): \frac{}{\Gamma \mid \alpha : A \vdash \alpha{:}B, \Delta}\,(A \leq_\cup B, A \in \mathcal{T}_p)$$

$$(\to R): \frac{\Gamma, x{:}A \vdash v : B \mid \Delta}{\Gamma \vdash \lambda x.v : A \to B \mid \Delta} \qquad (\to L): \frac{\Gamma_1 \vdash v : A \mid \Delta_1 \quad \Gamma_2 \mid e : B \vdash \Delta_2}{\Gamma_1 \cap \Gamma_2 \mid v \cdot e : A \to B \vdash \Delta_1 \cup \Delta_2}$$

$$(\mu): \frac{c : \Gamma \vdash \alpha{:}A, \Delta}{\Gamma \vdash \mu\alpha.c : A \mid \Delta} \qquad (\tilde{\mu}): \frac{c : \Gamma, x{:}A \vdash \Delta}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

$$(\cap R): \frac{\Gamma_i \vdash v : A_i \mid \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \vdash v : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}}\Delta_i}\,(n \geq 2) \qquad (\cap L): \frac{\Gamma \mid e : A_i \vdash \Delta \quad (\forall i \in \underline{n})}{\Gamma \mid e : \cap_{\underline{n}} A_i \vdash \Delta}\,(n \geq 2)$$

$$(\cup R): \frac{\Gamma \vdash v : A_i \mid \Delta \quad (\forall i \in \underline{n})}{\Gamma \vdash v : \cup_{\underline{n}} A_i \mid \Delta}\,(n \geq 2) \qquad (\cup L): \frac{\Gamma_i \mid e : A_i \vdash \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \mid e : \cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i}\,(n \geq 2)$$

We call a derivation constructed via these rules *proper* if it does not end with one of the bottom four rules.

Notice that this system contains the minimal one of Example 3.5, but for the absence of rules $(\top)$ and $(\bot)$; it adds the rules $(\cup R)$ and $(\cup L)$.

This system is a variant of system $\mathcal{M}^{\cap\cup}$ of [18] in that we use a multiplicative style (combine environments in rules via $\cap$ and $\cup$) rather than assuming the same environment is used in the sub-derivations; this is done mainly for convenience when drawing derivations. Notice that, since we allow any Γ and Δ in our rules, the normal style of rules, as in

$$(\to L): \frac{\Gamma \vdash v : A \mid \Delta \quad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid v \cdot e : A \to B \vdash \Delta}$$

now is a special case; this also follows from Lemma 5.1. Notice also that, in rule $(Ax\text{-}R)$ (and $(Ax\text{-}L)$), a type is selected from within an intersection (a union) for a term (context) variable; since the extracted types are in $\mathcal{T}_p$, these rules actually are $(\cap v)$ and $(\cup v)$. In aim to obtain some of the same functionality for the *implicit* variables, the system adds rules $(\cup R)$ and $(\cap L)$.

Other than for being inspired by the logical right-rule for '*or*' and the logical left-rule for '*and*', there seems to be no reason why the rules $(\cup R)$ and $(\cap L)$ are added; since intersection types are limited to left-hand environments, and union types to right-hand environments, these rules add no operational power, as the behaviour they model is already sufficiently expressed via rules $(Ax\text{-}R)$ and $(Ax\text{-}L)$. In fact, we only see a use for these rules in a system that would allow all types in both left and right-hand environments, as illustrated in Example 9.10.

In Section 10 we will consider two restrictions of this system, defined as follows:

Definition 4.6 (CBV and CBN intersection and union typing for $\overline{\lambda}\mu\tilde{\mu}$) *i*) The notion of typing $\vdash_{\mathrm{v}}$ is defined as $\vdash_{\mathcal{M}}$, by changing rule $(\cap R)$:

$$(\cap R_{\mathrm{v}}): \frac{\Gamma_i \vdash V : A_i \mid \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \vdash V : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}}\Delta_i}\,(n \geq 2)$$

ii) The notion of typing $\vdash_{\mathrm{N}}$ is defined as $\vdash_{\mathcal{M}}$, by changing rule $(\cup L)$:

$$(\cup L_{\mathrm{N}}): \frac{\Gamma_i \mid E : A_i \vdash \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \mid E : \cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i}\,(n \geq 2)$$

For these system we will be able to show a soundness result; we cannot achieve that for any of the other systems we study here. Since $\top$ and $\bot$ only play a role for completeness, we need not consider those types for these two systems.

System $\mathcal{M}$ is defined in [18] without any relation on types, it is just stated that types are considered to be defined modulo commutativity and associativity for $\cap$ and $\cup$; this corresponds to the type inclusion relations we use here. Notice we do not have a contra-variant type inclusion relation (which would state that $C \leq A \wedge B \leq D \Rightarrow A{\to}B \leq C{\to}D$); this combined with the full BCD-intersection types gives a notion of typing that is closest to Krivine's *Système Dω* (see Theorem 9.5).

This system does not have *choice*, *i.e.* we cannot show that, if $\Gamma \vdash_{\mathcal{M}} v : A \cup B \mid \Delta$, then either $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$ or $\Gamma \vdash_{\mathcal{M}} v : B \mid \Delta$, as would hold in an intuitionistic system.

Example 4.7 Take the term $\mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle$; we can type this term as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{x{:}A \vdash x : A \mid}\quad \overline{\mid \delta : A \vdash \beta{:}B, \delta{:}A}}{\langle x|\delta\rangle : x{:}A \vdash \beta{:}B, \delta{:}A}(cut)}{x{:}A \vdash \mu\beta.\langle x|\delta\rangle : B \mid \delta{:}A}(\mu)}{\vdash \lambda x\mu\beta.\langle x|\delta\rangle : A{\to}B \mid \delta{:}A}(\to R)\quad \overline{\mid \delta : A{\to}B \vdash \delta{:}A{\to}B}}{\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : \vdash \delta{:}A \cup (A{\to}B)}(cut)}{\vdash \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A \cup (A{\to}B) \mid}(\mu)$$

Notice that we need *both* types for δ to type the whole term $\mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle$. It is therefore impossible to derive either $\vdash_{\mathcal{M}} \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A \mid$ or $\vdash_{\mathcal{M}} \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A{\to}B \mid$.

A similar observation can be made with respect to the necessity of intersection in order to type $\tilde{\mu}y.\langle y|y{\cdot}\beta\rangle$,

$$\dfrac{\dfrac{\overline{y{:}C{\to}D \vdash y : C{\to}D \mid}\quad \dfrac{\overline{y{:}C \vdash y : C \mid}\quad \overline{\mid \beta : D \vdash \beta{:}D}}{y{:}C \mid y{\cdot}\beta : C{\to}D \vdash \beta{:}D}(\to L)}{\langle y|y{\cdot}\beta\rangle : y{:}(C{\to}D) \cap C \vdash \beta{:}D}(cut)}{\mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : (C{\to}D) \cap C \vdash \beta{:}D}(\tilde{\mu})$$

5 Basic properties of System $\mathcal{M}$

We start our investigation of System $\mathcal{M}$ by showing some basic properties. As is usual, we can restrict the environment to just those statements that are relevant, or generalise them by adding statements.

Lemma 5.1 (Thinning and weakening) *i) If* $c : \Gamma \vdash_{\mathcal{M}} \Delta$ *and* $\Gamma' = \{x{:}B \in \Gamma \mid x \in fv(c)\}$ *and* $\Delta' = \{\alpha{:}B \in \Gamma \mid \alpha \in fv(c)\}$, *then also* $c : \Gamma' \vdash_{\mathcal{M}} \Delta'$. *Similar for* $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$ *and* $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$.

ii) If $c : \Gamma \vdash_{\mathcal{M}} \Delta$, $\Gamma' \leq_{\cap} \Gamma$ *and* $\Delta \leq_{\cup} \Delta'$, *then* $c : \Gamma' \vdash_{\mathcal{M}} \Delta'$. *Similar for* $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$ *and* $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$.

Proof: Both parts are shown by simultaneous induction. □

We can show that we can type all terms in normal form.

Definition 5.2 Normal forms are defined by the grammar

$$\begin{array}{rcl} v_{nf} & ::= & x \mid \lambda x.v_{nf} \mid \mu\alpha.c_{nf} \\ e_{nf} & ::= & \alpha \mid v_{nf}{\cdot}e_{nf} \mid \tilde{\mu}x.c_{nf} \\ c_{nf} & ::= & \langle x|\alpha\rangle \mid \langle x \,|\, v_{nf}{\cdot}e_{nf}\rangle \mid \langle \lambda x.v_{nf} \,|\, \alpha\rangle \end{array}$$

Theorem 5.3 *i) If v is in normal form, then there exist Γ, A, and Δ such that $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$.*

ii) If e is in normal form, then there exist Γ, A, and Δ such that $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$.

iii) If c is in normal form, then there exist Γ and Δ such that $c : \Gamma \vdash_{\mathcal{M}} \Delta$.

Proof: By simultaneous induction; we show only some of the cases.

(x): Take $\Gamma = x{:}A$, and $\Delta = \emptyset$.

$(\lambda x.v_{nf})$: By induction, there are Γ, B, and Δ such that $\Gamma \vdash_{\mathcal{M}} v_{nf} : B \mid \Delta$. If x occurs free in v_{nf}, it has a type C in Γ, and by applying rule $(\to R)$ we can derive $\Gamma\backslash x{:}C \vdash_{\mathcal{M}} \lambda x.v_{nf} : C \to B \mid \Delta$. If x is not free in v_{nf}, we can add $x{:}C$ by weakening, and conclude the same way.

$(v_{nf} \cdot e_{nf})$: By induction, there are Γ_1, B, Δ_1, and Γ_2, C, and Δ_2 such that $\Gamma_1 \vdash_{\mathcal{M}} v_{nf} : B \mid \Delta_1$ and $\Gamma_2 \mid e_{nf} : C \vdash_{\mathcal{M}} \Delta_2$. By applying rule $(\to L)$ we derive $\Gamma_1 \cap \Gamma_2 \mid v_{nf} \cdot e_{nf} : B \to C \vdash_{\mathcal{M}} \Delta_1 \cup \Delta_2$.

$(\langle x \mid \alpha \rangle)$: Take $\Gamma = x{:}A$, and $\Delta = \alpha{:}A$; the result follows by rules $(Ax\text{-}R)$, $(Ax\text{-}L)$, and (cut).

$(\langle x \mid v_{nf} \cdot e_{nf} \rangle)$: By induction, there are Γ_1, B, Δ_1, and Γ_2, C, and Δ_2 such that $\Gamma_1 \vdash_{\mathcal{M}} v_{nf} : B \mid \Delta_1$ and $\Gamma_2 \mid e_{nf} : C \vdash_{\mathcal{M}} \Delta_2$. By applying rule $(\to L)$ we derive $\Gamma_1 \cap \Gamma_2 \mid v_{nf} \cdot e_{nf} : B \to C \vdash_{\mathcal{M}} \Delta_1 \cup \Delta_2$. We have $x{:}B \to C \vdash_{\mathcal{M}} x : B \to C \mid$ by rule $(Ax\text{-}R)$ and the result follows by rule (cut). □

Notice the role of intersection and union in the third and fifth cases.

The relation between syntax and derivable judgements is formulated through:

Lemma 5.4 (Generation Lemma) *i) If $\langle v \mid e \rangle : \Gamma \vdash_{\mathcal{M}} \Delta$, then $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$ and $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$ for some A.*

ii) If $\Gamma, x{:}A \vdash_{\mathcal{M}} x : B \mid \Delta$, then $A \leq B$.

iii) If $\Gamma \vdash_{\mathcal{M}} \lambda x.v : A \mid \Delta$, then there are $A_i\,(\forall i \in \underline{n})$ such that $\cap_{\underline{n}} A_i \leq A$, and for all $i \in \underline{n}$ there exist B_i, C_i such that $A_i = B_i \to C_i$, and $\Gamma, x{:}B_i \vdash_{\mathcal{M}} v : C_i \mid \Delta$.

iv) If $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \mid \Delta$, then there are $A_i\,(\forall i \in \underline{n})$ such that $\cap_{\underline{n}} A_i \leq A$, and, for all $i \in \underline{n}$, $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A_i, \Delta$.

v) If $\Gamma \mid \alpha : A \vdash_{\mathcal{M}} \alpha{:}B, \Delta$, then $A \leq B$.

vi) If $\Gamma \mid v \cdot e : A \vdash_{\mathcal{M}} \Delta$, then there are $A_i\,(\forall i \in \underline{n})$ such that $A \leq \cup_{\underline{n}} A_i$, and for all $i \in \underline{n}$ there are B_i, C_i such that $A_i = B_i \to C_i$, and $\Gamma \vdash_{\mathcal{M}} v : B_i \mid \Delta$ and $\Gamma \mid e : C_i \vdash_{\mathcal{M}} \Delta$.

vii) If $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathcal{M}} \Delta$, then there are $A_i\,(\forall i \in \underline{n})$ such that $A \leq \cup_{\underline{n}} A_i$, and, for all $i \in \underline{n}$, $c : \Gamma, x{:}A_i \vdash_{\mathcal{M}} \Delta$.

Proof: Straightforward. □

In particular, the Generation Lemma does *not* state that "*if* $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \mid \Delta$, *then* $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A, \Delta$"; in fact, we could have used $(\cap R)$ when deriving the first statement:

$$\dfrac{\dfrac{\boxed{\phantom{c : \Gamma \vdash \alpha{:}A_i, \Delta}}}{c : \Gamma \vdash \alpha{:}A_i, \Delta}\,(\mu)}{\Gamma \vdash \mu\alpha.c : A_i \mid \Delta}\;(\forall i \in \underline{n})}{\Gamma \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \Delta}\,(\cap R)$$

We can perhaps derive $c : \Gamma \vdash_{\overline{\lambda}} \alpha{:}A_i, \Delta$, for all $i \in \underline{n}$, but cannot derive $c : \Gamma \vdash_{\overline{\lambda}} \alpha{:}\cap_{\underline{n}} A_i, \Delta$, since we do not allow intersection types in right-hand type environments. Similarly, we can derive

$$\dfrac{\dfrac{\boxed{\phantom{c : \Gamma, x{:}A_i \vdash \Delta}}}{c : \Gamma, x{:}A_i \vdash \Delta}\,(\tilde{\mu})}{\Gamma \mid \tilde{\mu}x.c : A_i \vdash \Delta}\;(\forall i \in \underline{n})}{\Gamma \mid \tilde{\mu}x.c : \cup_{\underline{n}} A_i \vdash \Delta}\,(\cup L)$$

but not $c : \Gamma, x{:}\cup_{\underline{n}} A_i \vdash_{\mathcal{M}} \Delta$. This creates a problem for soundness, as we will see below.

Notice that the Generation Lemma is formulated in terms of the type inclusion relation $\leq$. In fact, we can show the following:

Lemma 5.5 The rules

$$(\leq R):\ \frac{\Gamma \vdash v : A \mid \Delta}{\Gamma \vdash v : B \mid \Delta}(A \leq B) \qquad (\leq L):\ \frac{\Gamma \mid e : B \vdash \Delta}{\Gamma \mid e : A \vdash \Delta}(A \leq B)$$

are admissible in $\vdash_{\mathcal{M}}$.

Proof: We show first that the rules

$$(\leq_\cup):\ \frac{\Gamma \vdash v : A \mid \Delta}{\Gamma \vdash v : B \mid \Delta}(A \leq_\cup B) \qquad (\leq_\cap):\ \frac{\Gamma \mid e : B \vdash \Delta}{\Gamma \mid e : A \vdash \Delta}(A \leq_\cap B)$$

are admissible. For $(\leq_\cup)$: if $A \leq_\cup B$, then $B = \cup_{\underline{n}} B_i$, $A = \cup_{\underline{m}} A_j$, and, for every $j \in \underline{m}$ there is an $i \in \underline{n}$ such that $A_j = B_i$, so $B = A \cup C$, for some C, and the result follows from rule $(\cup R)$. For $(\leq_\cap)$, the reasoning is similar.

For the first case, $(\leq R)$, assume $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$; if $A \leq B$, then there exists $C \in \mathcal{T}_p$ such that $A \leq_\cap C$ and $C \leq_\cup B$, or $A \leq_\cup B$. This derivation ends with either:

$(Ax\text{-}R)$: Then $v = x$, and $x{:}D \in \Gamma$ with $D \leq_\cap A$; then also $D \leq_\cap C$, so also $\Gamma \vdash_{\mathcal{M}} x : C \mid \Delta$.

$(\to R)$: Then $v = \lambda x.v'$, and $A = D \to E$. Since $D \to E \leq_\cap C$, in fact $D \to E = C$; then $D \to E \leq_\cup B$.

(μ): Then $v = \mu\alpha.c$, and $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A, \Delta$. Then A is not an intersection type, so $A \leq_\cup B$.

$(\cap R)$: Then $A = \cap_{\underline{n}} A_i$, $C = A$, and $D = C \cup E$, for some E.

$(\cup R)$: Then $A = \cup_{\underline{n}} A_i \leq_\cup B$.

In all cases, we conclude using rule $(\leq_\cup)$.

As for the second rule, $(\leq L)$, if $A \leq B$, then either B is not a union, and then $A \leq_\cap B$, or there exists $C \in \mathcal{T}_p$ such that $A \leq_\cap C \leq_\cup B$. Assume $\Gamma \mid e : B \vdash_{\mathcal{M}} \Delta$, then this derivation ends with either:

$(Ax\text{-}L)$: Then $e = \alpha$, and $\alpha{:}D \in \Delta$ with $B \leq_\cup D$; then also $C \leq_\cup D$, so also $\Gamma \mid \alpha : C \vdash_{\mathcal{M}} \Delta$.

$(\to L)$: Then $e = v \cdot e'$, and $B = D \to E$. Since $C \leq_\cup D \to E$, in fact $D \to E = C$; then $A \leq_\cap D \to E$.

$(\tilde{\mu})$: Then $e = \tilde{\mu}x.c$, and $c : \Gamma, x{:}B \vdash_{\mathcal{M}} \Delta$. Then B is not an union type, $A \leq_\cap B$.

$(\cup L)$: Then $A = \cup_{\underline{n}} A_i$, $C = \cup_{\underline{m}} C_j$, where for every $j \in \underline{m}$ there is an $i \in \underline{n}$ such that $C_j = A_i$, and $D = \cap_{\underline{m}} C_j \cap E$, for some E. Then, by rule $(\cup L)$, also $\Gamma \mid e : \cup_{\underline{m}} C_j \vdash_{\mathcal{M}} \Delta$.

$(\cap L)$: Then $A = C$.

In all cases, we now conclude using rule $(\leq_\cap)$. □

As a direct consequence, also the following rules are admissible:

$$(\cap E):\ \frac{\Gamma \vdash v : \cap_{\underline{n}} A_i \mid \Delta}{\Gamma \vdash v : A_j \mid \Delta}(j \in \nu) \qquad (\cup E):\ \frac{\Gamma \mid e : \cup_{\underline{n}} A_i \vdash \Delta}{\Gamma \mid e : A_j \vdash \Delta}(j \in \nu)$$

One might think that being able to model $(\cap E)$ is enough to show that the interpretation of λ-terms now preserves assignable types, but this is not the case. The problem is in the fact that intersection types are not allowed in right-hand environment, as is clear from the following:

Example 5.6 Assume a derivation in $\vdash_{\mathrm{D}}$ ends with $(\to E)$: then the term in question is an application PQ, and there is a B such that $\Gamma \vdash_{\mathrm{D}} P : B \to A$ and $\Gamma \vdash_{\mathrm{D}} Q : B$. Assume, by induction, that both $\Gamma \vdash_{\mathcal{M}} [\![P]\!]^{\mathrm{T}}_{\vec{\mu}} : B \to A \mid$ and $\Gamma \vdash_{\mathcal{M}} [\![Q]\!]^{\mathrm{T}}_{\vec{\mu}} : B \mid$ are derivable: since $[\![PQ]\!]^{\mathrm{T}}_{\vec{\mu}} = \mu\alpha.\langle [\![P]\!]^{\mathrm{T}}_{\vec{\mu}} \mid [\![Q]\!]^{\mathrm{T}}_{\vec{\mu}} \cdot \alpha \rangle$

we would like to construct

$$\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash [\![P^{\eta}_{\tilde{\mu}} : B \to A \mid}}\qquad \dfrac{\boxed{\phantom{\Gamma \vdash [\![Q^{\eta}_{\tilde{\mu}} : B \mid}}\qquad \overline{\mid \alpha : A \vdash \alpha{:}A}}{\Gamma \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha : B \to A \vdash \alpha{:}A}\,(\to L)}{\langle [\![P^{\eta}_{\tilde{\mu}} \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha \rangle : \Gamma \vdash \alpha{:}A}\,(cut)}{\Gamma \vdash \mu\alpha.\langle [\![P^{\eta}_{\tilde{\mu}} \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha \rangle : A \mid}\,(\mu)$$

but A might be an intersection type $\cap_{\underline{n}} A_i$, and then we cannot derive $\mid \alpha : A \vdash_{\mathcal{M}} \alpha{:}A$. What we then perhaps *could* show is the following:

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash [\![P^{\eta}_{\tilde{\mu}} : B \to A_i \mid}}\qquad \dfrac{\boxed{\phantom{\Gamma \vdash [\![Q^{\eta}_{\tilde{\mu}} : B \mid}}\qquad \overline{\mid \alpha : A_i \vdash \alpha{:}A_i}}{\Gamma \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha : B \to A_i \vdash \alpha{:}A_i}\,(\to L)}{\langle [\![P^{\eta}_{\tilde{\mu}} \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha \rangle : \Gamma \vdash \alpha{:}A_i}\,(cut)}{\Gamma \vdash \mu\alpha.\langle [\![P^{\eta}_{\tilde{\mu}} \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha \rangle : A_i \mid}\,(\mu)}{\Gamma \vdash \mu\alpha.\langle [\![P^{\eta}_{\tilde{\mu}} \mid [\![Q^{\eta}_{\tilde{\mu}} \cdot \alpha \rangle : \cap_{\underline{n}} A_i \mid}\,(\cap R)$$

but this depends on being able to show that $\Gamma \vdash_{\mathcal{M}} [\![P^{\eta}_{\tilde{\mu}} : B \to \cap_{\underline{n}} A_i \mid$ implies $\Gamma \vdash_{\mathcal{M}} [\![P^{\eta}_{\tilde{\mu}} : B \to A_i \mid$ for all $i \in \underline{n}$, a property that holds only in a system closed under a contra-variant relation on types, which $\mathcal{M}$ is not. So it seems that, although $\mathcal{M}$ uses full BCD [11] types, it is, in fact, strict [1] in nature.

We will be able to show the type preservation result for System $\mathcal{M}^{\mathrm{c}}$ in Section 9.

Notice that typing is not preserved for the rule (η): for example, we can derive:

$$\dfrac{\dfrac{\dfrac{\overline{y{:}A \to B \vdash y : A \to B \mid}\qquad \dfrac{\overline{x{:}A \cap C \vdash x : A \mid \beta{:}B}\qquad \overline{\mid \beta : B \vdash \beta{:}B}}{x{:}A \cap C \mid x \cdot \beta : A \to B \vdash \beta{:}B}\,(\to L)}{\langle y \mid x \cdot \beta \rangle : y{:}A \to B, x{:}A \cap C \vdash \beta{:}B}\,(cut)}{y{:}A \to B, x{:}A \cap C \vdash \mu\beta.\langle y \mid x \cdot \beta \rangle : B \mid}\,(\mu)}{y{:}A \to B \vdash \lambda x \mu\beta.\langle y \mid x \cdot \beta \rangle : (A \cap C) \to B \mid}\,(\to R)$$

but cannot show $y{:}A \to B \vdash_{\mathcal{M}} y : (A \cap C) \to B \mid$; we would again need a $\leq$-relation on types that is contra-variant over arrow types.

However, we can show that $\tilde{\mu}$-reduction towards an intersection and μ-reduction towards a union are safe, as well as rule $(\to)$; we will start with the latter:

Theorem 5.7 (Soundness for rule $(\to)$) *If* $\langle \lambda x.v_1 \mid v_2 \cdot e \rangle : \Gamma \vdash_{\mathcal{M}} \Delta$, *then* $\langle v_2 \mid \tilde{\mu} x.\langle v_1 \mid e \rangle \rangle : \Gamma \vdash_{\mathcal{M}} \Delta$.

Proof: If $\langle \lambda x.v_1 \mid v_2 \cdot e \rangle : \Gamma \vdash_{\mathcal{M}} \Delta$, then, by Lemma 5.4, we have:

$$\dfrac{\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}^1_i}}{\Gamma, x{:}C_i \vdash v_1 : D_i \mid \Delta}\,(\to R)}{\Gamma \vdash \lambda x.v_1 : C_i \to D_i \mid \Delta}\,(\forall i \in \underline{n})\,(\cap R)}{\Gamma \vdash \lambda x.v_1 : \cap_n (C_i \to D_i) \mid \Delta}\,(\leq R)}{\Gamma \vdash \lambda x.v_1 : A \mid \Delta}\qquad \dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}^2_j}}{\Gamma \vdash v_2 : E_j \mid \Delta}\qquad \dfrac{\boxed{\mathcal{D}^3_j}}{\Gamma \mid e : F_j \vdash \Delta}}{\Gamma \mid v_2 \cdot e : E_j \to F_j \vdash \Delta}\,(\to L)}{\Gamma \mid v_2 \cdot e : \cup_m (E_j \to F_j) \vdash \Delta}\,(\forall j \in m)\,(\cup L)}{\Gamma \mid v_2 \cdot e : A \vdash \Delta}\,(\leq L)$$

$$\dfrac{}{\langle \lambda x.v_1 \mid v_2 \cdot e \rangle : \Gamma \vdash \Delta}\,(cut)$$

for some n and m. Notice that then either $n = 1$ or $m = 1$, or $\cap_n(C_i \to D_i) \leq \cup_m(E_j \to F_j)$, so, in particular, there exists an $i \in \underline{n}, j \in \underline{m}$ and A, B such that $C_i \to D_i = A \to B = E_j \to F_j$, so such that $E_j = C_i = A$ and $D_i = F_j = B$. But then we can derive:

$$\dfrac{\mathcal{D}_j^2 \quad \dfrac{\dfrac{\mathcal{D}_i^1 \quad \mathcal{D}_j^3}{\Gamma, x{:}A \vdash v_1 : B \mid \Delta \quad \Gamma \mid e : B \vdash \Delta}(cut)}{\langle v_1 \mid e \rangle : \Gamma, x{:}A \vdash \Delta}}{\Gamma \vdash v_2 : A \mid \Delta \quad \Gamma \mid \tilde{\mu}x.\langle v_1 \mid e \rangle : A \vdash \Delta}(\tilde{\mu})}{\langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e \rangle \rangle : \Gamma \vdash \Delta}(cut)$$

as desired. In fact, we can argue that both the derived intersection type as well as the derived union type are obsolete: since the implicit output of $\lambda x.v_1$ and the implicit input of $v_2 \cdot e$ are unique, there is no need for either intersection or union. □

The following lemmas show that a well-typed substitution of a term for a term variable and of a context for a context variable is sound. This gives a partial result for reduction, stating that a μ-contraction towards a union type A such that $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \mid \Delta$ is sound, as well as that a $\tilde{\mu}$-contraction towards an intersection type A such that $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathcal{M}} \Delta$ is.

Lemma 5.8 (Term substitution lemma) *Let* $\Gamma \vdash_{\mathcal{M}} v : B \mid \Delta$.

i) If $c : \Gamma, x{:}B \vdash_{\mathcal{M}} \Delta$, *then* $c\{v/x\} : \Gamma \vdash_{\mathcal{M}} \Delta$.

ii) If $\Gamma, x{:}B \vdash_{\mathcal{M}} v' : A \mid \Delta$, *then* $\Gamma \vdash_{\mathcal{M}} v'\{v/x\} : A \mid \Delta$.

iii) If $\Gamma, x{:}B \mid e : A \vdash_{\mathcal{M}} \Delta$, *then* $\Gamma \mid e\{v/x\} : A \vdash_{\mathcal{M}} \Delta$.

Proof: By simultaneously induction; we show only some of the cases.

($c = \langle v' \mid e' \rangle$): By Lemma 5.4, we have both $\Gamma, x{:}B \vdash_{\mathcal{M}} v' : A \mid \Delta$ and $\Gamma, x{:}B \mid e' : A \vdash_{\mathcal{M}} \Delta$ for some A. Then by induction (*ii*) and (*iii*), both $\Gamma \vdash_{\mathcal{M}} v'\{v/x\} : A \mid \Delta$ and $\Gamma \mid e'\{v/x\} : A \vdash_{\mathcal{M}} \Delta$, and $c\{v/x\} : \Gamma \vdash_{\mathcal{M}} \Delta$ follows by rule (cut).

($v' = x$): Then, by Lemma 5.4, $B \leq A$. By rule $(\leq R)$, also $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$.

($v' = \mu\alpha.c$): Then, by Lemma 5.4, there are A_i $(\forall i \in \underline{n})$ such that $\cap_{\underline{n}} A_i \leq A$, and $c : \Gamma, x{:}B \vdash_{\mathcal{M}} \alpha{:}A_i, \Delta$. By induction (*i*), $c\{v/x\} : \Gamma \vdash_{\mathcal{M}} \alpha{:}A_i, \Delta$, so $\Gamma \vdash_{\mathcal{M}} (\mu\alpha.c)\{v/x\} : A_i \mid \Delta$ by rule (μ); the result follows by $(\cap R)$ and $(\leq R)$.

($e = \alpha$): By thinning, $\Gamma \mid \alpha : A \vdash_{\mathcal{M}} \Delta$.

($e = v' \cdot e'$): Then, by Lemma 5.4, there are C_i, D_i $(\forall i \in \underline{n})$ such that $A \leq \cup_n(C_i \to D_i)$, and both $\Gamma, x{:}B \vdash_{\mathcal{M}} v' : C_i \mid \Delta$ and $\Gamma, x{:}B \mid e' : D_i \vdash_{\mathcal{M}} \Delta$ for all $i \in \underline{n}$. We have $\Gamma \vdash_{\mathcal{M}} v'\{v/x\} : C_i \mid \Delta$ by induction (*ii*) and $\Gamma \mid e'\{v/x\} : D_i \vdash_{\mathcal{M}} \Delta$ by induction (*iii*), so $\Gamma \mid (v' \cdot e')\{v/x\} : C_i \to D_i \vdash_{\mathcal{M}} \Delta$ by rule $(\to L)$; the result follows from $(\cup L)$ and $(\leq L)$. □

Notice, since left-hand environments contain only intersection types, we do not need the condition $B \in \mathcal{T}_\cap$.

Lemma 5.9 (Context substitution lemma) *Let* $\Gamma \mid e : B \vdash_{\mathcal{M}} \Delta$.

i) If $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}B, \Delta$, *then* $c\{e/\alpha\} : \Gamma \vdash_{\mathcal{M}} \Delta$.

ii) If $\Gamma \vdash_{\mathcal{M}} v : A \mid \alpha{:}B, \Delta$, *then* $\Gamma \vdash_{\mathcal{M}} v\{e/\alpha\} : A \mid \Delta$.

iii) If $\Gamma \mid e' : A \vdash_{\mathcal{M}} \alpha{:}B, \Delta$, *then* $\Gamma \mid e'\{e/\alpha\} : A \vdash_{\mathcal{M}} \Delta$.

Proof: By simultaneously induction; we show only some of the cases.

($c = \langle v' \mid e' \rangle$): By Lemma 5.4, we have both $\Gamma \vdash_{\mathcal{M}} v' : A \mid \alpha{:}B, \Delta$ and $\Gamma \mid e' : A \vdash_{\mathcal{M}} \alpha{:}B, \Delta$ for some A. Then by induction (*ii*) and (*iii*), both $\Gamma \vdash_{\mathcal{M}} v'\{e/\alpha\} : A \mid \Delta$ and $\Gamma \mid e'\{e/\alpha\} : A \vdash_{\mathcal{M}} \Delta$, and $c\{e/\alpha\} : \Gamma \vdash_{\mathcal{M}} \Delta$ by rule (cut).

$(v = x)$: By thinning, $\Gamma \vdash_{\mathcal{M}} x : A \mid \Delta$.

$(v = \lambda y.v')$: Then, by Lemma 5.4, there are C_i, D_i $(\forall i \in \underline{n})$ such that $\cap_n (C_i \to D_i) \leq A$, and $\Gamma, x{:}B, y{:}C_i \vdash_{\mathcal{M}} v'' : D_i \mid \Delta$, for all $i \in \underline{n}$. By induction (*ii*), $\Gamma, y{:}C_i \vdash_{\mathcal{M}} v''\{e/\alpha\} : D_i \mid \Delta$, so also $\Gamma \vdash_{\mathcal{M}} (\lambda y.v'')\{e/\alpha\} : C_i \to D_i \mid \Delta$ by rule $(\to R)$, and the result follows from $(\cap R)$ and $(\leq R)$.

$(e = \alpha)$: Then, by Lemma 5.4, $A \leq B$. By rule $(\leq L)$, also $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$.

$(e = v' \cdot e')$: Then, by Lemma 5.4, there are C_i, D_i $(\forall i \in \underline{n})$ such that $A \leq \cup_n (C_i \to D_i)$, with $\Gamma \vdash_{\mathcal{M}} v' : C_i \mid \alpha{:}B, \Delta$ and $\Gamma \mid e' : D_i \vdash_{\mathcal{M}} \alpha{:}B, \Delta$, for all $i \in \underline{n}$. Then $\Gamma \vdash_{\mathcal{M}} v'\{e/\alpha\} : C_i \mid \Delta$ by induction (*ii*), and $\Gamma \mid e'\{e/\alpha\} : D_i \vdash_{\mathcal{M}} \Delta$ by induction (*iii*), so $\Gamma \mid (v' \cdot e')\{e/\alpha\} : C_i \to D_i \vdash_{\mathcal{M}} \Delta$ by rule $(\to L)$; the result follows from $(\cup L)$ and $(\leq L)$. □

The promised (conditional) reduction results for both $\tilde{\mu}$ and μ reduction are implied by the first parts, respectively, of these lemmas.

Theorem 5.10 *i) If $\langle \mu\alpha.c \,|\, e\rangle : \Gamma \vdash_{\mathcal{M}} \Delta$ such that $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \mid \Delta$ is derived in an immediate sub-derivation with $A \notin \mathcal{T}_\cap$, then $c\{e/\alpha\} : \Gamma \vdash_{\mathcal{M}} \Delta$.*

ii) If $\langle v \,|\, \tilde{\mu}x.c\rangle : \Gamma \vdash_{\mathcal{M}} \Delta$ such that $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathcal{M}} \Delta$ is derived in an immediate sub-derivation with $A \notin \mathcal{T}_\cup$, then $c\{v/x\} : \Gamma \vdash_{\mathcal{M}} \Delta$.

Proof: i) If $\langle \mu\alpha.c \,|\, e\rangle : \Gamma \vdash_{\mathcal{M}} \Delta$, then there exists A such that $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \mid \Delta$ and $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$. If $A \notin \mathcal{T}_\cap$, then, by Lemma 5.4, $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A, \Delta$, and $c\{e/\alpha\} : \Gamma \vdash_{\mathcal{M}} \Delta$ follows by Lemma 5.9.

ii) Similar, using Lemma 5.8. □

Notice that, in part (*i*), if $A = \cap_{\underline{n}} A_i$, then, as a result of $(\cap R)$ and (μ), also $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A_i, \Delta$; we now cannot show the result, since all A_i might be necessary for $\Gamma \mid e : \cap_{\underline{n}} A_i \vdash_{\mathcal{M}} \Delta$, so we cannot derive $\Gamma \mid e : A_i \vdash_{\mathcal{M}} \Delta$ (see Example 7.1).

As a consequence, using run-time type checking it is possible to build a sound implementation of this system; we will show in Section 7 that we cannot achieve a stronger result.

6 Loss of completeness

The main goal of the authors of [18] was to characterise strong normalisation; an important property in the proof for that result in systems with abstraction is that the system is complete (see the proof in [1] for the characterisation of strong normalisation in the context of the λ-calculus, as well as [3, 5] for two new proofs), so we would need to show that, if $c_1 \to c_2$ and $c_2 : \Gamma \vdash_{\mathcal{M}} \Delta$ then also $c_1 : \Gamma \vdash_{\mathcal{M}} \Delta$. But, surprisingly, and considering it is the minimal system we identified above, the presence of intersection and union notwithstanding, System $\mathcal{M}$ is *not* complete, not even for non-cancelling reduction, where $\top$ and $\bot$ could be useful.

To show this, we will give a number of counter examples. We give two terms c_1 and c_2 such that $c_1 \to c_2$, and give a derivation for c_2 that we cannot expand to a derivation for c_1.

Example 6.1 Take the reduction $\langle \lambda x.v_1 \,|\, v_2 \cdot e\rangle \to \langle v_2 \,|\, \tilde{\mu}x.\langle v_1 \,|\, e\rangle\rangle$ via rule $(\to)$. Assume that for the right-hand side we can derive:

$$\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\Gamma \vdash v_2 : \cup_{\underline{n}} A_i \mid \Delta} \qquad \dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_2^i}}{\Gamma, x{:}A_i \vdash v_1 : B_i \mid \Delta} \quad \dfrac{\boxed{\mathcal{D}_3^i}}{\Gamma \mid e : B_i \vdash \Delta}\,(cut)}{\langle v_1 \,|\, e\rangle : \Gamma, x{:}A_i \vdash \Delta}}{\Gamma \mid \tilde{\mu}x.\langle v_1 \,|\, e\rangle : A_i \vdash \Delta}\,(\tilde{\mu})}{\Gamma \mid \tilde{\mu}x.\langle v_1 \,|\, e\rangle : \cup_{\underline{n}} A_i \vdash \Delta}\,(\forall i \in \underline{n})\,(\cup L)}{\langle v_2 \,|\, \tilde{\mu}x.\langle v_1 \,|\, e\rangle\rangle : \Gamma \vdash \Delta}\,(cut)$$

(note that for v_2 we can pick a term that truly needs all types mentioned in the union as in Example 4.7). Given the collection of sub-derivations, there is little choice when constructing the derivation for $\langle \lambda x.v_1 | v_2 \cdot e \rangle$. Notice that we are looking for a type C to derive

$$\dfrac{\boxed{} \quad \boxed{}}{\langle \lambda x.v_1 | v_2 \cdot e \rangle : \Gamma \vdash \Delta}\,(cut)$$

Focussing on the right-hand sub-derivation, not considering rules $(\cup L)$ and $(\cap L)$ (since these would require a sub-derivation for $v_2 \cdot e$ as well), this gives:

$$\dfrac{\boxed{\phantom{\Gamma \vdash \lambda x.v_1 : \cup_{\underline{n}} A_i \to D \mid \Delta}} \qquad \dfrac{\boxed{\mathcal{D}_1\ \Gamma \vdash v_2 : \cup_{\underline{n}} A_i \mid \Delta} \quad \boxed{\mathcal{D}_3\ \Gamma \mid e : D \vdash \Delta}}{\Gamma \mid v_2 \cdot e : \cup_{\underline{n}} A_i \to D \vdash \Delta}\,(\to L)}{\langle \lambda x.v_1 | v_2 \cdot e \rangle : \Gamma \vdash \Delta}\,(cut)$$

so $C = \cup_{\underline{n}} A_i \to D$; note that, *a priori*, since the system lacks choice, derivation $\mathcal{D}_1$ cannot be decomposed, and that D in $\mathcal{D}_3$ is composed out of the B_i, either via application of rule $(\cap L)$ or $(\cup L)$. Focussing now on the left-hand sub-derivation, since this derives an arrow type, we get:

$$\dfrac{\dfrac{\boxed{\mathcal{D}_2\ \Gamma, x{:}\cup_{\underline{n}} A_i \vdash v_1 : D \mid \Delta}}{\Gamma \vdash \lambda x.v_1 : \cup_{\underline{n}} A_i \to D \mid \Delta}\,(\to R) \qquad \dfrac{\boxed{\mathcal{D}_1\ \Gamma \vdash v_2 : \cup_{\underline{n}} A_i \mid \Delta} \quad \boxed{\mathcal{D}_3\ \Gamma \mid e : D \vdash \Delta}}{\Gamma \mid v_2 \cdot e : \cup_{\underline{n}} A_i \to D \vdash \Delta}\,(\to L)}{\langle \lambda x.v_1 | v_2 \cdot e \rangle : \Gamma \vdash \Delta}\,(cut)$$

Now the derivation $\mathcal{D}_2$ must be constructed from the $\mathcal{D}_2^i$ by either rule $(\cap R)$ or $(\cup R)$; this is not possible, since neither creates a union type for x. As suggested in Remark 4.3, the last step for $\mathcal{D}_2$ requires a rule like

$$(\,) : \dfrac{\Gamma_i, x{:}A_i \vdash v : B_i \mid \Delta_i \,(\forall i \in \underline{n})}{\cap_{\underline{n}} \Gamma_i, x{:}\cup_{\underline{n}} A_i \vdash v : \cup_{\underline{n}} B_i \mid \cup_{\underline{n}} \Delta_i}$$

which is not admissible in System $\mathcal{M}$. As mentioned above, to achieve soundness, [18] treats implicit and explicit variables *differently*, thereby diminishing the necessary expressiveness of the system.

As shown in the next example, also μ-expansion is problematic.

Example 6.2 Let $c_1 = \langle \mu\alpha.\langle x | \alpha \rangle \mid \tilde{\mu}y.\langle y | y \cdot \beta \rangle \rangle$, and $c_2 = \langle x \mid \tilde{\mu}y.\langle y | y \cdot \beta \rangle \rangle$, and notice that $c_1 \to_\mu c_2$. Let $\mathcal{D}$ be the derivation for $\tilde{\mu}y.\langle y | y \cdot \beta \rangle$ given in Example 4.7, then we can type c_2 as follows:

$$\dfrac{\dfrac{\overline{x{:}C \to D \vdash x : C \to D \mid} \qquad \overline{x{:}C \vdash x : C \mid}}{x{:}(C \to D) \cap C \vdash x : (C \to D) \cap C \mid}\,(\cap R) \qquad \boxed{\mathcal{D}\ \mid \tilde{\mu}y.\langle y | y \cdot \beta \rangle : (C \to D) \cap C \vdash \beta{:}D}}{\langle x \mid \tilde{\mu}y.\langle y | y \cdot \beta \rangle \rangle : x{:}(C \to D) \cap C \vdash \beta{:}D}\,(cut)$$

We cannot derive c_2's typing for c_1. Extracting the context $\tilde{\mu}y.\langle y | y \cdot \beta \rangle$ from the command

$\langle x \,|\, x{\cdot}\tilde{\mu}y.\langle y|y{\cdot}\beta\rangle\rangle$ would create:

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{x{:}C{\to}D \vdash x : C{\to}D \,|} \quad \overline{x{:}C \vdash x : C \,|}}{x{:}(C{\to}D){\cap}C \vdash x : (C{\to}D){\cap}C \,|}\,(\cap R) \quad \dfrac{}{|\,\alpha : (C{\to}D){\cap}C \vdash \alpha{:}(C{\to}D){\cap}C}\,(?)}{\langle x|\alpha\rangle : x{:}(C{\to}D){\cap}C \vdash \beta{:}D, \alpha{:}(C{\to}D){\cap}C}\,(cut)}{x{:}(C{\to}D){\cap}C \vdash \mu\alpha.\langle x|\alpha\rangle : (C{\to}D){\cap}C \,|\, \beta{:}D}\,(\mu) \quad \begin{array}{c}\vdots\\ \mathcal{D}\\ \vdots\end{array}}{\langle \mu\alpha.\langle x|\alpha\rangle \,|\, \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle\rangle : x{:}(C{\to}D){\cap}C \vdash \beta{:}D}\,(cut)$$

which is illegal: there is no way we can justify step (?) - remember that system $\mathcal{M}$ does not permit intersection types for context variables. So there is a completeness problem with respect to $\to_{\mu}$.

We can also give a similar example that shows that union types can be problematic as well for expansion towards a $\tilde{\mu}$-reduction.

Example 6.3 Let $c_1 = \langle \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \tilde{\mu}y.\langle y|\beta\rangle\rangle$ and $c_2 = \langle \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \beta\rangle$, and notice that $c_1 \to_{\tilde{\mu}} c_2$. Let $\mathcal{D}$ be the derivation for $\vdash_{\mathcal{M}} \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A \cup (A{\to}B) \,|$ from Example 4.7. Then for c_2 we can derive the typing:

$$\dfrac{\dfrac{\boxed{\mathcal{D}}}{\vdash \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A \cup (A{\to}B) \,|} \quad \dfrac{\overline{|\,\beta : A \vdash \beta{:}A} \quad \overline{|\,\beta : A{\to}B \vdash \beta{:}A{\to}B}}{|\,\beta : A \cup (A{\to}B) \vdash \beta{:}A \cup (A{\to}B)}\,(\cup L)}{\langle \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \beta\rangle : \vdash \beta{:}A \cup (A{\to}B)}\,(cut)$$

and extracting the context $\mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle$ would yield:

$$\dfrac{\dfrac{\boxed{\mathcal{D}}}{\vdash \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle : A \cup (A{\to}B) \,|} \quad \dfrac{\dfrac{\dfrac{}{y{:}A \cup (A{\to}B) \vdash y : A \cup (A{\to}B) \,|}\,(?) \quad \begin{array}{c}\vdots\end{array} \quad \dfrac{\overline{|\,\beta : A \vdash \beta{:}A} \quad \overline{|\,\beta : A{\to}B \vdash \beta{:}A{\to}B}}{|\,\beta : A \cup (A{\to}B) \vdash \beta{:}A \cup (A{\to}B)}\,(\cup L)}{\langle y|\beta\rangle : y{:}A \cup (A{\to}B) \vdash \beta{:}A \cup (A{\to}B)}\,(cut)}{|\,\tilde{\mu}y.\langle y|\beta\rangle : A \cup (A{\to}B) \vdash \beta{:}A \cup (A{\to}B)}\,(\tilde{\mu})}{\langle \mu\delta.\langle \lambda x\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \tilde{\mu}y.\langle y|\beta\rangle\rangle : \vdash \beta : A \cup (A{\to}B)}\,(cut)$$

which, again, is not a correct derivation: step (?) cannot be justified, given the restriction to intersection types for term-variables, so there is also a completeness problem with respect to $\to_{\tilde{\mu}}$.

This lack of completeness is relatively easy to correct, by adding the rules () and () (see Definition 9.1), and to allow all types in both left and right-hand environments; this will be the approach of Section 9. Unfortunately, this will then jeopardise the substitution lemmae 5.8 and 5.9, so we will have to abandon this solution when looking for a soundness result.

7 Loss of soundness

Contrary to the efforts made, perhaps surprisingly, System $\mathcal{M}$ is *also not sound*: we cannot show that, if $c_1 \to c_2$ and $c_1 : \Gamma \vdash_{\mathcal{M}} \Delta$ then also $c_2 : \Gamma \vdash_{\mathcal{M}} \Delta$. In fact, it fails for the two main reduction rules, $\to_{\mu}$ and $\to_{\tilde{\mu}}$, as we will illustrate in the next two examples, and is mainly due to the fact that $\Gamma \vdash_{\mathcal{M}} \mu\alpha.c : A \,|\, \Delta$ does not always imply $c : \Gamma \vdash_{\mathcal{M}} \alpha{:}A, \Delta$, and $\Gamma \,|\, \tilde{\mu}x.c : A \vdash_{\mathcal{M}} \Delta$

does not always imply $c : \Gamma, x{:}A \vdash_{\mathcal{M}} \Delta$. This was in part also reported on in [19], and was the motivation for the restriction made in the system presented in that paper; as we will show in Section 8, this was not enough.

Example 7.1 Let $\begin{array}{rcl} c_1 & = & \langle \mu\alpha.\langle x | x{\cdot}\alpha\rangle \,|\, \tilde{\mu}y.\langle y | y{\cdot}\beta\rangle\rangle \\ c_2 & = & \langle x \,|\, x{\cdot}(\mu\alpha.\langle x | x{\cdot}\alpha\rangle{\cdot}\beta)\rangle \\ c_3 & = & \langle x \,|\, x{\cdot}\tilde{\mu}y.\langle y | y{\cdot}\beta\rangle\rangle \end{array}$

It is straightforward to verify that $c_1 \to_{\mathrm{N}} c_2$

$$\begin{array}{rcll} c_1 & = & \langle \mu\alpha.\langle x | x{\cdot}\alpha\rangle \,|\, \tilde{\mu}y.\langle y | y{\cdot}\beta\rangle\rangle & \to_{\tilde{\mu}} \\ & & \langle \mu\alpha.\langle x | x{\cdot}\alpha\rangle \,|\, (\mu\alpha.\langle x | x{\cdot}\alpha\rangle){\cdot}\beta\rangle & \to_{\mu} \\ & & \langle x \,|\, x{\cdot}(\mu\alpha.\langle x | x{\cdot}\alpha\rangle{\cdot}\beta)\rangle & = \quad c_2 \end{array}$$

and $c_1 \to_{\mathrm{v}} c_3$ in one step via $\to_{\mu}$. Let $\mathcal{D}_1$ be the derivation for $\mid \tilde{\mu}y.\langle y | y{\cdot}\beta\rangle : (C{\to}D){\cap}C \vdash_{\mathcal{M}} \beta{:}D$ from Example 4.7, $\mathcal{D}_2$ be the derivation

$$\dfrac{\dfrac{}{x{:}A{\to}C \vdash x : A{\to}C \mid} \qquad \dfrac{\dfrac{}{x{:}A \vdash x : A \mid} \qquad \dfrac{}{\mid \alpha : C \vdash \alpha{:}C}}{x{:}A \mid x{\cdot}\alpha : A{\to}C \vdash \alpha{:}C}\,(\to L)}{\dfrac{\langle x | x{\cdot}\alpha\rangle : x{:}A \cap (A{\to}C) \vdash \alpha{:}C}{x{:}A \cap (A{\to}C) \vdash \mu\alpha.\langle x | x{\cdot}\alpha\rangle : C \mid}\,(\mu)}\,(cut)$$

and $\mathcal{D}_3$ be

$$\dfrac{\dfrac{}{x{:}A{\to}C{\to}D \vdash x : A{\to}C{\to}D \mid} \qquad \dfrac{\dfrac{}{x{:}A \vdash x : A \mid} \qquad \dfrac{}{\mid \alpha : C{\to}D \vdash \alpha{:}C{\to}D}}{x{:}A \mid x{\cdot}\alpha : A{\to}C{\to}D \vdash \alpha{:}C{\to}D}\,(\to L)}{\dfrac{\langle x | x{\cdot}\alpha\rangle : x{:}A \cap (A{\to}C{\to}D) \vdash \alpha{:}C{\to}D}{x{:}A \cap (A{\to}C{\to}D) \vdash \mu\alpha.\langle x | x{\cdot}\alpha\rangle : C{\to}D \mid}\,(\mu)}\,(cut)$$

(notice that the main difference between $\mathcal{D}_2$ and $\mathcal{D}_3$ is the type for α).

Take the following typing for c_1:

$$\dfrac{\dfrac{\boxed{\mathcal{D}_3} \qquad \boxed{\mathcal{D}_2}}{x{:}A \cap (A{\to}C{\to}D) \vdash \mu\alpha.\langle x | x{\cdot}\alpha\rangle : C{\to}D \mid \quad x{:}A \cap (A{\to}C) \vdash \mu\alpha.\langle x | x{\cdot}\alpha\rangle : C \mid}\,(\cap R)}{\begin{array}{l} x{:}A \cap (A{\to}C) \cap (A{\to}C{\to}D) \vdash \mu\alpha.\langle x | x{\cdot}\alpha\rangle : (C{\to}D){\cap}C \mid \\ \vdots \qquad\qquad \boxed{\mathcal{D}_1} \\ \qquad\qquad \mid \tilde{\mu}y.\langle y | y{\cdot}\beta\rangle : (C{\to}D){\cap}C \vdash \beta{:}D \end{array}\,(cut)}$$
$$\langle \mu\alpha.\langle x | x{\cdot}\alpha\rangle \,|\, \tilde{\mu}y.\langle y | y{\cdot}\beta\rangle\rangle : x{:}A \cap (A{\to}C) \cap (A{\to}C{\to}D) \vdash \beta{:}D$$

Let us look at soundness starting from this particular derivation. Since for c_2 we can derive:

$$\dfrac{\begin{array}{l} \dfrac{}{x{:}A{\to}C{\to}D \vdash x : A{\to}C{\to}D \mid} \qquad \boxed{\mathcal{D}_2} \qquad \dfrac{}{\mid \beta : D \vdash \beta{:}D} \\ \vdots \qquad\qquad\qquad x{:}A \cap (A{\to}C) \mid \mu\alpha.\langle x | x{\cdot}\alpha\rangle{\cdot}\beta : C{\to}D \vdash \beta{:}D \\ \qquad \dfrac{x{:}A \vdash x : A \mid}{x{:}A \cap (A{\to}C) \mid x{\cdot}(\mu\alpha.\langle x | x{\cdot}\alpha\rangle{\cdot}\beta) : A{\to}C{\to}D \vdash \beta{:}D}\,(\to L) \end{array}\,(\to L)}{\langle x \,|\, x{\cdot}(\mu\alpha.\langle x | x{\cdot}\alpha\rangle{\cdot}\beta)\rangle : x{:}A \cap (A{\to}C) \cap (A{\to}C{\to}D) \vdash \beta{:}D}\,(cut)$$

it will be obvious that, when reducing c_1 to c_2, there is no problem. When reducing $c_1 \to_{\mathrm{v}} c_3$ on the other hand, contracting the $\mu\alpha$ redex in c_1, the context $\tilde{\mu}y.\langle y | y{\cdot}\beta\rangle$ will be inserted for the context variable α. But we cannot derive:

$$\langle x \,|\, x{\cdot}\tilde{\mu}y.\langle y | y{\cdot}\beta\rangle\rangle : x{:}A \cap (A{\to}C) \cap (A{\to}C{\to}D) \vdash_{\mathcal{M}} \beta{:}D$$

We would expect the derivation $\mathcal{D}_1$ to be inserted for α (in one way or another; since intersection and union are involved in this system, it might be that a derivation gets divided into pieces before being inserted) in the derivation for $\langle x \,|\, x{\cdot}\alpha \rangle$. However, notice that we do not have a sub-derivation for $\mid \alpha : (C{\to}D){\cap}C \vdash_{\mathcal{M}} \alpha{:}(C{\to}D){\cap}C$, but two, for $\mid \alpha : C \vdash_{\mathcal{M}} \alpha{:}C$ and $\mid \alpha : C{\to}D \vdash_{\mathcal{M}} \alpha{:}C{\to}D$. Now we cannot split the intersection needed to derive $\mathcal{D}_1$: both types C and $C{\to}D$ are needed to type $\tilde{\mu}y.\langle y|y{\cdot}\beta\rangle$, and no rule allows the elimination of an intersection type for a context (nor of a union for a term), so we cannot derive neither $\mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : C{\to}D \vdash_{\mathcal{M}} \beta{:}D$, nor $\mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : C \vdash_{\mathcal{M}} \beta{:}D$; we can also not derive

$$\mid \alpha : C{\to}D \vdash_{\mathcal{M}} \alpha{:}\,(C{\to}D){\cap}C$$

which could have solved the problem, nor $\mid \alpha : C \vdash_{\mathcal{M}} \alpha{:}\,(C{\to}D){\cap}C$. Given the structure of the rules in system $\mathcal{M}$, this implies that for c_3 we can at most derive:

$$\dfrac{\dfrac{}{x{:}A{\to}(C{\to}D){\cap}C \vdash x : A{\to}(C{\to}D){\cap}C \mid} \quad \dfrac{\dfrac{}{x{:}A \vdash x : A \mid} \quad \boxed{\mathcal{D}_1}\ \mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : (C{\to}D){\cap}C \vdash \beta{:}D}{x{:}A \mid x{\cdot}\tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : A{\to}(C{\to}D){\cap}C \vdash \beta{:}D}\,(\to L)}{\langle x \,|\, x{\cdot}\tilde{\mu}y.\langle y|y{\cdot}\beta\rangle\rangle : x{:}A{\cap}(A{\to}(C{\to}D){\cap}C) \vdash \beta{:}D}\,(cut)$$

Notice that this is incomparable to the desired result: there is no way we can transform A, $A{\to}C$ and $A{\to}C{\to}D$ (*i.e.* $A{\cap}(A{\to}C){\cap}(A{\to}C{\to}D)$) into $A{\cap}(A{\to}(C{\to}D){\cap}C)$. We would need the contra-variant relation $\leq$ on types to achieve this; since that relation is not considered in [18], this is not possible. So we have a type assignment for c_1 that is a valid typing also for c_2, but not for c_3.

Note that the above typing for c_3 shows the necessity of allowing intersection on the right of arrow types. Remark also that it is impossible to derive $c_1 : x{:}A{\cap}(A{\to}(C{\to}D){\cap}C) \vdash_{\mathcal{M}} \beta{:}D$, since that would require

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}A{\to}(C{\to}D){\cap}C \vdash x : A{\to}(C{\to}D){\cap}C \mid} \quad \vdots}{\langle x|x{\cdot}\alpha\rangle : x{:}A{\cap}(A{\to}(C{\to}D){\cap}C) \vdash \alpha{:}(C{\to}D){\cap}C, \beta{:}D}\,(cut)}{x{:}A{\cap}(A{\to}(C{\to}D){\cap}C) \vdash \mu\alpha.\langle x|x{\cdot}\alpha\rangle : (C{\to}D){\cap}C \mid \beta{:}D}\,(\mu) \quad \boxed{\mathcal{D}}\ \mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle : (C{\to}D){\cap}C \vdash \beta{:}D}{\langle \mu\alpha.\langle x|x{\cdot}\alpha\rangle \mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle\rangle : x{:}A{\cap}(A{\to}(C{\to}D){\cap}C) \vdash \beta{:}D}\,(cut)$$

$$\dfrac{\dfrac{}{x{:}A \vdash x : A \mid} \quad \dfrac{}{\mid \alpha : (C{\to}D){\cap}C \vdash \alpha{:}(C{\to}D){\cap}C}\,(?)}{x{:}A \mid x{\cdot}\alpha : A{\to}(C{\to}D){\cap}C \vdash \alpha{:}(C{\to}D){\cap}C, \beta{:}D}\,(\to L)$$

but this is not valid in $\mathcal{M}$; this gives another counterexample against completeness, which is essentially the same as Example 6.2.

We will now give an example that shows that union types can be problematic as well.

Example 7.2 Take $\begin{array}{rcl} c_1 & = & \langle \mu\delta.\langle \lambda x.\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle \\ c_2 & = & \langle \lambda v.\mu\delta.\langle \lambda x.\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \gamma\rangle \\ c_3 & = & \langle \lambda w x.\mu\beta.\langle x \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle \,|\, \gamma\rangle \end{array}$

Then $c_1 \to c_2$ in one step, and $c_1 \to c_3$:

$$\begin{array}{rcll} c_1 & = & \langle \mu\delta.\langle \lambda x.\mu\beta.\langle x|\delta\rangle \,|\, \delta\rangle \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle & \to_{\mu} \\ & & \langle \lambda x.\mu\beta.\langle x \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle & \to_{\tilde{\mu}} \\ & & \langle \lambda v x.\mu\beta.\langle x \,|\, \tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle \,|\, \gamma\rangle & = \quad c_3 \end{array}$$

Let $\mathcal{D} :: \vdash_{\mathcal{M}} \mu\delta.\langle \lambda x \mu\beta.\langle x|\delta\rangle\,|\delta\rangle : A \cup (A{\to}B)\,|$ be as in Example 4.7, then we can type c_1 as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{v{:}C, z{:}A \vdash z : A\,|}}{z{:}A \vdash \lambda v.z : C{\to}A\,|}(\to R) \quad \dfrac{}{|\,\gamma : C{\to}A \vdash \gamma{:}C{\to}A}\ \vdots}{\langle \lambda v.z|\gamma\rangle : z{:}A \vdash \gamma{:}C{\to}A}(cut)}{|\,\tilde{\mu}z.\langle \lambda v.z|\gamma\rangle : A \vdash \gamma{:}C{\to}A}(\tilde{\mu}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{v{:}C, z{:}A{\to}B \vdash z : A{\to}B\,|}}{z{:}A{\to}B \vdash \lambda v.z : C{\to}A{\to}B\,|}(\to R) \quad \dfrac{}{|\,\gamma : C{\to}A{\to}B \vdash \gamma{:}C{\to}A{\to}B}\ \vdots}{\langle \lambda v.z|\gamma\rangle : z{:}A{\to}B \vdash \gamma{:}C{\to}A{\to}B}(cut)}{|\,\tilde{\mu}z.\langle \lambda v.z|\gamma\rangle : A{\to}B \vdash \gamma{:}C{\to}A{\to}B}(\tilde{\mu})}{|\,\tilde{\mu}z.\langle \lambda v.z|\gamma\rangle : A \cup (A{\to}B) \vdash \gamma{:}(C{\to}A){\cup}(C{\to}A{\to}B)}(\cup L)$$

$$\dfrac{\boxed{\mathcal{D}}\ \vdots}{\vdash \mu\delta.\langle \lambda x \mu\beta.\langle x|\delta\rangle\,|\delta\rangle : A \cup (A{\to}B)\,|}$$

$$\dfrac{\vdash \mu\delta.\langle \lambda x \mu\beta.\langle x|\delta\rangle\,|\delta\rangle : A \cup (A{\to}B)\,|}{\langle \mu\delta.\langle \lambda x \mu\beta.\langle x|\delta\rangle\,|\delta\rangle\,|\tilde{\mu}z.\langle \lambda v.z|\gamma\rangle\rangle : \vdash \gamma{:}(C{\to}A){\cup}(C{\to}A{\to}B)}(cut)$$

Consider the CBN reduction to c_2; we now expect the derivation $\mathcal{D}$ to be inserted for z in the derivation for $\langle \lambda v.z|\gamma\rangle$. But we cannot derive:

$$\langle \lambda v \mu\delta.\langle \lambda x \mu\beta.\langle x|\delta\rangle\,|\delta\rangle\,|\gamma\rangle : \vdash_{\mathcal{M}} \gamma{:}(C{\to}A){\cup}(C{\to}A{\to}B)$$

Notice that we have two sub-derivations in $\mathcal{D}_2$, each using a different type for z; since we cannot separate the components of the union type derived in $\mathcal{D}_1$, we cannot construct a derivation for the contractum.

These two examples show that we also cannot achieve soundness in System $\mathcal{M}$.

8 The system $\mathcal{M}^{\cap}$ of [19]

A typical reason to set up a notion of typing with intersection types is to prove a characterisation result for strong normalisation, and that was one of the motivations for [18]. In order to bring in the right expressiveness to achieve this result, in an attempt to deal with the problem that the symmetry of $\overline{\lambda}\mu\tilde{\mu}$ poses on a proof, the authors of that paper decided to modify the system they defined in [18] quite drastically, adding negation and expressing union types via intersection and negation; this conforms to their view that union is only of limited use. This is accompanied by transforming derivable statements from sequent style to natural deduction style, writing only a environment to the left and to have a single result (conclusion).

We will briefly revisit that system here (adapting notation and syntax), and show we can give counterexamples for both completeness and soundness for that modified system as well.

Remark 8.1 It is important to point out that [19] presents a slightly different version of $\overline{\lambda}\mu\tilde{\mu}$'s reduction relation (Definition 1.2): instead of rule $(\to)$ it uses [16]'s rule

$$(\to') : \langle \lambda x.v_1 | v_2 \cdot e\rangle \to \langle v_1\{v_2/x\}\,|e\rangle$$

Although this is the desirable reduction result when considering the command $\langle \lambda x.v_1|v_2\cdot e\rangle$ an application, it limits $\overline{\lambda}\mu\tilde{\mu}$'s original reduction relation. For example, for the full reduction as we consider here, we can reduce as follows:

$$\langle \lambda y.\mu\beta.\langle y|y\cdot\beta\rangle\,|\,\mu\gamma.\langle x|x\cdot\gamma\rangle\cdot\alpha\rangle \;\to\; \langle \mu\gamma.\langle x|x\cdot\gamma\rangle\,|\,\tilde{\mu}y.\langle \mu\beta.\langle y|y\cdot\beta\rangle\,|\alpha\rangle\rangle$$

Now this latter term can be reduced in *two* ways: the first is:

$$\langle \mu\gamma.\langle x|x\cdot\gamma\rangle\,|\,\tilde{\mu}y.\langle \mu\beta.\langle y|y\cdot\beta\rangle\,|\alpha\rangle\rangle \;\to_{\mu}\; \langle x\,|\,x\cdot\tilde{\mu}y.\langle \mu\beta.\langle y|y\cdot\beta\rangle\,|\alpha\rangle\rangle$$

which corresponds to a CBV reduction. The other contraction gives:

$$\langle \mu\gamma.\langle x|x\cdot\gamma\rangle \,|\, \tilde{\mu}y.\langle \mu\beta.\langle y|y\cdot\beta\rangle \,|\, \alpha\rangle\rangle \;\to_{\tilde{\mu}}\; \langle \mu\beta.\langle \mu\gamma.\langle x|x\cdot\gamma\rangle \,|\, \mu\gamma.\langle x|x\cdot\gamma\rangle\cdot\beta\rangle \,|\, \alpha\rangle$$

which corresponds to a CBN reduction. Using the reduction rule $(\to')$, we can only reduce:

$$\langle \lambda y.\mu\beta.\langle y|y\cdot\beta\rangle \,|\, \mu\gamma.\langle x|x\cdot\gamma\rangle\cdot\alpha\rangle \;\to\; \langle \mu\beta.\langle \mu\gamma.\langle x|x\cdot\gamma\rangle \,|\, \mu\gamma.\langle x|x\cdot\gamma\rangle\cdot\beta\rangle \,|\, \alpha\rangle$$

So [19] partially (*i.e.* essentially) restricts reduction to CBN.

The notion of type assignment defined in [19] is in fact inspired by the strict system of [1]; it uses a rule (ax) expressing $(\cap E)$ for variables, and hides $(\cap I)$ in another rule.

Definition 8.2 (SYSTEM $\mathcal{M}^{\cap}$ [19]) *i*) Types are defined by

$$A, B \;::=\; \varphi \mid A\to B \mid A\cap B \mid \neg A$$

Intersection is associative and commutative, and $\neg(\neg A) = A$.

ii) The typing rules are defined by (changing it again to combine environments in rules):

$$(\mathrm{ax}):\ \frac{}{\Sigma, x{:}\cap_{\underline{n}} A_i \vdash x : A_i} \qquad (\mathrm{cut}):\ \frac{\Sigma_1 \vdash v : A \quad \Sigma_2 \vdash e : \neg A}{\Sigma_1\cap\Sigma_2 \vdash \langle v|e\rangle : \bot}$$

$$(\to\mathrm{r}):\ \frac{\Sigma, x{:}A \vdash v : B}{\Sigma \vdash \lambda x.v : A\to B} \qquad (\to\mathrm{e}):\ \frac{\Sigma_i \vdash v : A_i\ (\forall i\in\underline{n}) \quad \Sigma \vdash e : \neg B}{\cap_n\Sigma_i\cap\Sigma \vdash v\cdot e : \neg((\cap_{\underline{n}} A_i)\to B)}$$

$$(\mu):\ \frac{\Sigma, \alpha{:}A \vdash c : \bot}{\Sigma \vdash \mu\alpha.c : \neg A} \qquad (\tilde{\mu}):\ \frac{\Sigma, x{:}A \vdash c : \bot}{\Sigma \vdash \tilde{\mu}x.c : \neg A}$$

We will use $\vdash_{\mathcal{M}^{\cap}}$ for this notion.

Notice that union types are omitted, but that they are 'silently' present as negated intersection types; it is the view of the authors of [19] that union is *not* the dual of intersection, a view we do not share here. Moreover, notice that rule $(\cap R)$ is missing altogether: this in particular creates a problem with the completeness result that collapses now also for other reasons than the one already exposed in Example 6.1 for System $\mathcal{M}$. There is only an implicit $(\cap I)$ in rule $(\to\mathrm{e})$, much as there was one in rule $(\to E)$ in the first presentation of the strict type assignment system in [1]; that system, however, also had an implicit general $(\cap I)$ rule that is missing here.

We first show that completeness does not hold:

Example 8.3 Notice that, since the derivation in Example 7.1 for c_1 uses rule $(\cap R)$, we cannot simulate that derivation in $\mathcal{M}^{\cap}$, so cannot derive

$$x{:}A\cap(A\to C)\cap(A\to C\to D), \beta{:}\neg D \vdash \langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \tilde{\mu}y.\langle y|y\cdot\beta\rangle\rangle : \bot$$

But we can, of course, derive that typing for c_2:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}A\to C\to D \vdash x : A\to C\to D} \quad \vdots \quad \dfrac{\dfrac{}{x{:}A\to C \vdash x : A\to C} \quad \dfrac{\dfrac{}{x{:}A \vdash x : A} \quad \dfrac{}{\alpha{:}\neg C \vdash \alpha : \neg C}}{x{:}A, \alpha{:}\neg C \vdash x\cdot\alpha : \neg(A\to C)}(\to\mathrm{e})}{x{:}A\cap(A\to C), \alpha{:}\neg C \vdash \langle x|x\cdot\alpha\rangle : \bot}(cut)}{x{:}A\cap(A\to C) \vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : C}(\mu) \quad \dfrac{}{\beta{:}\neg D \vdash \beta : \neg D}}{x{:}A\cap(A\to C), \beta{:}\neg D \vdash \mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta : \neg(C\to D)}(\to\mathrm{e})}{x{:}A\cap(A\to C), \beta{:}\neg D \vdash x\cdot(\mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta) : \neg(A\to C\to D)}(\to\mathrm{e})}{x{:}A\cap(A\to C)\cap(A\to C\to D) \vdash \langle x|x\cdot(\mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta)\rangle : \bot}(cut)$$

This now gives an immediate counter example against completeness: c_1 reduces to c_2, but not every typing for c_2 is valid for c_1.

As for the counterexamples given in Section 7 for System $\mathcal{M}^{\cap\cup}$, as intended we cannot represent Example 7.1 nor Example 7.2, since both depend on rule $(\cap R)$ which cannot be modelled. This notwithstanding, we can show that soundness also fails.

Surprisingly, although no intersection type is ever derived in this system (after all, only strict types are derived for variables, and rule $(\cap R)$ is missing), they are now implicitly present in the system, but for context-variables, whereas that was explicitly excluded for system $\mathcal{M}$.

Example 8.4 Take, again, c_1 as in Example 7.1. First we construct $\mathcal{D}_1$:

$$\dfrac{\dfrac{}{x{:}A\to((C\to D)\cap C)\vdash x : A\to((C\to D)\cap C)} \quad \dfrac{\dfrac{}{x{:}A\vdash x : A} \quad \dfrac{}{\alpha{:}\neg((C\to D)\cap C)\vdash \alpha : \neg((C\to D)\cap C)}}{x{:}A,\alpha{:}\neg((C\to D)\cap C)\vdash x\cdot\alpha : \neg(A\to((C\to D)\cap C))}(\to\mathrm{e})}{x{:}A\cap(A\to(C\to D)\cap C),\alpha{:}\neg((C\to D)\cap C)\vdash \langle x|x\cdot\alpha\rangle : \bot}(cut)$$
$$\dfrac{}{x{:}A\cap(A\to(C\to D)\cap C)\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : (C\to D)\cap C}(\mu)$$

Note that we cannot derive the corresponding result in $\mathcal{M}$, which would be

$$\dfrac{\dfrac{}{x{:}A\to(C\to D)\cap C\vdash_{\mathcal{M}} x : A\to(C\to D)\cap C\,|} \quad \dfrac{\dfrac{}{x{:}A\vdash_{\mathcal{M}} x : A\,|} \quad \dfrac{}{|\,\alpha : (C\to D)\cap C\vdash_{\mathcal{M}} \alpha{:}(C\to D)\cap C}(?)}{x{:}A\,|\,x\cdot\alpha : A\to(C\to D)\cap C\vdash_{\mathcal{M}} \alpha{:}(C\to D)\cap C,\beta{:}D}(\to L)}{\langle x|x\cdot\alpha\rangle : x{:}A\cap(A\to(C\to D)\cap C)\vdash_{\mathcal{M}} \alpha{:}(C\to D)\cap C,\beta{:}D}(cut)$$
$$\dfrac{}{x{:}A\cap(A\to(C\to D)\cap C)\vdash_{\mathcal{M}} \mu\alpha.\langle x|x\cdot\alpha\rangle : (C\to D)\cap C\,|\,\beta{:}D}(\mu)$$

since we are not allowed to use an intersection type for α in that system. Let $\mathcal{D}_2$ be:

$$\dfrac{\dfrac{}{y{:}C\to D\vdash y : C\to D} \quad \dfrac{\dfrac{}{y{:}C\vdash y : C} \quad \dfrac{}{\beta{:}\neg D\vdash \beta : \neg D}}{y{:}C,\beta{:}\neg D\vdash y\cdot\beta : \neg(C\to D)}(\to\mathrm{e})}{y{:}(C\to D)\cap C,\beta{:}\neg D\vdash \langle y|y\cdot\beta\rangle : \bot}(cut)$$
$$\dfrac{}{\beta{:}\neg D\vdash \tilde{\mu}y.\langle y|y\cdot\beta\rangle : \neg((C\to D)\cap C)}(\tilde{\mu})$$

then a derivation for $c_1 = \langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \tilde{\mu}y.\langle y|y\cdot\beta\rangle\rangle$ is given by

$$\dfrac{\boxed{\mathcal{D}_1} \quad \boxed{\mathcal{D}_2}}{x{:}A\cap(A\to(C\to D)\cap C),\beta{:}\neg D\vdash \langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \tilde{\mu}y.\langle y|y\cdot\beta\rangle\rangle : \bot}(cut)$$

Now we have $\langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \tilde{\mu}y.\langle y|y\cdot\beta\rangle\rangle \to_{\tilde{\mu}} \langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta\rangle$, which would yield something like:

$$\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{x{:}A\cap(A\to(C\to D)\cap C)\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : (C\to D)\cap C}(?)}{x{:}A\cap(A\to(C\to D)\cap C)\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : C} \quad \dfrac{}{\beta{:}\neg D\vdash \beta : \neg D}}{\beta{:}\neg D\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta : \neg(C\to D)}(\to\mathrm{e})$$
$$\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{x{:}A\cap(A\to(C\to D)\cap C)\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : (C\to D)\cap C}(?)}{x{:}A\cap(A\to(C\to D)\cap C)\vdash \mu\alpha.\langle x|x\cdot\alpha\rangle : C\to D} \quad \vdots}{\beta{:}\neg D\vdash \langle \mu\alpha.\langle x|x\cdot\alpha\rangle \,|\, \mu\alpha.\langle x|x\cdot\alpha\rangle\cdot\beta\rangle : \bot}(cut)$$

But we cannot make the $(?)$ steps, since we cannot admit rule $(\cap E)$ in $\mathcal{M}^{\cap}$.

So, both soundness and completeness collapse for System $\mathcal{M}^{\cap}$.

9 Fixing completeness: system $\mathcal{M}^{\mathrm{c}}$

As mentioned above in Section 7, neither completeness nor soundness hold for System $\mathcal{M}$. We have argued above, in Examples 6.2 and 6.3, that in order to obtain completeness, it is at least necessary to allow also for union types for term variables and intersection types for context variables. Notice that this is not in contradiction with our findings when analysing the minimal requirements in Section 3: there we found that left-hand environments should be allowed to contain intersection types, and right-hand environments union types, which does not exclude intersections in right-hand environments or union in left-hand environments.

This generalisation is part of the approach of the system presented in this section. With respect to the rules of $\mathcal{M}$, the main difference is that we add the rules $(\top)$ and $(\bot)$ to deal with erasing reductions, and add the rules $()$ and $()$ to deal with the problem noted in Example 6.1; we keep, however, the non-contra-variant relation on types.

Definition 9.1 (System $\mathcal{M}^{\mathrm{c}}$) The system $\mathcal{M}^{\mathrm{c}}$ is defined by:

i) The left and right-hand environment Γ and Δ for derivable statements can contain all types in $\mathcal{T}$.

ii) Typing in $\mathcal{M}^{\mathrm{c}}$ is defined by the following sequent system:

$$(cut): \frac{\Gamma_1 \vdash v : A \mid \Delta_1 \quad \Gamma_2 \mid e : A \vdash \Delta_2}{\langle v|e\rangle : \Gamma_1 \cap \Gamma_2 \vdash \Delta_1 \cup \Delta_2}$$

$$(Ax\text{-}R): \frac{}{\Gamma, x{:}A \vdash x : A \mid \Delta} \qquad (Ax\text{-}L): \frac{}{\Gamma \mid \alpha : A \vdash \alpha{:}A, \Delta}$$

$$(\to R): \frac{\Gamma, x{:}A \vdash v : B \mid \Delta}{\Gamma \vdash \lambda x.v : A \to B \mid \Delta} \qquad (\to L): \frac{\Gamma_1 \vdash v : A \mid \Delta_1 \quad \Gamma_2 \mid e : B \vdash \Delta_2}{\Gamma_1 \cap \Gamma_2 \mid v \cdot e : A \to B \vdash \Delta_1 \cup \Delta_2}$$

$$(\mu): \frac{c : \Gamma \vdash \alpha{:}A, \Delta}{\Gamma \vdash \mu\alpha.c : A \mid \Delta} \qquad (\tilde{\mu}): \frac{c : \Gamma, x{:}A \vdash \Delta}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

$$(\cap R): \frac{\Gamma_i \vdash v : A_i \mid \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \vdash v : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}}\Delta_i} \qquad (\cup L): \frac{\Gamma_i \mid e : A_i \vdash \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \mid e : \cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i}$$

$$(\top): \frac{}{\Gamma \vdash v : \top \mid \Delta} \qquad (\bot): \frac{}{\Gamma \mid e : \bot \vdash \Delta}$$

$$(\leq R): \frac{\Gamma \vdash v : A \mid \Delta}{\Gamma \vdash v : B \mid \Delta}\,(A \leq B) \qquad (\leq L): \frac{\Gamma \mid e : B \vdash \Delta}{\Gamma \mid e : A \vdash \Delta}\,(A \leq B)$$

$$(\cup R\text{-}v): \frac{\Gamma_i, x{:}A_i \vdash v : B_i \mid \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i, x{:}\cup_{\underline{n}} A_i \vdash v : \cup_{\underline{n}} B_i \mid \cup_{\underline{n}}\Delta_i} \qquad (\cap L\text{-}e): \frac{\Gamma_i \mid e : A_i \vdash \alpha{:}B_i, \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \mid e : \cap_{\underline{n}} A_i \vdash \alpha{:}\cap_{\underline{n}} B_i, \cup_{\underline{n}}\Delta_i}$$

$$(\cup\Gamma): \frac{c : \Gamma_i, x{:}A_i \vdash \Delta_i \quad (\forall i \in \underline{n})}{c : \cap_{\underline{n}}\Gamma_i, x{:}\cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i} \qquad (\cap\Delta): \frac{c : \Gamma_i \vdash \alpha{:}B_i, \Delta_i \quad (\forall i \in \underline{n})}{c : \cap_{\underline{n}}\Gamma_i \vdash \alpha{:}\cap_{\underline{n}} B_i, \cup_{\underline{n}}\Delta_i}$$

We will use $\vdash_{\mathrm{c}}$ for this notion.

Notice that the $(\cap E)/(\cup E)$ character of rules $(Ax\text{-}R)$ and $(Ax\text{-}L)$ of system $\mathcal{M}$ is no longer part of the rules above; instead, we just pick a statement from a environment, and use the rules $(\leq R)$ and $(\leq L)$ to perform the necessary selection of sub-types; notice also that these rules supersede the rules $(\cup R)$ and $(\cap L)$ of $\mathcal{M}$. And in fact, we can show:

Theorem 9.2 *If* $c : \Gamma \vdash_{\mathcal{M}} \Delta$, *then* $c : \Gamma \vdash_{\mathrm{c}} \Delta$, *and similarly for* $\Gamma \vdash_{\mathcal{M}} v : A \mid \Delta$ *and* $\Gamma \mid e : A \vdash_{\mathcal{M}} \Delta$.

Proof: By easy induction. □

Notice also that the rules $(\top)$ and $(\bot)$ are special cases of $(\cap R)$ and $(\cup L)$ by allowing $n = 0$ as well and conveniently ignoring the fact that an intersection (or union) of an empty

collection of sets is the empty set. So we will rather be using the rules

$$(\cap R):\ \frac{\Gamma_i \vdash v : A_i \mid \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \vdash v : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}}\Delta_i}(n \geq 0, n \neq 1) \qquad (\cup L):\ \frac{\Gamma_i \mid e : A_i \vdash \Delta_i \quad (\forall i \in \underline{n})}{\cap_{\underline{n}}\Gamma_i \mid e : \cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i}(n \geq 0, n \neq 1)$$

Remark 9.3 To be more consistent with the system of Definition 4.5, we could have used the rules:

$$(Ax\text{-}R):\ \frac{}{\Gamma, x{:}A \vdash x : B \mid \Delta}(A \leq_\cap B, B \in \mathcal{T}_p) \qquad (Ax\text{-}L):\ \frac{}{\Gamma \mid \alpha : A \vdash \alpha{:}B, \Delta}(A \leq_\cup B, A \in \mathcal{T}_p)$$

$$(\cup R):\ \frac{\Gamma \vdash v : A_i \mid \Delta \quad (\forall i \in \underline{n})}{\Gamma \vdash v : \cup_{\underline{n}} A_i \mid \Delta}(n \geq 2) \qquad (\cap L):\ \frac{\Gamma \mid e : A_i \vdash \Delta \quad (\forall i \in \underline{n})}{\Gamma \mid e : \cap_{\underline{n}} A_i \vdash \Delta}(n \geq 2)$$

instead of rules $(Ax\text{-}R)$, $(Ax\text{-}L)$, $(\leq R)$, and $(\leq L)$ of $\mathcal{M}$. It is easy to check that these alternative rules are derivable in the system of Definition 9.1; for the converse, we can show, by induction on the structure of derivations, that then rules $(\leq R)$ and $(\leq L)$ are admissible.

We can characterise typeability in $\vdash_c$ as usual:

Lemma 9.4 (Generation Lemma) *i) If* $\langle v|e\rangle : \Gamma \vdash_c \Delta$, *then there exists* A *such that* $\Gamma \vdash_c v : A \mid \Delta$ *and* $\Gamma \mid e : A \vdash_c \Delta$.

ii) a) If $\Gamma \vdash_c x : A \mid \Delta$, *then there exists* $x{:}B \in \Gamma$ *such that* $B \leq A$.

b) If $\Gamma \vdash_c \lambda x.v : A \mid \Delta$, *then there exists* $n \geq 1$, $B_i, C_i\,(\forall i \in \underline{n})$ *such that* $\Gamma, x{:}B_i \vdash_c v : C_i \mid \Delta$, *and* $\cap_{\underline{n}}(B_i \to C_i) \leq A$.

c) If $\Gamma \vdash_c \mu\alpha.c : A \mid \Delta$, *then there exists* B *such that* $c : \Gamma \vdash_c \alpha{:}B, \Delta$, *and* $B \leq A$.

iii) a) If $\Gamma \mid \alpha : A \vdash_c \Delta$, *then there exists* $\alpha{:}B \in \Delta$ *such that* $A \leq B$.

b) If $\Gamma \vdash_c v{\cdot}e : A \mid \Delta$, *then there exists* $n \geq 1$, $B_i, C_i\,(\forall i \in \underline{n})$ *such that* $\Gamma \vdash_c v : B_i \mid \Delta$ *and* $\Gamma \mid e : C_i \vdash_c \Delta$, *and* $A \leq \cup_{\underline{n}}(B_i \to C_i)$.

c) If $\Gamma \mid \tilde{\mu}x.c : A \vdash_c \Delta$, *then there exists* B *such that* $c : \Gamma, x{:}B \vdash_c \Delta$, *and* $A \leq B$.

Proof: Straightforward. □

We can now show that typing is preserved by the interpretation of the λ-calculus in $\overline{\lambda}\mu\tilde{\mu}$:

Theorem 9.5 *If* $\Gamma \vdash_\cap M : A$ *then* $\Gamma \vdash_c [\![M]\!]^{\mathrm{n}}_{\mu} : A \mid$.

Proof: By easy induction on the structure of derivations in $\vdash_\cap$.

(Ax): Then $M = x$, and $x{:}A \in \Gamma$; by rule $(Ax\text{-}R)$, $\Gamma \vdash_c x : A \mid$.

$(\to I)$: Then $M = \lambda x.N$, $A = B \to C$, and $\Gamma, x{:}B \vdash_\cap N : C$. By induction we can assume that $\Gamma, x{:}B \vdash_c [\![N]\!]^{\mathrm{n}}_{\mu} : C \mid$; then, by rule $(\to R)$, also $\Gamma \vdash_c [\![\lambda x.N]\!]^{\mathrm{n}}_{\mu} : A \mid$.

$(\to E)$: Then $M = PQ$, and there is a B such that $\Gamma \vdash_\cap P : B \to A$ and $\Gamma \vdash_\cap Q : B$. By induction, we have both $\Gamma \vdash_c [\![P]\!]^{\mathrm{n}}_{\mu} : B \to A \mid$ and $\Gamma \vdash_c [\![Q]\!]^{\mathrm{n}}_{\mu} : B \mid$, and we can construct:

$$\dfrac{\dfrac{\boxed{\Gamma \vdash [\![P]\!]^{\mathrm{n}}_{\mu} : B \to A \mid} \quad \dfrac{\boxed{\Gamma \vdash [\![Q]\!]^{\mathrm{n}}_{\mu} : B \mid} \quad \dfrac{}{\mid \alpha : A \vdash \alpha{:}A}}{\Gamma \mid [\![Q]\!]^{\mathrm{n}}_{\mu}{\cdot}\alpha : B \to A \vdash \alpha{:}A}(\to L)}{\langle [\![P]\!]^{\mathrm{n}}_{\mu} \mid [\![Q]\!]^{\mathrm{n}}_{\mu}{\cdot}\alpha\rangle : \Gamma \vdash \alpha{:}A}(cut)}{\Gamma \vdash \mu\alpha.\langle [\![P]\!]^{\mathrm{n}}_{\mu} \mid [\![Q]\!]^{\mathrm{n}}_{\mu}{\cdot}\alpha\rangle : A \mid}(\mu)$$

$(\cap I)$: Then $A = \cap_{\underline{n}} A_i$, and, for every $i \in \underline{n}$, $\Gamma \vdash_\cap M : A_i$. Then, by induction we can assume that $\Gamma \vdash_c [\![M]\!]^{\mathrm{n}}_{\mu} : A_i \mid$ for all $i \in \underline{n}$, and by $(\cap R)$, $\Gamma \vdash_c [\![M]\!]^{\mathrm{n}}_{\mu} : A \mid$.

$(\cap E)$: Then $\Gamma \vdash_\cap M : A \cap B$, for some B. Then, by induction, $\Gamma \vdash_c [\![M]\!]^{\mathrm{n}}_{\mu} : A \cap B \mid$, and also $\Gamma \vdash_c [\![M]\!]^{\mathrm{n}}_{\mu} : A \mid$ by rule $(\leq R)$. □

Remember that this proof would not be valid in $\mathcal{M}$, since, in case $(\to E)$, A might be an intersection type.

We will now show that $\mathcal{M}^{\mathrm{c}}$ is complete. First, we show some of the properties that this system satisfies. As is usual, we can constrict the environment to just those statements that are relevant, and add irrelevant statements to environments.

Lemma 9.6 (Thinning and weakening) *i) If* $c : \Gamma \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma' = \{x{:}B \in \Gamma \mid x \in fv(c)\}$ *and* $\Delta' = \{\alpha{:}B \in \Gamma \mid \alpha \in fv(c)\}$, *then also* $c : \Gamma' \vdash_{\mathrm{c}} \Delta'$. *Similar for* $\Gamma \mid e : A \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma \vdash_{\mathrm{c}} v : A \mid \Delta$.

ii) If $c : \Gamma \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma' \leq_{\cap} \Gamma$ *as well as* $\Delta \leq_{\cup} \Delta'$, *then also* $c : \Gamma' \vdash_{\mathrm{c}} \Delta'$. *Also similar for* $\Gamma \mid e : A \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma \vdash_{\mathrm{c}} v : A \mid \Delta$.

Proof: Both properties are shown by induction on the structure of derivations. □

We start by showing the following expansion lemmas that express that, given a typeable term, context or command, we can extract a subterm (which might occur more than once, *i.e.* the extraction is expressed via substitution) and type both the result of the extraction and the extracted term/context.

First for the extraction of a term:

Lemma 9.7 (Term expansion lemma) *i) If* $c\{v/x\} : \Gamma \vdash_{\mathrm{c}} \Delta$ *there exists* $B \in \mathcal{T}$ *such that* $c : \Gamma, x{:}B \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma \vdash_{\mathrm{c}} v : B \mid \Delta$.

ii) If $\Gamma \vdash_{\mathrm{c}} v'\{v/x\} : A \mid \Delta$ *then there exists* $B \in \mathcal{T}$ *such that* $\Gamma, x{:}B \vdash_{\mathrm{c}} v' : A \mid \Delta$ *and* $\Gamma \vdash_{\mathrm{c}} v : B \mid \Delta$.

iii) If $\Gamma \mid e\{v/x\} : A \vdash_{\mathrm{c}} \Delta$ *then there exists* $B \in \mathcal{T}$ *such that* $\Gamma, x{:}B \mid e : A \vdash_{\mathrm{c}} \Delta$ *and* $\Gamma \vdash_{\mathrm{c}} v : B \mid \Delta$.

Proof: By simultaneous induction; we only show some of the cases.

i) If $c\{v/x\} : \Gamma \vdash_{\mathrm{c}} \Delta$ then $c = \langle v' | e' \rangle$, and we have $\Gamma \vdash_{\mathrm{c}} v'\{v/x\} : C \mid \Delta$ and $\Gamma \mid e'\{v/x\} : C \vdash_{\mathrm{c}} \Delta$ for some $C \in \mathcal{T}$ (by weakening we can assume the environments to be the same). Then, by induction *(ii)* and *(iii)*, there exist B_1 such that $\Gamma, x{:}B_1 \vdash_{\mathrm{c}} v' : C \mid \Delta$ and $\Gamma \vdash_{\mathrm{c}} v : B_1 \mid \Delta$, and B_2 such that $\Gamma, x{:}B_2 \mid e' : C \vdash_{\mathrm{c}} \Delta$ and $\Gamma \vdash_{\mathrm{c}} v : B_2 \mid \Delta$. Then, by Lemma 9.6 and rule (cut), we have $\langle v' | e' \rangle : \Gamma, x{:}B_1 \cap B_2 \vdash_{\mathrm{c}} \Delta$, and, by rule $(\cap R)$, $\Gamma \vdash_{\mathrm{c}} v : B_1 \cap B_2 \mid \Delta$.

ii) We have four cases, of which we show two; the other two follow by induction:

$(v' = x)$: Then $\Gamma \vdash_{\mathrm{c}} v : A \mid \Delta$. Take $B = A$; by rule $(Ax\text{-}R)$, $\Gamma, x{:}A \vdash_{\mathrm{c}} x : A \mid \Delta$. (Notice that A can be a union type.)

$(v' = y \neq x)$: Then $\Gamma \vdash_{\mathrm{c}} y : A \mid \Delta$. Take $B = \top$; then by rule $(\top)$, we have $\Gamma \vdash_{\mathrm{c}} v : \top \mid \Delta$, and by weakening, also $\Gamma, x{:}\top \vdash_{\mathrm{c}} y : A \mid \Delta$.

iii) We have three cases, of which we show two:

$(e = \alpha)$: Then $\Gamma \mid \alpha : A \vdash_{\mathrm{c}} \Delta$. Take $B = \top$, then $\Gamma, x{:}\top \mid \alpha : A \vdash_{\mathrm{c}} \Delta$ by weakening, and, by rule $(\top)$, $\Gamma \vdash_{\mathrm{c}} v : \top \mid \Delta$.

$(e = v' \cdot e')$: Without loss of generality, assume that the derivation ends with rule $(\to L)$. Then there exist C, D such that $A = C \to D$, $\Gamma \vdash_{\mathrm{c}} v'\{v/x\} : C \mid \Delta$, and $\Gamma \mid e'\{v/x\} : D \vdash_{\mathrm{c}} \Delta$. Then, by induction *(ii)*, there exists a type B_1 such that both $\Gamma, x{:}B_1 \vdash_{\mathrm{c}} v' : C \mid \Delta$ and $\Gamma \vdash_{\mathrm{c}} v' : B_1 \mid \Delta$ and there exists B_2 such that $\Gamma, x{:}B_2 \mid e' : D \vdash_{\mathrm{c}} \Delta$ and $\Gamma \vdash_{\mathrm{c}} v : B_2 \mid \Delta$ by induction *(iii)*. Then, by Lemma 9.6 and rule $(\to L)$, we have $v' \cdot e' : \Gamma, x{:}B_1 \cap B_2 \vdash_{\mathrm{c}} \Delta$, and, by rule $(\cap R)$, $\Gamma \vdash_{\mathrm{c}} v : B_1 \cap B_2 \mid \Delta$. □

Notice that, in this proof, we only ever build intersection types for x; however, since, by rule (μ), we can build a union type for a term, we do not always have that $B \in \mathcal{T}_{\cap}$, and can only show our property for $B \in \mathcal{T}$.

We can show a similar result for the extraction of a context:

Lemma 9.8 (Context expansion lemma) *i) If* $c\{e/\alpha\} : \Gamma \vdash_c \Delta$ *then there exists* $B \in \mathcal{T}$ *such that* $c : \Gamma \vdash_c \alpha{:}B, \Delta$ *and* $\Gamma \mid e : B \vdash_c \Delta$.

ii) If $\Gamma \vdash_c v'\{e/\alpha\} : A \mid \Delta$ *then there exists* $B \in \mathcal{T}$ *such that* $\Gamma \vdash_c v' : A \mid \alpha{:}B, \Delta$ *and* $\Gamma \mid e : B \vdash_c \Delta$.

iii) If $\Gamma \mid e'\{e/\alpha\} : A \vdash_c \Delta$ *then there exists* $B \in \mathcal{T}$ *such that* $\Gamma \mid e' : A \vdash_c \alpha{:}B, \Delta$ *and* $\Gamma \mid e : B \vdash_c \Delta$.

Proof: By simultaneous induction, much like Lemma 9.7, using union and $\bot$ rather than intersection and $\top$. □

Notice that intersections and $\top$ are crucial for Lemma 9.7, as are union and $\bot$ for Lemma 9.8. Using these two lemmas, it is easy to show the following completeness result.

Theorem 9.9 (Subject expansion) *Let* $c_1 \to c_2$*: if* $c_2 : \Gamma \vdash_c \Delta$ *then* $c_1 : \Gamma \vdash_c \Delta$.

Proof: By induction on the definition of reduction, where we focus on the logical rules:

$(\langle \lambda x.v_1 \mid v_2 \cdot e\rangle \to \langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e\rangle\rangle)$: assume $\langle v_2 \mid \tilde{\mu}x.\langle v_1 \mid e\rangle\rangle : \Gamma \vdash_c \Delta$, then by Lemma 9.4,

- there exists A such that $\mathcal{D}_1 :: \Gamma \vdash_c v_2 : A \mid \Delta$ and $\Gamma \mid \tilde{\mu}x.\langle v_1 \mid e\rangle : A \vdash_c \Delta$, so
- there exists B such that $\mathcal{D}_2 :: \langle v_1 \mid e\rangle : \Gamma, x{:}B \vdash_c \Delta$, and $A \leq B$, so
- there exists C such that $\mathcal{D}_2^1 :: \Gamma, x{:}B \vdash_c v_1 : C \mid \Delta$ and $\mathcal{D}_2^2 :: \Gamma \mid e : C \vdash_c \Delta$.

(note that we can assume that x does not occur in e). Then we can construct

$$\dfrac{\dfrac{\boxed{\mathcal{D}_2^1}}{\Gamma, x{:}B \vdash v_1 : C \mid \Delta}(\to R)}{\Gamma \vdash \lambda x.v_1 : B \to C \mid \Delta} \quad \dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\Gamma \vdash v_2 : A \mid \Delta}(\leq R)}{\Gamma \vdash v_2 : B \mid \Delta} \quad \dfrac{\boxed{\mathcal{D}_2^2}}{\Gamma \mid e : C \vdash \Delta}(\to L)}{\langle \lambda x.v_1 \mid v_2 \cdot e\rangle : \Gamma \vdash \Delta}(cut)$$

$(\langle v \mid \tilde{\mu}x.c\rangle \to c\{v/x\})$: By Lemma 9.7, there exists an A such that both $c : \Gamma, x{:}A \vdash_c \Delta$, and $\Gamma \vdash_c v : A \mid \Delta$. Then we can derive:

$$\dfrac{\boxed{} \quad \dfrac{\boxed{\phantom{c : \Gamma, x{:}A \vdash \Delta}}}{c : \Gamma, x{:}A \vdash \Delta}(\tilde{\mu})}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}}{\langle v \mid \tilde{\mu}x.c\rangle : \Gamma \vdash \Delta}(cut)$$

$(\langle \mu\beta.c \mid e\rangle \to c\{e/\beta\})$: By Lemma 9.8, there exists an A such that both $c : \Gamma \vdash_c \beta{:}A, \Delta$, and $\Gamma \mid e : A \vdash_c \Delta$. Then we can derive:

$$\dfrac{\dfrac{\boxed{\phantom{c : \Gamma \vdash \beta{:}A, \Delta}}}{c : \Gamma \vdash \beta{:}A, \Delta}(\mu)}{\Gamma \vdash \mu\beta.c : A \mid \Delta} \quad \dfrac{\boxed{}}{\Gamma \mid e : A \vdash \Delta}}{\langle \mu\beta.c \mid e\rangle : \Gamma \vdash \Delta}(cut)$$

Notice that this proof only works in $\mathcal{M}^c$. □

Allowing all types in left and right-hand environments increases the typeability of terms:

Example 9.10 Let c_1, c_2 and c_3 be as in Example 7.1, where we have shown

- $\langle \mu\alpha.\langle x \mid x \cdot \alpha\rangle \mid \tilde{\mu}y.\langle y \mid y \cdot \beta\rangle\rangle : x{:}A \cap (A \to C) \cap (A \to C \to D) \vdash_c \beta{:}D$,
- $\langle x \mid x \cdot (\mu\alpha.\langle x \mid x \cdot \alpha\rangle \cdot \beta)\rangle : x{:}A \cap (A \to C) \cap (A \to C \to D) \vdash_c \beta{:}D$, and
- $\langle x \mid x \cdot \tilde{\mu}y.\langle y \mid y \cdot \beta\rangle\rangle : x{:}A \cap (A \to (C \to D) \cap C) \vdash_c \beta{:}D$.

and $c_1 \to_{\mathrm{N}} c_2$ and $c_1 \to_{\mathrm{v}} c_3$. As remarked after that example, we cannot derive c_3's type for c_1 in $\mathcal{M}$; we can, however, derive that in $\mathcal{M}^{\mathrm{c}}$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}A \vdash x : A \mid}\quad\dfrac{}{\mid \alpha : (C{\to}D){\cap}C \vdash \alpha{:}(C{\to}D){\cap}C}}{x{:}A \mid x{\cdot}\alpha : A{\to}(C{\to}D){\cap}C \vdash \alpha{:}(C{\to}D){\cap}C, \beta{:}D}(\to L)}}{\vdots}}{\dfrac{\dfrac{\dfrac{x{:}A{\to}(C{\to}D){\cap}C \vdash x : A{\to}(C{\to}D){\cap}C \mid}{\langle x|x{\cdot}\alpha\rangle : x{:}A \cap (A{\to}(C{\to}D){\cap}C) \vdash \alpha{:}(C{\to}D){\cap}C, \beta{:}D}(cut)}{x{:}A \cap (A{\to}(C{\to}D){\cap}C) \vdash \mu\alpha.\langle x|x{\cdot}\alpha\rangle : (C{\to}D){\cap}C \mid \beta{:}D}(\mu)}{\langle \mu\alpha.\langle x|x{\cdot}\alpha\rangle \mid \tilde{\mu}y.\langle y|y{\cdot}\beta\rangle\rangle : x{:}A \cap (A{\to}(C{\to}D){\cap}C) \vdash \beta{:}D}(cut)$$

so there is no soundness nor completeness problem for this typing when reducing c_1 to c_3. In the other direction, reducing c_1 to c_2, the soundness problem for this typing does not exist, since we can derive c_3's type for c_2:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}A{\to}C \cap (C{\to}D) \vdash x : A{\to}C \cap (C{\to}D) \mid}}{\vdots}\quad\dfrac{\dfrac{}{x{:}A{\to}C \cap (C{\to}D) \vdash x : A{\to}C \cap (C{\to}D) \mid}\quad\dfrac{\dfrac{\dfrac{}{\mid \alpha : C \vdash \alpha{:}C}}{\mid \alpha : C \cap (C{\to}D) \vdash \alpha{:}C}(\leq L)}{x{:}A \mid x{\cdot}\alpha : A{\to}C \cap (C{\to}D) \vdash \alpha{:}C}(\to L)}{\langle x|x{\cdot}\alpha\rangle : x{:}A \cap (A{\to}C \cap (C{\to}D)) \vdash \alpha{:}C}(cut)}{x{:}A \cap (A{\to}C \cap (C{\to}D)) \vdash \mu\alpha.\langle x|x{\cdot}\alpha\rangle : C \mid}(\mu)\quad\dfrac{}{\mid \beta : D \vdash \beta{:}D}}{x{:}A \cap (A{\to}C \cap (C{\to}D)) \mid \mu\alpha.\langle x|x{\cdot}\alpha\rangle{\cdot}\beta : C{\to}D \vdash \beta{:}D}(\to L)}{\dfrac{\dfrac{}{x{:}A \vdash x : A \mid}\quad x{:}A \cap (A{\to}C \cap (C{\to}D)) \mid \mu\alpha.\langle x|x{\cdot}\alpha\rangle{\cdot}\beta : C \cap (C{\to}D) \vdash \beta{:}D}{x{:}A \cap (A{\to}C \cap (C{\to}D)) \mid x{\cdot}(\mu\alpha.\langle x|x{\cdot}\alpha\rangle{\cdot}\beta) : A{\to}C \cap (C{\to}D) \vdash \beta{:}D}(\to L)}(\leq L)}{\langle x \mid x{\cdot}(\mu\alpha.\langle x|x{\cdot}\alpha\rangle{\cdot}\beta)\rangle : x{:}A \cap (A{\to}C \cap (C{\to}D)) \vdash \beta{:}D}(cut)$$

Notice that now the two uses of rule $(\leq L)$ actually correspond to $(\cap L)$, which (finally) is of use.

Although using $\mathcal{M}^{\mathrm{c}}$ rather than $\mathcal{M}$ solves one subject reduction problem, it does not solve that of Example 7.2. Moreover, we still cannot show soundness result for the reduction rule $\to_\mu$, as we will illustrate in the next example.

Example 9.11 Take

$$\begin{array}{rcl} c_1 & = & \langle \mu\alpha.\langle \lambda x\mu\beta.\langle v|\alpha\rangle \mid e\rangle \mid \tilde{\mu}y.\langle y|y{\cdot}\delta\rangle\rangle \text{ and} \\ c_2 & = & \langle \lambda x\mu\beta.\langle v \mid \tilde{\mu}y.\langle y|y{\cdot}\delta\rangle\rangle \mid e\rangle, \end{array}$$

where α does not occur in v or e; notice that $c_1 \to_\mu c_2$. Assuming we have the derivations for, respectively,

$$\begin{array}{l} \mathcal{D}_1 :: x{:}C \vdash_{\mathrm{c}} v : A{\to}B \mid \beta{:}E \\ \mathcal{D}_2 :: \Gamma \mid e : C{\to}E \vdash_{\mathrm{c}} \Delta \\ \mathcal{D}_3 :: x{:}D \vdash_{\mathrm{c}} v : A \mid \beta{:}F \\ \mathcal{D}_4 :: \Gamma \mid e : D{\to}F \vdash_{\mathrm{c}} \Delta \end{array}$$

then we can type c_1 as follows. First, take $\mathcal{D}_5 :: \mid \tilde{\mu}y.\langle y|y{\cdot}\delta\rangle : (A{\to}B){\cap}A \vdash_{\overline{\lambda}} \delta{:}B$ from Exam-

ple 4.7. Then take

$$\mathcal{D}_6 :: \dfrac{\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\Gamma, x{:}C \vdash v : A{\to}B \mid \beta{:}E, \Delta} \quad \dfrac{}{\mid \alpha : A{\to}B \vdash \alpha{:}A{\to}B}}{\langle v \mid \alpha \rangle : \Gamma, x{:}C \vdash \beta{:}E, \alpha{:}A{\to}B, \Delta}\,(cut)}{\Gamma, x{:}C \vdash \mu\beta.\langle v \mid \alpha \rangle : E \mid \alpha{:}A{\to}B, \Delta}\,(\mu)}{\Gamma \vdash \lambda x\mu\beta.\langle v \mid \alpha \rangle : C{\to}E \mid \alpha{:}A{\to}B, \Delta}\,(\to R) \quad \boxed{\mathcal{D}_2}}{\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : \Gamma \vdash \alpha{:}A{\to}B, \Delta}\,(cut)}{\Gamma \vdash \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : A{\to}B \mid \Delta}\,(\mu)$$

and

$$\mathcal{D}_7 :: \dfrac{\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_3}}{\Gamma, x{:}D \vdash v : A \mid \beta{:}F, \Delta} \quad \dfrac{}{\mid \alpha : A \vdash \alpha{:}A}}{\langle v \mid \alpha \rangle : \Gamma, x{:}D \vdash \beta{:}F, \alpha{:}A, \Delta}\,(cut)}{\Gamma, x{:}D \vdash \mu\beta.\langle v \mid \alpha \rangle : F \mid \alpha{:}A, \Delta}\,(\mu)}{\Gamma \vdash \lambda x\mu\beta.\langle v \mid \alpha \rangle : D{\to}F \mid \alpha{:}A, \Delta}\,(\to R) \quad \boxed{\mathcal{D}_4}}{\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : \Gamma \vdash \alpha{:}A, \Delta}\,(cut)}{\Gamma \vdash \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : A \mid \Delta}\,(\mu)$$

then we can construct:

$$\dfrac{\dfrac{\boxed{\mathcal{D}_6}}{\Gamma \vdash \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : A{\to}B \mid \Delta} \quad \dfrac{\boxed{\mathcal{D}_7}}{\Gamma \vdash \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : A \mid \Delta}\,(\cap R)}{\Gamma \vdash \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle : (A{\to}B){\cap}A \mid \Delta} \quad \begin{array}{c} \\ \vdots \end{array} \quad \dfrac{\boxed{\mathcal{D}_5}}{\mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle : (A{\to}B){\cap}A \vdash \delta{:}B}\,(cut)}{\langle \mu\alpha.\langle \lambda x\mu\beta.\langle v \mid \alpha \rangle \mid e \rangle \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle \rangle : \Gamma \vdash \delta{:}B, \Delta}$$

Now we expect the derivation for $c_2 : \Gamma \vdash_{\overline{\lambda}} \delta{:}B, \Delta$ to be composed out of $\mathcal{D}_1$ to $\mathcal{D}_5$; however, since the intersection type in derivation $\mathcal{D}_5$ cannot be split, we can at most derive:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\boxed{\mathcal{D}_1}}{\Gamma, x{:}C \vdash v : A{\to}B \mid \beta{:}E, \Delta} \quad \dfrac{\boxed{\mathcal{D}_2}}{\Gamma, x{:}D \vdash v : A \mid \beta{:}F, \Delta}\,(\cap R)}{\Gamma, x{:}C{\cap}D \vdash v : (A{\to}B){\cap}A \mid \beta{:}E{\cup}F, \Delta} \quad \begin{array}{c} \\ \vdots \end{array} \quad \dfrac{\boxed{\mathcal{D}_5}}{\mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle : (A{\to}B){\cap}A \vdash \delta{:}B}\,(cut)}{\langle v \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle \rangle : \Gamma, x{:}C{\cap}D \vdash \beta{:}E{\cup}F, \Delta}\,(\mu)}{\Gamma, x{:}C{\cap}D \vdash \mu\beta.\langle v \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle \rangle : E{\cup}F \mid \delta{:}B, \Delta}\,(\to R) \quad \dfrac{\boxed{\mathcal{D}_3} \quad \boxed{\mathcal{D}_4}}{\Gamma \mid e : C{\to}E \vdash \Delta \quad \Gamma \mid e : D{\to}F \vdash \Delta}\,(\cup L)}{\Gamma \vdash \lambda x\mu\beta.\langle v \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle \rangle : e{:}C{\cap}D{\to}E{\cup}F \mid \delta{:}B, \Delta}\,(?)}{\langle \lambda x\mu\beta.\langle v \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta \rangle \rangle \mid e \rangle : ? \vdash ?}$$

We could only solve this problem via $\leq$ if $C{\cap}D{\to}E{\cup}F$ is smaller than $(C{\to}E){\cup}(D{\to}F)$; this is not the case.

10 Sound restrictions of System $\mathcal{M}$

We have seen above that, in order to achieve completeness (and to show Theorem 9.5), we need to allow all types in both kinds of environments. However, apart from solving a problem for completeness, this also creates one for soundness, even for just mere renaming.

To better understand the problem, assume we have the derivation

$$\dfrac{\dfrac{\boxed{\phantom{c : \Gamma \vdash \alpha{:}A_i, \Delta}} \quad (\forall i \in \underline{n})}{\Gamma \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \Delta}\,(\cap R) \qquad \dfrac{}{\mid \beta : \cap_{\underline{n}} A_i \vdash \beta{:}\cap_{\underline{n}} A_i}}{\langle \mu\alpha.c \mid \beta \rangle : \Gamma \vdash \beta{:}\cap_{\underline{n}} A_i, \Delta}\,(cut)$$

Now $\langle \mu\alpha.c \mid \beta \rangle \to_\mu c\{\beta/\alpha\}$, and we can easily derive $c\{\beta/\alpha\} : \Gamma \vdash_{\mathrm{c}} \beta{:}A_i, \Delta$ for all $i \in \underline{n}$, but we cannot, in general, derive $c\{\beta/\alpha\} : \Gamma \vdash_{\mathrm{c}} \beta{:}\cap_{\underline{n}} A_i, \Delta$, since we lack a rule that builds an intersection type for a context variable. To deal with this issue, we have to add the rules $()$, $()$, $(\cap L)$ and $(\cup R)$ (which, in turn, we can motivate with the reduction $\langle x \mid \tilde{\mu} y.c \rangle \to_\mu c\{x/y\}$).

$$(\cap\Delta): \ \dfrac{c : \Gamma_i \vdash \alpha{:}A_i, \Delta_i \,(\forall i \in \underline{n})}{c : \cap_{\underline{n}} \Gamma_i \vdash \alpha{:}\cap_{\underline{n}} A_i, \cup_{\underline{n}} \Delta_i} \qquad (\cup L\text{-}c): \ \dfrac{c : \Gamma_i, x{:}A_i \vdash \Delta_i \,(\forall i \in \underline{n})}{c : \cap_{\underline{n}} \Gamma_i, x{:}\cup_{\underline{n}} A_i \vdash \cup_{\underline{n}} \Delta_i}$$

The presence of these rules, however, creates ambiguity in the system, in that there is more than one way to derive a result, since we can derive both

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\mathcal{D}_i}}}{c : \Gamma_i \vdash \alpha{:}A_i, \Delta_i}\,(\mu)}{\Gamma_i \vdash \mu\alpha.c : A_i \mid \Delta_i} \quad (\forall i \in \underline{n})}{\cap_{\underline{n}} \Gamma_i \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}} \Delta_i}\,(\cap R) \qquad \dfrac{\boxed{\phantom{\mathcal{D}}}}{\Gamma \mid e : \cap_{\underline{n}} A_i \vdash \Delta}}{\langle \mu\alpha.c \mid e \rangle : \cap_{\underline{n}} \Gamma_i \cap \Gamma \vdash \cup_{\underline{n}} \Delta_i \cup \Delta}\,(cut)$$

and

$$\dfrac{\dfrac{\dfrac{\dfrac{\boxed{\phantom{\mathcal{D}_i}}}{c : \Gamma_i \vdash \alpha{:}A_i, \Delta_i} \quad (\forall i \in \underline{n})}{c : \cap_{\underline{n}} \Gamma_i \vdash \alpha{:}\cap_{\underline{n}} A_i, \cup_{\underline{n}} \Delta_i}\,(\cap\Delta)}{\cap_{\underline{n}} \Gamma_i \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}} \Delta_i}\,(\mu) \qquad \dfrac{\boxed{\phantom{\mathcal{D}}}}{\Gamma \mid e : \cap_{\underline{n}} A_i \vdash \Delta}}{\langle \mu\alpha.c \mid e \rangle : \cap_{\underline{n}} \Gamma_i \cap \Gamma \vdash \cup_{\underline{n}} \Delta_i \cup \Delta}\,(cut)$$

This makes reasoning about derivations much more complex; there might be ways to avoid this, by allowing rules to be applied only in a certain order, but we would still have a cumbersome generation lemma for $\mathcal{M}^{\mathrm{c}}$. Moreover, not even for $\mathcal{M}^{\mathrm{c}}$ can we show a general subject *reduction* result: the counterexamples we gave for $\mathcal{M}$ are also valid for $\mathcal{M}^{\mathrm{c}}$, and we will show that we need to restrict any system before that property becomes provable. Also, the elegant solution we present below (restricting $(\cap R)$ to *values* or $(\cup L)$ to *slots*) would need to be extended (modified) in $\mathcal{M}^{\mathrm{c}}$, since there more rules introduce intersections or unions.

So, rather, we revert to System $\mathcal{M}$; however, the solution we will present below will, in all likelihood, be central also to a solution for $\mathcal{M}^{\mathrm{c}}$.

Looking at the essence of the counterexamples 7.1 and 7.2, it is evident that it is problematic to: (1) perform a μ-reduction towards an intersection type, introduced via $(\cap R)$, or (2) perform a $\tilde{\mu}$-reduction towards a union type, introduced via $(\cup L)$. Let's first of all analyse the situation. If an intersection has been used to type a μ-abstraction, we find ourselves in the following situation:

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\mathcal{D}_i}}}{c : \Gamma_i \vdash \alpha{:}A_i, \Delta_i}\,(\mu)}{\Gamma_i \vdash \mu\alpha.c : A_i \mid \Delta_i} \quad (\forall i \in \underline{n})}{\cap_{\underline{n}} \Gamma_i \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \cup_{\underline{n}} \Delta_i}\,(\cap R) \qquad \dfrac{\boxed{\phantom{\mathcal{D}}}}{\Gamma \mid e : \cap_{\underline{n}} A_i \vdash \Delta}}{\langle \mu\alpha.c \mid e \rangle : \cap_{\underline{n}} \Gamma_i \cap \Gamma \vdash \cup_{\underline{n}} \Delta_i \cup \Delta}\,(cut)$$

We normally cannot remove any of the types in $\cap_{\underline{n}} A_i$ in the derivation $\mathcal{D}$, so we cannot safely propagate $\mathcal{D}$ into the various $\mathcal{D}_i$. So this is the problem case that should be caught.

Similarly, for union, we have the situation

$$\dfrac{\boxed{\mathcal{D}}\ \Gamma \vdash v : \cup_{\underline{n}} A_i \mid \Delta \qquad \dfrac{\dfrac{\boxed{\mathcal{D}_i}\ c : \Gamma_i, x{:}A_i \vdash \Delta_i}{\Gamma_i \mid \tilde{\mu}x.c : A_i \vdash \Delta_i}(\tilde{\mu})}{\cap_{\underline{n}}\Gamma_i \mid \tilde{\mu}x.c : \cup_{\underline{n}} A_i \vdash \cup_{\underline{n}}\Delta_i}\ (\forall i \in \underline{n})\,(\cup L)}{\langle v \mid \tilde{\mu}x.c\rangle : \cap_{\underline{n}}\Gamma_i \cap \Gamma \vdash \cup_{\underline{n}}\Delta_i \cup \Delta}\,(cut)$$

As above, propagating $\mathcal{D}$ into the $\mathcal{D}_i$ should be avoided.

This problem also appears when dealing with quantification, as in ML:

Example 10.1 Assume a system with quantification[3], so containing the additional rules

$$(\forall R):\ \frac{\Gamma \vdash v : A \mid \Delta}{\Gamma \vdash v : \forall\phi.A \mid \Delta}\,(\phi \textit{ not free in } \Gamma, \Delta) \qquad (\forall v):\ \frac{}{\Gamma, x{:}\forall\phi.A \vdash x : A\{B/\phi\} \mid \Delta}$$

$$(\exists v):\ \frac{}{\Gamma \mid \alpha : A\{B/\phi\} \vdash \alpha{:}\exists\phi.A, \Delta} \qquad (\exists L):\ \frac{\Gamma \mid e : A \vdash \Delta}{\Gamma \mid e : \exists\phi.A \vdash_{\overline{\lambda}} \Delta}\,(\phi \textit{ not free in } \Gamma, \Delta)$$

(as with intersection and union, the connectors $\forall$ and $\exists$ are not treated as logical, so the rules $(\forall L)$ and $(\exists R)$ need not be added) in which we have a derivation for

$$\langle \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta\rangle\rangle : y{:}\forall\phi.\phi{\to}\phi \vdash_{\overline{\lambda}} \delta{:}\phi{\to}\phi$$

constructed as follows. Take $\mathcal{D}_1$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}\phi, y{:}\phi \vdash x : \phi \mid}}{x{:}\phi \vdash \lambda y.x : \phi{\to}\phi \mid}(\to R) \qquad \dfrac{}{\mid \gamma : \phi{\to}\phi \vdash \beta{:}\phi, \gamma{:}\phi{\to}\phi}}{\langle \lambda y.x \mid \gamma\rangle : x{:}\phi \vdash \beta{:}\phi, \gamma{:}\phi{\to}\phi}(cut)}{x{:}\phi \vdash \mu\beta.\langle \lambda y.x \mid \gamma\rangle : \phi \mid \gamma{:}\phi{\to}\phi}(\mu)}{\vdash \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle : \phi{\to}\phi \mid \gamma{:}\phi{\to}\phi}(\to R) \qquad \dfrac{}{\mid \gamma : \phi{\to}\phi \vdash \gamma{:}\phi{\to}\phi}}{\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle : \Gamma \vdash \gamma{:}\phi{\to}\phi}(cut)}{\Gamma \vdash \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle : \phi{\to}\phi \mid}(\mu)}{\Gamma \vdash \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle : \forall\phi.\phi{\to}\phi \mid}(\forall R)$$

and $\mathcal{D}_2$:

$$\dfrac{\dfrac{}{y{:}\forall\phi.\phi{\to}\phi \vdash y : (\phi'{\to}\phi')'{\to}\phi'{\to}\phi' \mid}(\forall v) \qquad \dfrac{\dfrac{\dfrac{}{y{:}\forall\phi.\phi{\to}\phi \vdash y : \phi'{\to}\phi' \mid}(\forall v) \qquad \dfrac{}{\mid \delta : \phi'{\to}\phi' \vdash \delta{:}\phi'{\to}\phi'}}{y{:}\forall\phi.\phi{\to}\phi \mid y{\cdot}\delta : (\phi'{\to}\phi')'{\to}\phi'{\to}\phi' \vdash \delta{:}\phi'{\to}\phi'}(\to L)}{\langle y \mid y{\cdot}\delta\rangle : \Gamma, y{:}\forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}(cut)}{y{:}\forall\phi.\phi{\to}\phi \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta\rangle : \forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}(\tilde{\mu})}$$

(notice that we cannot derive $y{:}\forall\phi.\phi{\to}\phi \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta\rangle : A{\to}A \vdash \delta{:}\phi'{\to}\phi'$, for any A, so the quantifier cannot be removed) then we can construct:

$$\dfrac{\boxed{\mathcal{D}_1}\ \Gamma \vdash \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle : \forall\phi.\phi{\to}\phi \mid \qquad \boxed{\mathcal{D}_2}\ y{:}\forall\phi.\phi{\to}\phi \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta\rangle : \forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}{\langle \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \mid \gamma\rangle \mid \gamma\rangle \mid \tilde{\mu}y.\langle y \mid y{\cdot}\delta\rangle\rangle : y{:}\forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}\,(cut)$$

[3] Essentially by Alexander J. Summers; see [43] for a detailed treatment of quantification in sequent calculi.

Now the μ-reduction

$$\langle \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x | \gamma\rangle \,|\, \gamma\rangle \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle \;\to_\mu\; \langle \lambda x.\mu\beta.\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle$$

will cause problems. Notice that $\gamma{:}\phi{\to}\phi$ is contracted in $\mathcal{D}_1$, so in fact occurs twice: bringing the quantifier inside, as would be required when μ-reducing, would force two independent closures (applications of rule $(\forall R)$) of the type $\phi{\to}\phi$. In fact, we would want to create a derivation like:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}\phi, y{:}\phi \vdash x : \phi \mid}}{x{:}\phi \vdash \lambda y.x : \phi{\to}\phi \mid}(\to R)}{x{:}\phi \vdash \lambda y.x : \forall\phi.\phi{\to}\phi \mid}(?) \quad \boxed{\mathcal{D}_2}\ y{:}\forall\phi.\phi{\to}\phi \mid \tilde{\mu}y.\langle y | y\cdot\delta\rangle : \forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}(cut)}{\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle : x{:}\phi \vdash \beta{:}\phi, \delta{:}\phi'{\to}\phi'}(\mu)}{x{:}\phi \vdash \mu\beta.\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle : \phi \mid \delta{:}\phi'{\to}\phi'}(\to R)}{\vdash \lambda x.\mu\beta.\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle : \phi{\to}\phi \mid \delta{:}\phi'{\to}\phi'}(\forall R)}{\vdots \quad \boxed{\mathcal{D}_2}\ y{:}\forall\phi.\phi{\to}\phi \mid \tilde{\mu}y.\langle y | y\cdot\delta\rangle : \forall\phi.\phi{\to}\phi \vdash \delta{:}\phi'{\to}\phi'}(cut)}{\langle \lambda x.\mu\beta.\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle : \Gamma \vdash \delta{:}\phi'{\to}\phi'}(\mu)}{\Gamma \vdash \mu\gamma.\langle \lambda x.\mu\beta.\langle \lambda y.x \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle \,|\, \tilde{\mu}y.\langle y | y\cdot\delta\rangle\rangle : \phi{\to}\phi \mid \delta{:}\phi'{\to}\phi'}(\forall R)}$$

this not a correct derivation: we would like to use rule $(\forall R)$ for $(?)$ but cannot, since ϕ is free in the left-hand environment $x{:}\phi$, which violates the side-condition on rule $(\forall R)$.

We now look at two restrictions of the notion of typing $\vdash_{\overline{\mathcal{M}}}$ that will avoid these two problematic cases, and will investigate if we can at least partially recover soundness by limiting reduction; we will look at both CBV and CBN. We will recover from the loss of soundness only partially, in that we will obtain a soundness result for both (limited) notions of reduction. Since the systems are restrictions of $\vdash_{\overline{\mathcal{M}}}$, union types are not allowed to appear in left-hand environments, and intersection types not in right-hand environments, completeness cannot be shown. This restriction on environments is used in the proofs below (Lemmae 10.4 and 10.2), so cannot be dropped.

The general idea of the restrictions is to make sure that a reduction towards an introduced intersection will be blocked by allowing rule $(\cap R)$ only for values; that way, if $\langle v | e\rangle$ is typeable, and $(\cap R)$ is used to type v, then v is not a μ-abstraction, so a reduction towards v is excluded. Likewise, a reduction towards an introduced union will be blocked by allowing rule $(\cup L)$ only for slots.

First we deal with CBN, as defined in Definition 4.6(*ii*), where we restricted the applicability of rule $(\cup L)$ to slots. It is clear that hereby reduction into a union is stopped, but how do we stop reducing into an intersection? Remember that this only happens in case $\langle \mu\alpha.c | v\cdot e\rangle$; in that case, the intersection type on the right (for $v\cdot e$) is superfluous, and only one proper type inside it is needed, as is shown in the following lemma:

Lemma 10.2 *If* $\Gamma \mid E : \cap_{\underline{n}} A_i \vdash_{\mathrm{N}} \Delta$, *then there exists* $\forall j \in \underline{n}$ *such that* $\Gamma \mid E : A_j \vdash_{\mathrm{N}} \Delta$.

Proof: By induction on the structure of derivations. If it ends with $(\cap L)$, the result is immediate; if it ends with $(\cup L)$, then there exists $B_i\,(\forall i \in \underline{m})$ such that $\cap_{\underline{n}} A_i = \cup_{\underline{m}} B_j$; this is only possible if $m = 1$ and $B_1 = \cap_{\underline{n}} A_i$, and the result follows by induction. Otherwise, we have

two cases: the derivation exists of $(Ax\text{-}L)$, or ends with $(\to L)$; then $n = 1$ and the result is immediate. □

Using this result, we can now show:

Theorem 10.3 (SOUNDNESS FOR $\to_{\mathrm{N}}$ IN $\vdash_{\mathrm{N}}$) *If* $c_1 : \Gamma \vdash_{\mathrm{N}} \Delta$ *and* $c_1 \to_{\mathrm{N}} c_2$, *then* $c_2 : \Gamma \vdash_{\mathrm{N}} \Delta$.

Proof: We focus on the main reduction rules:

$(\to)$: By Theorem 5.7.

$(\to_{\mu_{\mathrm{N}}})$: Then $c_1 = \langle \mu\alpha.c \,|\, E\rangle$, $c_2 = c\{E/\alpha\}$, and there exists A such that we have $\Gamma \vdash_{\mathrm{N}} \mu\alpha.c : A \mid \Delta$ and $\Gamma \mid E : A \vdash_{\mathrm{N}} \Delta$. We conclude by induction on the structure of the derivation for the first result.

(μ): then $c : \Gamma \vdash_{\mathrm{N}} \alpha{:}A, \Delta$, and, by Lemma 5.9, $c\{E/\alpha\} : \Gamma \vdash_{\mathrm{N}} \Delta$.

$(\cap R)$: then there exists $A_i\,(\forall i \in \underline{n})$ (with $n \geq 2$) such that, for all $i \in \underline{n}$, $\Gamma \mid E : \cap_{\underline{n}} A_i \vdash_{\mathrm{N}} \Delta$, and $c : \Gamma \vdash_{\mathrm{N}} \alpha{:}A_i, \Delta$. Then the derivation is shaped like:

$$\dfrac{\dfrac{\boxed{\mathcal{D}_1^i}}{c : \Gamma \vdash \alpha{:}A_i, \Delta}}{\Gamma \vdash \mu\alpha.c : A_i \mid \Delta}\,(\mu) \quad (\forall i \in \underline{n})}{\Gamma \vdash \mu\alpha.c : \cap_{\underline{n}} A_i \mid \Delta}\,(\cap R) \qquad \boxed{\mathcal{D}_2}\ \Gamma \mid E : \cap_{\underline{n}} A_i \vdash \Delta}{\langle \mu\alpha.c \,|\, E\rangle : \Gamma \vdash \Delta}\,(cut)$$

(by weakening, we can assume that only one left and right-hand environment is used). Since E is a slot, by Lemma 10.2, $\Gamma \mid E : A_j \vdash_{\mathrm{N}} \Delta$, for some $\forall j \in \underline{n}$; since also $c : \Gamma \vdash_{\mathrm{N}} \alpha{:}A_j, \Delta$, by Lemma 5.9 we get $c\{E/\alpha\} : \Gamma \vdash_{\mathrm{N}} \Delta$.

$(\cup R)$: Then the derivation is shaped like:

$$\dfrac{\dfrac{\boxed{}}{\Gamma \vdash \mu\alpha.c : A_j \mid \Delta}\ (\forall j \in \underline{n})}{\Gamma \vdash \mu\alpha.c : \cup_{\underline{n}} A_i \mid \Delta}\,(\cup R) \qquad \boxed{\mathcal{D}_2}\ \Gamma \mid E : \cup_{\underline{n}} A_i \vdash \Delta}{\langle \mu\alpha.c \,|\, E\rangle : \Gamma \vdash \Delta}\,(cut)$$

Since also $\Gamma \mid E : A_j \vdash_{\mathrm{N}} \Delta$ by rule $(\cup E)$ (admissible by Lemma 5.5), the result follows by induction.

$(\to_{\tilde{\mu}})$: Then $c_1 = \langle v \,|\, \tilde{\mu}x.c\rangle$, and $c_2 = c\{v/x\}$, and $\Gamma \vdash_{\mathrm{N}} v : A \mid \Delta$ and $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathrm{N}} \Delta$ for some A. We conclude by induction on the structure of the derivation for $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathrm{N}} \Delta$.

$(\tilde{\mu})$: Then $c : \Gamma, x{:}A \vdash_{\mathrm{N}} \Delta$, and by Lemma 5.8, $c\{v/x\} : \Gamma \vdash_{\mathrm{N}} \Delta$.

$(\cup L)$: Not applicable, since $(\cup L)$ has been restricted to slots.

$(\cap L)$: Then the derivation is shaped like:

$$\dfrac{\boxed{\phantom{\Gamma \vdash v : \cap_{\underline{n}} A_i \mid \Delta}} \qquad \dfrac{\boxed{\mathcal{D}_2}\ \Gamma \mid E : A_j \vdash \Delta}{\Gamma \mid E : \cap_{\underline{n}} A_i \vdash \Delta}\ (j \in \underline{n})\,(\cap L)}{\langle \mu\alpha.c \,|\, E\rangle : \Gamma \vdash \Delta}\,(cut)$$

Then, by rule $(\cap E)$ (admissible by Lemma 5.5), also $\Gamma \vdash_{\mathrm{N}} v : A_j \mid \Delta$, and the result follows by induction. □

For CBV, as defined in Definition 4.6(*i*), we take the dual approach, and restrict the use of $(\cap R)$ to values; this is similar to the approach in ML with side-effects, where rule $(\forall I)$ is

limited to values [27, 47, 36]. To avoid a μ-reduction into an intersection, we make sure that no intersection is introduced for $\mu\alpha.c$ via the restriction on $(\cap R)$. The only thing then to solve is to avoid the contraction of $\langle v|\tilde{\mu}x.c\rangle$ into a union, which is solved by the fact that then v is a value. We first show:

Lemma 10.4 *If* $\Gamma \vdash_{\mathrm{v}} V : \cup_{\underline{n}} A_i \mid \Delta$, *then there exists* $\forall j \in \underline{n}$ *such that* $\Gamma \vdash_{\mathrm{v}} V : A_j \mid \Delta$.

Proof: By induction on the structure of derivations. If it ends with $(\cup R)$, the result is immediate; if it ends with $(\cap R)$, then there exists $B_i\,(\forall i \in \underline{m})$ such that $\cup_{\underline{n}} A_i = \cap_{\underline{m}} B_j$; this is only possible if $n = 1$ and $A_1 = \cup_{\underline{m}} B_j$, and the result follows by induction. Otherwise, we have two cases: the derivation exists of $(Ax\text{-}R)$ or ends with $(\to R)$, and then $n = 1$ and the result is immediate.

We conclude this paper by showing:

Theorem 10.5 (Soundness for $\to_{\mathrm{v}}$ in $\vdash_{\mathrm{v}}$) *If* $c_1 : \Gamma \vdash_{\mathrm{v}} \Delta$ *and* $c_1 \to_{\mathrm{v}} c_2$, *then* $c_2 : \Gamma \vdash_{\mathrm{v}} \Delta$.

Proof: We focus on the main reduction rules:

$(\to)$: By Theorem 5.7.

$(\to_{\mu})$: Then $c_1 = \langle \mu\alpha.c|e\rangle$, $c_2 = c\{e/\alpha\}$, and both $\Gamma \vdash_{\mathrm{v}} \mu\alpha.c : A \mid \Delta$ and $\Gamma \mid e : A \vdash_{\mathrm{v}} \Delta$ for some A. By induction on the structure of derivations for $\Gamma \vdash_{\mathrm{v}} \mu\alpha.c : A \mid \Delta$:

(μ): Then $c : \Gamma \vdash_{\mathrm{v}} \alpha{:}A, \Delta$, and by Lemma 5.9, $c\{e/\alpha\} : \Gamma \vdash_{\mathrm{v}} \Delta$.

$(\cap R)$: Impossible, since $(\cap R)$ has been restricted to values.

$(\cup R)$: Then the derivation is shaped like:

$$\dfrac{\dfrac{\boxed{}}{\Gamma \vdash \mu\alpha.c : A_j \mid \Delta} \quad (j \in \underline{n})}{\Gamma \vdash \mu\alpha.c : \cup_{\underline{n}} A_i \mid \Delta}\,(\cup R) \qquad \boxed{\Gamma \vdash e : \cup_{\underline{n}} A_i \mid \Delta}}{\langle \mu\alpha.c|e\rangle : \Gamma \vdash \Delta}\,(cut)$$

By $(\cup E)$ also $\Gamma \mid e : A_j \vdash_{\mathrm{v}} \Delta$, and the result follows by induction.

$(\to_{\tilde{\mu}\mathrm{v}})$: Then $c_1 = \langle V|\tilde{\mu}x.c\rangle$, and $c_2 = c\{V/x\}$, and $\Gamma \vdash_{\mathrm{v}} V : A \mid \Delta$ and $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathrm{v}} \Delta$ for some A. By induction on the structure of derivation for $\Gamma \mid \tilde{\mu}x.c : A \vdash_{\mathrm{v}} \Delta$.

$(\tilde{\mu})$: then $c : \Gamma, x{:}A \vdash_{\mathrm{v}} \Delta$, and, by Lemma 5.8, $c\{V/x\} : \Gamma \vdash_{\mathrm{v}} \Delta$.

$(\cup L)$: then there exists $A_i\,(\forall i \in \underline{n})$ (without loss of generality, $n \geq 2$) such that, for all $i \in \underline{n}$, $\Gamma \mid V : \cap_{\underline{n}} A_i \vdash_{\mathrm{v}} \Delta$, and $c : \Gamma \vdash_{\mathrm{v}} \alpha{:}A_i, \Delta$. Then the derivation is shaped like:

$$\dfrac{\boxed{\Gamma \vdash V : \cup_{\underline{n}} A_i \mid \Delta} \qquad \dfrac{\dfrac{\boxed{\phantom{c : \Gamma, x{:}A_i \vdash \Delta}}}{c : \Gamma, x{:}A_i \vdash \Delta}}{\Gamma \mid \tilde{\mu}x.c : A_i \vdash \Delta} \quad (\forall i \in \underline{n})}{\Gamma \vdash \tilde{\mu}x.c : \cup_{\underline{n}} A_i \mid \Delta}\,(\cup L)}{\langle V|\tilde{\mu}x.c\rangle : \Gamma \vdash \Delta}\,(cut)$$

Since V is a value, by Lemma 10.4, $\Gamma \vdash_{\mathrm{v}} V : A_j \mid \Delta$ for some $\forall j \in \underline{n}$; since $c : \Gamma, x{:}A_j \vdash_{\mathrm{v}} \Delta$, we get $c\{V/x\} : \Gamma \vdash_{\mathrm{v}} \Delta$ by Lemma 5.9.

$(\cap L)$: Then the derivation is shaped like:

$$\dfrac{\boxed{\Gamma \vdash V : \cap_{\underline{n}} A_i \mid \Delta} \qquad \dfrac{\boxed{\phantom{\Gamma \vdash \tilde{\mu}x.c : A_j \mid \Delta}}}{\Gamma \vdash \tilde{\mu}x.c : A_j \mid \Delta} \quad (j \in \underline{n})}{\Gamma \vdash \tilde{\mu}x.c : \cap_{\underline{n}} A_i \mid \Delta}\,(\cap L)}{\langle V|\tilde{\mu}x.c\rangle : \Gamma \vdash \Delta}\,(cut)$$

By $(\cap E)$ also $\Gamma \vdash_{\mathrm{v}} V : A_j \mid \Delta$, and the result follows by induction. □

11 Concluding remarks

We have seen that which notion of intersection and union type assignment to use for $\overline{\lambda}\mu\tilde{\mu}$ is not as easily decided as might seem. The most natural approach, inspired by the superficial correspondence between intersection types and the logical connector $\wedge$ and a quick analysis of the problem of completeness resulted in Dougherty, Ghilezan and Lescanne's System $\mathcal{M}^{\cap\cup}$, as presented in [18]. Although this gives a perfectly reasonable system in terms of the structure of rules, it does not satisfy almost any of the important properties. First of all, soundness fails for this system, as also remarked by Herbelin, and commented on in [19]; in fact, the system $\mathcal{M}^{\cap}$ presented in that paper is a modification of $\mathcal{M}^{\cap\cup}$, modified in order to recover from that flaw. As shown in this paper, this attempt failed. But, more unexpectedly, also completeness fails for both $\mathcal{M}^{\cap\cup}$ and $\mathcal{M}^{\cap}$, as was shown in this paper. This is the more remarkable, seen that the main role of intersection is to deal exactly with completeness, and System $\mathcal{M}^{\cap\cup}$ is defined in much the same way as the BCD system was.

System $\mathcal{M}^{\cap\cup}$ is just *one* of the possible ways of dealing with intersection and union in the context of $\overline{\lambda}\mu\tilde{\mu}$. In fact, allowing intersection and union for activated formulae only actually leans too strongly on the similarity between those type constructs and the logical $\wedge$ and $\vee$. We have seen that *none* of the important (and expected) properties (soundness and completeness, conservativeness with respect to *Système Dω*) hold, and that in order to achieve completeness the system at least needs to allow union and intersection freely in both left and right-hand environment, and that both need to be assigned also to *in*active statements, by adding new rules, resulting in System $\mathcal{M}^{\mathrm{c}}$. However, this does not solve the soundness problem at all.

The approach followed (essentially, since it deals with a different calculus) in [4], is to not depend on the logical foundation at all any more, and to allow $(\cup L)$ and $(\cap R)$ *only* for commands:

$$(\cap R): \frac{c : \Gamma_i \vdash_{\overline{\lambda}} \alpha{:}A_i, \Delta_i \quad (\forall i \in \underline{n})}{c : \cap_{\underline{n}}\Gamma_i \vdash_{\overline{\lambda}} \alpha{:}\cap_{\underline{n}}A_i, \cup_{\underline{n}}\Delta_i}(n \geq 0) \qquad (\cup L): \frac{c : \Gamma_i, x{:}A_i \vdash_{\overline{\lambda}} \Delta_i \quad (\forall i \in \underline{n})}{c : \cap_{\underline{n}}\Gamma_i, x{:}\cup_{\underline{n}}A_i \vdash_{\overline{\lambda}} \cup_{\underline{n}}\Delta_i}(n \geq 0)$$

It is still possible then to preserve the types assignable in the $\vdash_{\cap}$-system, but we would need a different interpretation function that explicitly names all outputs.

As in similar notions for the λ-calculus, combining union and intersection types breaks the soundness of the system; unlike the λ-calculus, however, also intersection is problematic. We have isolated the problem cases, and seen that it is exactly the non-logical behaviour of *both* type constructors that causes the problem. We have looked at restrictions for either CBN or CBV reduction that overcome this defect, but all with the loss of the completeness.

We have argued that any full system of intersection and union types is doomed to fail. We presented System $\mathcal{M}^{\mathrm{c}}$ – an extension of $\mathcal{M}^{\cap\cup}$ – that satisfies completeness, but could not fix soundness. Only by limiting the notion of reduction (in our case, to either Call-By-Name of Call-By-Value reduction), and the applicability of some type assignment rules can we come to sound notions of type assignment, but at the price of sacrificing completeness. Perhaps we can recover from this failure for CBN and CBV, but would need a different approach - we leave this for future work; as we have show in detail, there is no reprieve for unrestricted reduction.

So we find that it is impossible to define a semantics using intersection and union types for the symmetric notion of reduction that is in $\overline{\lambda}\mu\tilde{\mu}$. This does not exclude the characterisation results that are mentioned in the introduction; although a correct proof is still missing, it is perfectly possible that all terms typeable in our System $\mathcal{M}^{\mathrm{c}}$ are strongly normalisable - and

by extension also those of $\mathcal{M}^{\cap\cup}$; the proof might depend on a proof that derivation reduction is strongly normalisable, as also used in [3, 6, 5]; this is left for future work.

Acknowledgements

I would like to thank Silvia Ghilezan, Pierre Lescanne, Mariangiola Dezani, Alexander J. Summers and especially Dan Dougherty for detailed, critical, and constructive discussions on the results of this paper, and Vanessa Loprete for valuable support.

References

[1] S. van Bakel. Complete restrictions of the Intersection Type Discipline. *Theoretical Computer Science*, 102(1):135–163, 1992.

[2] S. van Bakel. Intersection Type Assignment Systems. *Theoretical Computer Science*, 151(2):385–435, 1995.

[3] S. van Bakel. Cut-Elimination in the Strict Intersection Type Assignment System is Strongly Normalising. *Notre Dame journal of Formal Logic*, 45(1):35–63, 2004.

[4] S. van Bakel. Reduction in $\mathcal{X}$ does not agree with Intersection and Union Types. In *Electronic Proceedings of 4th International Workshop* Intersection Types and Related Systems *(ITRS'08), Turin, Italy*, 2008.

[5] S. van Bakel. Strict Intersection Types for the Lambda Calculus. To appear in ACM Surveys, 2010.

[6] S. van Bakel and M. Fernández. Normalisation, Approximation, and Semantics for Combinator Systems. *Theoretical Computer Science*, 290:975–1019, 2003.

[7] S. van Bakel, S. Lengrand, and P. Lescanne. The language $\mathcal{X}$: circuits, computations and Classical Logic. In Mario Coppo, Elena Lodi, and G. Michele Pinna, editors, *Proceedings of Ninth Italian Conference on Theoretical Computer Science* (ICTCS'05), *Siena, Italy*, volume 3701 of *Lecture Notes in Computer Science*, pages 81–96. Springer Verlag, 2005.

[8] S. van Bakel and P. Lescanne. Computation with Classical Sequents. *Mathematical Structures in Computer Science*, 18:555–609, 2008.

[9] F. Barbanera, M. Dezani-Ciancaglini, and U. de'Liguoro. Intersection and Union Types: Syntax and Semantics. *Information and Computation*, 119(2):202–230, 1995.

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

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

[12] G. M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In *Proceedings of Symposium on Mathematical Foundations of Computer Science.*, volume 1450 of *Lecture Notes in Computer Science*, pages 336–345. Springer Verlag, 1998.

[13] R. Bloo and K.H. Rose. Preservation of Strong Normalisation in Named Lambda Calculi with Explicit Substitution and Garbage Collection. In *CSN'95 – Computer Science in the Netherlands*, pages 62–72, 1995.

[14] A. Church. A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1(1):40–41, 1936.

[15] M. Coppo and M. Dezani-Ciancaglini. A New Type Assignment for Lambda-Terms. *Archive für Mathematischer Logic und Grundlagenforschung*, 19:139–156, 1978.

[16] P.-L. Curien and H. Herbelin. The Duality of Computation. In *Proceedings of the 5th ACM SIGPLAN International Conference on Functional Programming (ICFP'00)*, volume 35.9 of *ACM Sigplan Notices*, pages 233–243. ACM, 2000.

[17] R. Davies and F. Pfenning. Intersection types and computational effects. In *Proceedings of the 5th ACM SIGPLAN International Conference on Functional Programming (ICFP'00)*, volume 35.9 of *ACM Sigplan Notices*, pages 198–208. ACM Press, 2000.

[18] D. Dougherty, S. Ghilezan, and P. Lescanne. Intersection and Union Types in the $\overline{\lambda}\mu\tilde{\mu}$-calculus. In *Electronic Proceedings of 2nd International Workshop* Intersection Types and Related Systems *(ITRS'04), Turku, Finland*, volume 136 of *Electronic Notes in Theoretical Computer Science*, pages 228–246, 2004.

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

[20] D. Dougherty, S. Ghilezan, P. Lescanne, and S. Likavec. Strong Normalization of the Dual Classical Sequent Calculus. In *Proceedings of 12th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'05)*, volume 3835 of *Lecture Notes in Computer Science*, pages 169–183, 2005.

[21] J. Dunfield and F. Pfenning. Type Assignment for Intersections and Unions in Call-by-Value Languages. In Andrew D. Gordon, editor, *Proceedings of 6th International Conference on* Foundations of Software Science and Computational Structures *(FOSSACS'03)*, pages 250–266, 2003.

[22] M. Felleisen, D. P. Friedman, E. Kohlbecker, and B. Duba. A syntactic theory of sequential control. *Theoretical Computer Science*, 52:205–237, 1987.

[23] G. Gentzen. Investigations into logical deduction. In *The Collected Papers of Gerhard Gentzen*. Ed M. E. Szabo, North Holland, 68ff (1969), 1935.

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

[25] T. Griffin. A formulae-as-types notion of control. In *Proceedings of the 17th Annual ACM Symposium on Principles Of Programming Languages, Orlando (Fla., USA)*, pages 47–58, 1990.

[26] Ph. de Groote. On the relation between the $\lambda\mu$-calculus and the syntactic theory of sequential control. In *Proceedings of 5th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'94)*, volume 822 of *Lecture Notes in Computer Science*, pages 31–43. Springer Verlag, 1994.

[27] B. Harper and M. Lillibridge. ML with callcc is unsound. Post to TYPES mailing list, July 8, 1991.

[28] H. Herbelin. *Séquents qu'on calcule : de l'interprétation du calcul des séquents comme calcul de λ-termes et comme calcul de stratégies gagnantes*. Thèse d'université, Université Paris 7, Janvier 1995.

[29] H. Herbelin. *C'est maintenant qu'on calcule: au cœur de la dualité*. Mémoire de habilitation, Université Paris 11, Décembre 2005.

[30] Hugo Herbelin. A Lambda-Calculus Structure Isomorphic to Gentzen-Style Sequent Calculus Structure. In Leszek Pacholski and Jerzy Tiuryn, editors, *Computer Science Logic, 8th International Workshop, CSL '94, Kazimierz, Poland, September 25-30, 1994, Selected Papers*, volume 933 of *Lecture Notes in Computer Science*, pages 61–75. Springer Verlag, 1995.

[31] J.R. Hindley. Coppo-Dezani Types do not Correspond to Propositional Logic. *Theoretical Computer Science*, 28:235–236, 1984.

[32] H. Ishihara and T. Kurata. Completeness of intersection and union type assignment systems for call-by-value λ-models. *Theoretical Computer Science*, 272(1–2):197–221, 2002.

[33] S.C. Kleene. *Introduction to Metamathematics*. Études et Recherches en Informatique. North Holland, Amsterdam, 1952.

[34] J.-L. Krivine. *Lambda calculus, types and models*. Ellis Horwood, 1993.

[35] S. Lengrand. Call-by-value, call-by-name, and strong normalization for the classical sequent calculus. In Bernhard Gramlich and Salvador Lucas, editors, *Post-proceedings of the 3rd Workshop on Reduction Strategies in Rewriting and Programming (WRS 2003)*, volume 86 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003.

[36] R. Milner, M. Tofte, R. Harper, and D. MacQueen. *The Definition of Standard ML*. MIT Press, 1990. Revised edition.

[37] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *Proceedings of the 24th Annual ACM Symposium on Principles Of Programming Languages, Paris (France)*, pages 215–227, 1997.

[38] M. Parigot. An algorithmic interpretation of classical natural deduction. In *Proceedings of 3rd International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'92)*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer Verlag, 1992.

[39] M. Parigot. Classical Proofs as Programs. In *Kurt Gödel Colloquium*, pages 263–276, 1993. Presented at TYPES Workshop, at Bǎstad, June 1992.

[40] G. Pottinger. A Type Assignment for the Strongly Normalizable λ-terms. In J.P. Seldin and J.R. Hindley, editors, *To H. B. Curry, Essays in combinatory logic, lambda-calculus and formalism*, pages 561–577. Academic press, New York, 1980.

[41] P. Sallé. Une extension de la théorie des types. In G. Ausiello and C. Böhm, editors, *Automata, languages and programming. Fifth Colloquium,* Udine, Italy, volume 62 of *Lecture Notes in Computer Science*, pages 398–410. Springer Verlag, 1978.

[42] A. Summers and S. van Bakel. Approaches to Polymorphism in Classical Sequent Calculus. In P. Sestoft, editor, *Proceedings of 15th European Symposium on Programming (ESOP'06),* Vienna, Austria, volume 3924 of *Lecture Notes in Computer Science*, pages 84 – 99. Springer Verlag, 2006.

[43] A.J. Summers. *Curry-Howard Term Calculi for Gentzen-Style Classical Logic*. PhD thesis, Imperial College London, 2008.

[44] C. Urban. *Classical Logic and Computation*. PhD thesis, University of Cambridge, October 2000.

[45] C. Urban and G. M. Bierman. Strong normalisation of cut-elimination in classical logic. *Fundamenta Informaticae*, 45(1,2):123–155, 2001.

[46] P. Wadler. Call-by-Value is Dual to Call-by-Name. In *Proceedings of the eighth ACM SIGPLAN international conference on Functional programming*, pages 189 – 201, 2003.

[47] A. K. Wright. Simple imperative polymorphism. *Lisp and Symbolic Computation*, 8(4):343–355, 1995.