Intersection Types for the $\lambda\mu$-Calculus

(Logical Methods in Computer Science 14(1) (2018))

Steffen van Bakel[1], Franco Barbanera[2], and Ugo de'Liguoro[3]

[1]Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2BZ, UK
[2]Dipartimento di Matematica e Informatica, Università degli Studi di Catania, Viale A. Doria 6, 95125 Catania, Italy
[3]Dipartimento di Informatica, Università degli Studi di Torino, Corso Svizzera 185, 10149 Torino, Italy

(With corrections with respect to the published version*)

Abstract

We introduce an intersection type system for the $\lambda\mu$-calculus that is invariant under subject reduction and expansion. The system is obtained by describing Streicher and Reus's denotational model of continuations in the category of ω-algebraic lattices via Abramsky's domain-logic approach. This provides at the same time an interpretation of the type system and a proof of the completeness of the system with respect to the continuation models by means of a filter model construction.
We then define a restriction of our system, such that a $\lambda\mu$-term is typeable if and only if it is strongly normalising. We also show that Parigot's typing of $\lambda\mu$-terms with classically valid propositional formulas can be translated into the restricted system, which then provides an alternative proof of strong normalisability for the typed $\lambda\mu$-calculus.

keywords: $\lambda\mu$-calculus, intersection types, filter semantics, strong normalisation.

Introduction

The $\lambda\mu$-calculus is a type-free calculus introduced by Parigot [46] to denote classical proofs and to compute with them. It is an extension of the proofs-as-programs paradigm where types can be understood as classical formulas and (closed) terms inhabiting a type as the respective proofs in a variant of Gentzen's natural deduction calculus for classical logic [32]. The study of the syntactic properties of the $\lambda\mu$-calculus has been challenging, which led to the introduction of variants of term syntax, reduction rules, and typing as, for example, in de Groote's variant of the $\lambda\mu$-calculus [35]. These changes have an impact on the deep nature of the calculus which emerges both in the typed and in the untyped setting [25, 51].

Types are of great help in understanding the computational properties of terms in an abstract way. Although in [16] Barendregt treats the theory of the pure λ-calculus without a reference to types, most of the fundamental results of the theory can be exposed in a quite elegant way by using the Coppo-Dezani intersection type system [21]. This is used by Krivine [38], where the treatment of the pure λ-calculus relies on intersection typing systems called $\mathcal{D}$ and $\mathcal{D}\Omega$.

The quest for more expressive notions of typing for $\lambda\mu$ is part of an ongoing investigation into calculi for classical logic. In order to come to a characterisation of strong normalisation for Curien and Herbelin's (untyped) sequent calculus $\overline{\lambda}\mu\tilde{\mu}$ [24], Dougherty, Ghilezan and

* In particular Lemma 5.2 and 5.3, Definition 6.13, and Lemma 6.14.

Lescanne presented System $\mathcal{M}^{\cap\cup}$ [28, 29], that defines a notion of intersection and union typing for that calculus. However, in [8] van Bakel showed that this system is not closed under conversion, an essential property of Coppo-Dezani systems; in fact, it is shown that it is *impossible* to define a notion of typing for $\overline{\lambda}\mu\tilde{\mu}$ that satisfies that property.

In [9] van Bakel brought intersection (and union) types to the context of the (untyped) $\lambda\mu$-calculus, and showed that for $\lambda\mu$-conversion *it is possible* to prove type preservation under conversion. However, union types are no longer dual to intersection types and play only a marginal role, as was also the intention of [29]. In particular, the normal $(\cup I)$ and $(\cup E)$ rules as used in [15], which are known to create a soundness problem in the context of the λ-calculus, are not allowed. In the view of the above mentioned failure noted in [8], the result of [9] came as a surprise, and led automatically to the question we answer here: does a *filter semantics* for $\lambda\mu$ exist?

The idea of building a λ-model out of a suitable type assignment system appeared first in [17]. In that system types are an extension of simple types with the binary operator $\wedge$ for intersection, and are pre-ordered by an axiomatisable (in fact decidable) relation $\leq$; if types are interpreted by subsets of the domain D (an applicative structure satisfying certain conditions), one can see $\wedge$ as set theoretic intersection and $\leq$ as containment. The discovery of [17] is that, taking a proper relation $\leq$, the set $\mathcal{F}_D$ of filters of types (sets of types closed under type intersection and $\leq$) is a λ-model, where (closed) terms can be interpreted by the set of types that can be assigned to them. This is what is called a filter semantics.

It emerged in [22] that models constructed as set of filters of intersection types are exactly the ω-algebraic lattices, a category of complete lattices, but with Scott-continuous maps as morphisms. ω-algebraic lattices are posets whose structure is fully determined by a countable subset of elements, called 'compact points' for topological reasons. Now the crucial fact is that given an ω-algebraic lattice D, the set $\mathcal{K}(D)$ of its compact points can be described by putting its elements into a one-to-one correspondence with a suitable set of intersection types, in such a way that the order over $\mathcal{K}(D)$ is reflected by the inverse of the $\leq$ pre-order over types. Then one can show that the filter structure $\mathcal{F}_D$ obtained from the type pre-order is isomorphic with the original D. In fact, Abramsky proved that this is not true only of ω-algebraic lattices, but of quite larger categories of domains, like 2/3-SFP domains that can be finitely described by a generalisation of intersection type theories, called the *logics* of the respective domains in [1].

Here, instead of defining a suitable type system for $\lambda\mu$, and then trying to prove that it actually induces a filter model, we follow the opposite route. We start from a model of the $\lambda\mu$-calculus in ω**-AlgL**, the category of ω-algebraic lattices. We then distill the type syntax and the corresponding type theory out of the construction of the model, and recover the typing rules from the clauses that define term interpretation in the given model $\mathcal{F}_D$ that is by construction isomorphic to the given D.

However, things are more complex than this. First we need a domain theoretic model of $\lambda\mu$; we use for that purpose Streicher and Reus's *models of continuations*. Building on Lafont's ideas and the papers [39, 45], in [52] Streicher and Reus proposed a model of both typed and untyped λ-calculi embodying a concept of continuation, including Felleisen's $\lambda\mathcal{C}$-calculus [31, 30] and a version of Parigot's $\lambda\mu$. The model is based on the solution of the domain equations $D = C \to R$ and $C = D \times C$, where R is an arbitrary domain of 'results'. The domain C is set of what are called 'continuations' in [52], which are infinite tuples of elements in D. D is the domain of continuous functions from C to R and is the set of 'denotations' of terms. We call the triple (R, D, C) a $\lambda\mu$-model, that exists in ω**-AlgL** by the inverse limit technique, provided that $R \in \omega$**-AlgL**.

The next step is to find type languages $\mathcal{L}_D$ and $\mathcal{L}_C$, and type theories axiomatising the re-

spective pre-orders $\leq_D$ and $\leq_C$, such that D and C are isomorphic to $\mathcal{F}_D$ and $\mathcal{F}_C$, respectively. To this aim we may suppose that logical description of R is given via a language of types $\mathcal{L}_R$ and a pre-order $\leq_R$. But then we need a detailed analysis of $\mathcal{K}(D)$ and $\mathcal{K}(C)$, keeping into account that D and C are both co-limits of certain chains of domains, and that their compact points are into one-to-one correspondence with the union of the compact points of the domains approximating D and C. This leads us to a mutually inductive definition of $\mathcal{L}_D$ and $\mathcal{L}_C$ and of $\leq_D$ and $\leq_C$. In this way, we obtain an extension of the type theory used in [17], which is a *natural equated* intersection type theory in terms of [2] and hence is isomorphic to the inverse limit construction of a D_∞ λ-model (as an aside, we observe that this matches perfectly with Theorem 3.1 in [52]).

Once the filter domains $\mathcal{F}_D$ and $\mathcal{F}_C$ have been constructed, we can consider the interpretation of terms and of commands ('unnamed' and 'named terms' respectively in Parigot's terminology). Following [52], we define the interpretation of expressions of Parigot's $\lambda\mu$-calculus inductively via a set of equations. Guided by these equations in the particular case of $\mathcal{F}_D$ and $\mathcal{F}_C$, and considering the correspondence we have established among types and compact points, we are able to reconstruct the inference rules of a type assignment system which forms the main contribution of our work.

The study of the properties of the system produces a series of results that confirm the validity of the construction. First we prove that in the filter model the meaning of M in the environment e, denoted by $[\![M]\!]e$, coincides with the filter of all types $\delta \in \mathcal{L}_D$ such that $\Gamma \vdash M : \delta \mid \Delta$ is derivable in the system, for Γ and Δ such that e satisfies both Γ and Δ, and similarly for $[\![\mathtt{c}]\!]e$, where $\mathtt{c}$ is a command. Then if two terms or commands are convertible, they must have the same types. In fact, we will prove this result twice: first abstractly, making essential use of the filter model construction; then concretely, by studying in depth the structure of the derivations in our system, and establishing that types are preserved under subject reduction and expansion.

We then face the problem of characterising strong normalisation in the case of $\lambda\mu$. This is a characteristic property of intersection types, stated the first time by Pottinger [49] for the ordinary λ-calculus: strongly normalising terms can be captured by certain 'restricted' type systems, ruling out the universal type ω. As will be apparent in the technical treatment, we cannot simply restrict our system by removing ω; however, the characterisation can be obtained by distinguishing certain 'good' occurrences of ω that cannot be eliminated, and the 'bad' ones that must be forbidden. This is still guided by the semantics and by the proof theoretic study of the system, and we can establish that there exists a subsystem of our system that is determined just by a restriction on type syntax, plus the elimination of the rule (ω) from our type system.

We conclude by looking at the relation between our type system and the original one proposed by Parigot [46] on the basis of the Curry-Howard correspondence between types and formulas and $\lambda\mu$-terms and proofs of classical logic. We show that there exists an interpretation of Parigot's first order types into intersection types such that the structure of derivations is preserved; moreover, since the translations are all restricted intersection types, we obtain a new proof that all proof-terms in $\lambda\mu$, *i.e.* those typeable in Parigot's system (even extended with negation), are strongly normalising.

This paper is the full version of [11], extended with a revised version of [12].

Outline of this paper

The paper is organised as follows. After recalling the $\lambda\mu$-calculus in Sect. 1, we study the domain theoretic models in Sect. 2. In Sect. 3 we introduce intersection types and type theories and we illustrate the filter model construction. The main part of the paper is Sect. 4, where we introduce the type assignment system; we study type invariance under reduction and expansion in Sect. 5. Sect. 6 is devoted to the characterisation of strongly normalising terms by means of a subsystem of ours obtained by suitably restricting the type syntax. Then, in Sect. 7, we compare our system with Parigot's, and show that Parigot's types are translatable into our restricted types while preserving type derivability (in the two systems). We finish by discussing some related work in Sect. 8 and draw our conclusions.

1 The untyped $\lambda\mu$-calculus

The $\lambda\mu$-calculus, as introduced in [46], is an extension of the untyped λ-calculus obtained by adding *names* and a name-abstraction operator, μ. It was intended as a proof calculus for a fragment of classical logic. Logical formulas of the implicational fragment of the propositional calculus can be assigned as types to $\lambda\mu$-terms much in the formulae-as-types paradigm of the Curry-Howard correspondence between typed λ-calculus and intuitionistic logic. With $\lambda\mu$ Parigot created a multi-conclusion typing system. In the notation of [51], the derivable statements have the shape $\Gamma \vdash M : A \mid \Delta$, where A is the main conclusion of the statement, expressed as the *active* conclusion, and Δ contains the alternative conclusions, consisting of pairs of names and types; the left-hand context Γ, as usual, is a mapping from term variables to types, and represents the assumptions about free term variables of M.

As with implicative intuitionistic logic, the reduction rules for the terms that represent proofs correspond to proof contractions; the difference is that the reduction rules for the λ-calculus are the *logical* reductions, *i.e.* deal with the elimination of a type construct that has been introduced directly above. In addition to these, Parigot expressed also the *structural* rules, where elimination takes place for a type constructor that appears in one of the alternative conclusions (the Greek variable is the name given to a subterm): he therefore needed to express that the focus of the derivation (proof) changes, and this is achieved by extending the syntax with two new constructs $[\alpha]M$ and $\mu\alpha.M$ that act as witness to *deactivation* and *activation*, which together move the focus of the derivation.

$\lambda\mu$ was conceived in the spirit of Felleisen's $\lambda\mathcal{C}$-calculus, that Griffin showed to be typeable with classical propositional logic in [34]. The $\lambda\mu$-calculus is type free and uses names and μ to model a form of functional programming with control [35].

Here we briefly revise the basic notions of the $\lambda\mu$-calculus, though slightly changing the notation and terminology, and defer the presentation of the typed $\lambda\mu$-calculus to Sect. 7.

Definition 1.1 (Term Syntax [46]) The sets Trm of *terms* (ranged over by M, N, L) and Cmd of *commands* (ranged over by $\mathtt{c}$) are defined inductively by the following grammar, where $x \in$ Var, the set of *term variables* (ranged over by x, y, z) and $\alpha \in$ Name, the set of *names* (ranged over by α, β, γ), both denumerable:

$$\begin{array}{rcll} M, N & ::= & x \mid \lambda x.M \mid MN \mid \mu\alpha.\mathtt{c} & \text{(terms)} \\ \mathtt{c} & ::= & [\alpha]M & \text{(commands)} \end{array}$$

We let T range over Trm $\cup$ Cmd.

As usual, $\lambda x.M$ binds x in M, and $\mu\alpha.\mathtt{c}$ binds α in $\mathtt{c}$. A variable or a name occurrence is free in a term if it occurs and is not bound: we denote the free variables and the free names

occurring in T by $fv(T)$ and $fn(T)$, respectively.

We identify terms and commands obtained by renaming of free variables or names, and we adopt Barendregt's convention that free and bound variables and names are distinct, assuming silent α-conversion during reduction to avoid capture. We will extend this convention to also consider occurrences of x and α bound over M in type judgements like $\Gamma, x{:}A \vdash M : B \mid \alpha{:}C, \Delta$ (see Sect. 4).

In [46] terms and commands are called 'terms' and 'named terms', respectively; names are called μ-variables, but might be better understood as 'continuation variables' (see [52]). Since this would imply a commitment to a particular interpretation, we prefer a more neutral terminology.

In the $\lambda\mu$-calculus, substitution takes the following three forms:

term substitution:	$T[N/x]$	(N is substituted for x in T)
renaming:	$T[\alpha/\beta]$	(β in T is replaced by α)
structural substitution:	$T[\alpha \Leftarrow L]$	(every subterm $[\alpha]N$ of T is replaced by $[\alpha]NL$)

In particular, *structural substitution* is defined by induction over terms and commands as follows:

Definition 1.2 (Structural Substitution) The key case for the structural substitution is defined as:

$$([\alpha]M)[\alpha \Leftarrow L] \triangleq [\alpha](M[\alpha \Leftarrow L])L$$

The other cases are defined as:

$$\begin{array}{rcl} x[\alpha \Leftarrow L] & \triangleq & x \\ (\lambda x.M)[\alpha \Leftarrow L] & \triangleq & \lambda x.M[\alpha \Leftarrow L] \\ (MN)[\alpha \Leftarrow L] & \triangleq & (M[\alpha \Leftarrow L])(N[\alpha \Leftarrow L]) \\ \\ (\mu\beta.\mathtt{c})[\alpha \Leftarrow L] & \triangleq & \mu\beta.\mathtt{c}[\alpha \Leftarrow L] \\ ([\beta]M)[\alpha \Leftarrow L] & \triangleq & [\beta]M[\alpha \Leftarrow L] \quad (\text{if } \alpha \neq \beta) \end{array}$$

Notice that the first case places the argument of the substitution to the right of a term with name α, and the others propagate the substitution towards subterms that are named α.

The reduction relation for the $\lambda\mu$-calculus is defined as follows.

Definition 1.3 (Reduction $\to_{\beta\mu}$ [46]) The reduction relation $\to_{\beta\mu}$ is the compatible closure of the following rules:

$$\begin{array}{rrcl} (\beta): & (\lambda x.M)N & \to & M[N/x] \\ (\mu): & (\mu\alpha.\mathtt{c})N & \to & \mu\alpha.\mathtt{c}[\alpha \Leftarrow N]^1 \\ (ren): & [\alpha]\mu\beta.\mathtt{c} & \to & \mathtt{c}[\alpha/\beta]. \end{array}$$

Note that Rule (β) is the normal β-reduction rule of the λ-calculus. Rule (μ) is characteristic for $\lambda\mu$; the intuition behind this rule has been explained by de Groote in [35], by arguing on the grounds of the intended typing of the μ-terms: '*in a $\lambda\mu$-term $\mu\alpha.M$ of type $A \to B$, only the subterms named by α are* really *of type $A \to B$ (...); hence, when such a μ-abstraction is applied to an argument, this argument must be passed over to the sub-terms named by α.*'. The 'renaming' rule

[1] This is the common notation for this rule, although one could argue that a better formulation would be: $(\mu\alpha.\mathtt{c})N \to \mu\gamma.\mathtt{c}[\alpha \Leftarrow N{\cdot}\gamma]$ where γ is fresh, and let the structural substitution rename the term: $([\alpha]M)[\alpha \Leftarrow L{\cdot}\gamma] = [\gamma](M[\alpha \Leftarrow L{\cdot}\gamma])L$; in fact, when making the substitution *explicit* (see [14]), this becomes necessary. This is reflected in Ex. 5.7, where before the reduction $(\mu\alpha.[\alpha]x)x \to \mu\alpha.[\alpha]xx$, α has type $\delta{\times}\kappa$, and after it has type κ.

(ren) is called 'structural reduction' in [46] and rule (ρ) in [50]; it is an auxiliary notion of reduction, aimed at simplifying proof terms.

Definition 1.4 (The theory $\lambda\mu$) The theory $\lambda\mu$ is the equational theory determined by the compatible closure of: $M \to_{\beta\mu} N \Rightarrow M =_{\beta\mu} N$.

Py [50] has shown that $\to_{\beta\mu}$ is confluent. Therefore the convertibility relation $=_{\beta\mu}$ determined by $\to_{\beta\mu}$ is consistent in the usual sense that distinct normal forms are not equated.

2 $\lambda\mu$-models and term interpretation

As is the case for the λ-calculus, in order to provide a semantics to the untyped $\lambda\mu$-calculus we need to look for a domain D and a mapping $[\![\cdot]\!]^D$ such that $[\![M]\!]^D e \in D$ for each term M, where e maps variables to terms and names to continuations. Since the interpretation of terms depends on the interpretation of names and commands, we need an auxiliary domain C and a mapping $[\![\cdot]\!]^C$ such that $e\alpha \in C$ for any name α, and $[\![\mathtt{c}]\!]^C e \in C$. The term interpretation is a *model of the theory* $\lambda\mu$ if $[\![M]\!]^D = [\![N]\!]^D$ whenever $M =_{\beta\mu} N$.

The semantics we consider here is due to Streicher and Reus [52], but for a minor variant explained below. The idea is to work in the category $\mathcal{N}_R$ of 'negated' domains of the shape $A \to R$, where R is a parameter for the domain of results. In such a category, continuations are directly modelled and treated as the fundamental concept, providing a semantics both to Felleisen's $\lambda\mathcal{C}$-calculus and to a variant of $\lambda\mu$ that has, next to the two sorts of term we consider here (terms and commands) also *continuation* terms.

Here we adapt that semantics to Parigot's original $\lambda\mu$. We rephrase the model definition in the setting of the normal categories of domains, obtaining something similar to Hindley-Longo 'syntactical models.' Our models are essentially a particular case of the definitions in [44, 36].

Definition 2.1 ($\lambda\mu$-Model) A triple $\mathcal{M} = (R, D, C)$ is a $\lambda\mu$-model in a category of domains $\mathcal{D}$ if $R \in \mathcal{D}$ is a fixed domain of *results* and D and C (called domains of *denotations* and of *continuations*, respectively) are solutions in $\mathcal{D}$ of the equations:

$$\begin{cases} D &= C \to R \\ C &= D \times C \end{cases}$$

In the terminology of [52], elements of D are *denotations*, while those of C are *continuations*. We refer to the above equations as the *continuation domain equations*. We let k range over C, and d over D.

Remark 2.2 If (R, D, C) is a $\lambda\mu$-model then C is (isomorphic to) the infinite product $D \times D \times D \times \cdots$. On the other hand, as observed in [52] §3.1, we also have:

$$D \;\simeq\; C \to R \;\simeq\; (D \times C) \to R \;\simeq\; D \to (C \to R) \;\simeq\; D \to D.$$

since categories of domains are cartesian closed. Therefore, a $\lambda\mu$-model as defined in Def. 2.1 is an extensional λ-model.

Definition 2.3 (Term Interpretation) Let $\mathcal{M} = (R, D, C)$ be a $\lambda\mu$-model.

i) We define Env $= (\text{Var} \to D) \times (\text{Name} \to C)$ and call elements of Env *environments*; We write $e(x) = e_1(x)$ and $e(\alpha) = e_2(\alpha)$ for $e = \langle e_1, e_2 \rangle \in$ Env, and $\text{Env}_{\mathcal{M}}$ for the set of environments interpreting variables and names into $\mathcal{M}$.

ii) We define an *environment update* as:

$$e[x\mapsto d]\,y \;=\; \begin{cases} d & (x=y)\\ e\,y & (\text{otherwise})\end{cases}$$
$$e[\alpha\mapsto k]\,\beta \;=\; \begin{cases} k & (\alpha=\beta)\\ e\,\beta & (\text{otherwise})\end{cases}$$

iii) The *interpretation mappings* $[\![\cdot]\!]^D_{\mathcal{M}} : \textsc{Trm} \to \textsc{Env} \to D$ and $[\![\cdot]\!]^C_{\mathcal{M}} : \textsc{Cmd} \to \textsc{Env} \to C$, written $[\![\cdot]\!]^D$ and $[\![\cdot]\!]^C$ when $\mathcal{M}$ is understood, are mutually defined by the equations:

$$\begin{array}{rcll}
[\![x]\!]^D\, e\, k &=& e\ x\ k & \\
[\![\lambda x.M]\!]^D\, e\, k &=& [\![M]\!]^D\ e[x\mapsto d]\ k' & (k=\langle d,k'\rangle)\\
[\![MN]\!]^D\, e\, k &=& [\![M]\!]^D\, e\, \langle [\![N]\!]^D e, k\rangle & \\
[\![\mu\alpha.\mathsf{c}]\!]^D\, e\, k &=& d\ k' & (\langle d,k'\rangle = [\![\mathsf{c}]\!]^C\, e[\alpha\mapsto k])\\
[\![[\alpha]M]\!]^C\, e &=& \langle [\![M]\!]^D e, e\,\alpha\rangle &
\end{array}$$

This definition has a strong similarity with Bierman's interpretation of $\lambda\mu$ [19]; however, he considers a *typed* version. In the second equation of the definition of $[\![\cdot]\!]^D$, the assumption $k=\langle d,k'\rangle$ is not restrictive: in particular, if $k=\bot_C=\langle \bot_D,\bot_C\rangle$, then $d=\bot_D$ and $k'=k=\bot_C$.

The last two equations differ from those in [19] and [52] since there the interpretation of a command is a result:

$$[\![[\alpha]M]\!]^R_*\, e \;=\; [\![M]\!]^D_*\, e\,(e\,\alpha)$$

and consequently

$$[\![\mu\alpha.\mathsf{c}]\!]^D_*\, e\, k \;=\; [\![\mathsf{c}]\!]^R_*\, e[\alpha\mapsto k],$$

writing $[\![\cdot]\!]^A_*$ for the resulting interpretation maps. This is not an essential difference however: let $e'=e[\alpha\mapsto k]$, then

$$[\![\mu\alpha.[\beta]M]\!]^D_*\, ek \;=\; [\![[\beta]M]\!]^R_*\, e' \;=\; [\![M]\!]^D\, e'(e'\,\beta).$$

On the other hand, by Definition 2.3:

$$[\![[\beta]M]\!]^C\, e' \;=\; \langle [\![M]\!]^D\, e', (e'\,\beta)\rangle,$$

so

$$[\![\mu\alpha.[\beta]M]\!]^D\, e\, k \;=\; [\![M]\!]^D\ e'\ (e'\,\beta).$$

Therefore we can show $[\![\mu\alpha.[\beta]M]\!]^D_* = [\![\mu\alpha.[\beta]M]\!]^D$ by induction.

The motivation for interpreting commands into continuations instead of results is that the latter are elements of the parametric domain R; hence in the system of Sect. 4 results do not have significant types. On the other hand, by choosing our interpretation of commands we get explicit typing of commands with continuation types. Conceptually this could be justified by observing that commands are peculiar 'evaluation contexts' of the $\lambda\mu$-calculus, and continuations have been the understood as evaluation contexts since Felleisen's work. Technically, here we have just a variant of the treatment of, for example, [40], which system is based on the more standard interpretation.

Below, we fix a $\lambda\mu$-model $\mathcal{M}$, and we shall write $[\![\cdot]\!]^{\mathcal{M}}$ or simply $[\![\cdot]\!]$ by omitting the superscripts C and D whenever clear from the context.

Lemma 2.4 i) If $x\notin fv(M)$, then $[\![M]\!]\,e = [\![M]\!]\,e[x\mapsto d]$, for all $d\in D$.

ii) If $\alpha\notin fn(M)$, then $[\![M]\!]\,e = [\![M]\!]\,e[\alpha\mapsto k]$, for all $k\in C$.

Proof Easy. □

We now establish the relation between the various kinds of substitution and the interpretation.

Lemma 2.5
$$\begin{array}{rcll} [\![M[N/x]]\!]\, e\, k &=& [\![M]\!]\, e[x\mapsto [\![N]\!]e]\; k & \\ [\![T[\alpha/\beta]]\!]\, e\, k &=& [\![T]\!]\, e[\beta\mapsto e\alpha]\; k & \text{where } T\in \textsc{Trm}\cup\textsc{Cmd} \\ [\![M[\alpha\Leftarrow N]]\!]\, e\, k &=& [\![M]\!]\, e[\alpha\mapsto\langle [\![N]\!]e, e\alpha\rangle]\; k & \end{array}$$

Proof By induction on the definition of (structural) substitution. The only non-trivial case is when $M\equiv\mu\beta.[\alpha]L$ with $\beta\neq\alpha$, so that $(\mu\beta.[\alpha]L)\,[\alpha\Leftarrow N]\equiv\mu\beta.[\alpha]L\,[\alpha\Leftarrow N]N$. By unravelling definitions we have:

$$[\![\mu\beta.[\alpha]L\,[\alpha\Leftarrow N]N]\!]\, e\, k \;=\; d'\,k'$$

where

$$\begin{array}{rcl} \langle d', k'\rangle &=& [\![[\alpha]L\,[\alpha\Leftarrow N]N]\!]\; e[\beta\mapsto k] \\ &=& \langle [\![L\,[\alpha\Leftarrow N]N]\!]\; e[\beta\mapsto k], e[\beta\mapsto k]\,\alpha\rangle, \end{array}$$

observing that $e[\beta\mapsto k]\,\alpha = e\alpha$, since $\beta\neq\alpha$. Then:

$$\begin{array}{rcl} [\![\mu\beta.[\alpha]L\,[\alpha\Leftarrow N]N]\!]\, e\, k &=& [\![L\,[\alpha\Leftarrow N]N]\!]\; e[\beta\mapsto k]\;\; (e\alpha) \\ &=& [\![L\,[\alpha\Leftarrow N]]\!]\; e[\beta\mapsto k]\;\; \langle [\![N]\!]\; e[\beta\mapsto k], e\alpha\rangle \\ &=& [\![L]\!]\; e[\beta\mapsto k,\alpha\mapsto\langle [\![N]\!]e, e\alpha\rangle]\;\; \langle [\![N]\!]e, e\alpha\rangle \end{array}$$

where the last equation follows by induction and the fact that we can assume that $\beta\notin fv(N)$, so that $[\![N]\!]\; e[\beta\mapsto k] = [\![N]\!]e$. Let $e' = e[\alpha\mapsto\langle [\![N]\!]e, e\alpha\rangle]$, then:

$$[\![L]\!]\; e[\beta\mapsto k,\alpha\mapsto\langle [\![N]\!]e, e\alpha\rangle]\;\; \langle [\![N]\!]e, e\alpha\rangle \;=\; [\![L]\!]\; e'[\beta\mapsto k]\;\; \langle [\![N]\!]e, e\alpha\rangle$$

and

$$\begin{array}{rcl} \langle [\![L]\!]\; e'[\beta\mapsto k], \langle [\![N]\!]e, e\alpha\rangle\rangle &=& \langle [\![L]\!]\; e'[\beta\mapsto k], e'\,\alpha\rangle \\ &=& \langle [\![L]\!]\; e'[\beta\mapsto k], e'[\beta\mapsto k]\,\alpha\rangle \\ &=& [\![[\alpha]L]\!]\; e'[\beta\mapsto k] \end{array}$$

which implies

$$\begin{array}{rcl} [\![\mu\beta.[\alpha]L]\!]\; e'\; k &=& [\![L]\!]\; e'[\beta\mapsto k]\;\; \langle [\![N]\!]e, e\alpha\rangle \\ &=& [\![\mu\beta.[\alpha]L\,[\alpha\Leftarrow N]N]\!]\; e\; k \end{array}$$ □

Since our interpretation in Def. 2.3 does not coincide exactly with the one of Streicher and Reus, we have to check that it actually models $\lambda\mu$ convertibility. We begin by stating the key fact about the semantics, *i.e.* that it satisfies the following 'swapping continuations' equation:[2]

Lemma 2.6 $[\![\mu\alpha.[\beta]M]\!]\; e\; k = [\![M]\!]\; e[\alpha\mapsto k]\;\; (e[\alpha\mapsto k]\,\beta)$.

Proof Since $[\![\mu\alpha.[\beta]M]\!]\; e\; k = d\,k'$, where $\langle d, k'\rangle = [\![[\beta]M]\!]\; e[\alpha\mapsto k] = \langle [\![M]\!]\; e[\alpha\mapsto k],$ $e[\alpha\mapsto k]\,\beta\rangle$. □

We are now in place to establish the soundness of the interpretation.

Theorem 2.7 (Soundness of $[\![\cdot]\!]$ with respect to $\lambda\mu$) *If $M =_{\beta\mu} N$ then $[\![M]\!] = [\![N]\!]$.*

[2] The equation is from [52], where it is actually: $[\![\mu\alpha.[\beta]M]\!]\; e\; k = [\![M]\!]\; e[\alpha\mapsto k]\;\; (e\,\beta)$, but this is certainly just a typo.

Proof By induction on the definition of $=_{\beta\mu}$; it suffices to check the axioms (β), (μ), and (ren):

$((\lambda x.M)N = M[N/x])$: $\begin{array}{rcll} [\![(\lambda x.M)N]\!]\, e\, k & \triangleq & [\![M]\!]\, e\, \langle [\![N]\!]\, e, k\rangle & \\ & = & [\![M]\!]\, e[x\mapsto [\![N]\!]\, e]\ k & \\ & = & [\![M[N/x]]\!]\, e\, k & \text{(Lem. 2.5)} \end{array}$

$((\mu\alpha.[\beta]M)N = \mu\alpha.([\beta]M)\,[\alpha \Leftarrow N])$: Notice that, by Barendregt's convention, we can assume that $\alpha \notin fn(N)$; let $e' = e[\alpha\mapsto\langle [\![N]\!]\, e, k\rangle]$, then:

$$\begin{array}{rcll} [\![(\mu\alpha.[\beta]M)N]\!]\, e\, k & \triangleq & [\![\mu\alpha.[\beta]M]\!]\, e\, \langle [\![N]\!]\, e, k\rangle & \\ & = & [\![M]\!]\, e'\ (e'\beta) & \text{(Lem. 2.6)} \\ & = & [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ (e'\,\beta) & \text{(Lem. 2.5)} \end{array}$$

Now if $\beta = \alpha$ we have:

$$\begin{array}{rcll} [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ (e'\,\beta) & = & [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ \langle [\![N]\!]\, e, k\rangle & (\beta = \alpha) \\ & = & [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ \langle [\![N]\!]\, e[\alpha\mapsto k], k\rangle & (\alpha \notin fn(N)) \\ & = & [\![M\,[\alpha \Leftarrow N]N]\!]\, e[\alpha\mapsto k]\ (e[\alpha\mapsto k]\,\alpha) & \\ & = & [\![\mu\alpha.[\alpha]M\,[\alpha \Leftarrow N]N]\!]\, e\, k & \text{(Lem. 2.6)} \\ & \triangleq & [\![\mu\alpha.([\alpha]M)\,[\alpha \Leftarrow N]]\!]\, e\, k & \end{array}$$

Otherwise, if $\beta \neq \alpha$ we have:

$$\begin{array}{rcll} [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ (e'\,\beta) & = & [\![M\,[\alpha \Leftarrow N]]\!]\, e[\alpha\mapsto k]\ (e[\alpha\mapsto k]\,\beta) & \\ & = & [\![\mu\alpha.[\beta]M\,[\alpha \Leftarrow N]]\!]\, e\, k & \text{(Lem. 2.6)} \\ & \triangleq & [\![\mu\alpha.([\beta]M)\,[\alpha \Leftarrow N]]\!]\, e\, k & \end{array}$$

$(\mu\psi.[\alpha]\mu\beta.[\gamma]M = \mu\psi.([\gamma]M)[\alpha/\beta])$: For any e' we have

$$\begin{array}{rcl} [\![[\alpha]\mu\beta.[\gamma]M]\!]\, e' & = & \langle [\![\mu\beta.[\gamma]M]\!]\, e', (e'\,\alpha)\rangle \\ & = & \langle \Phi_{e'}, (e'\,\alpha)\rangle \end{array}$$

where

$$\begin{array}{rcl} \Phi_{e'} & = & \lambda k \in C.\ [\![\mu\beta.[\gamma]M]\!]\, e'\ k \\ & = & \lambda k \in C.\ [\![M]\!]\, e'[\beta\mapsto k]\ (e'[\beta\mapsto k]\,\gamma). \end{array}$$

On the other hand, by definition of interpretation and by Lem. 2.5 we have:

$$\begin{array}{rcl} [\![([\gamma]M)[\alpha/\beta]]\!]\, e' & = & [\![[\gamma]M]\!]\, e'[\beta\mapsto e\,\alpha] \\ & = & \langle [\![M]\!]\, e'[\beta\mapsto e\,\alpha], (e'[\beta\mapsto e\,\alpha]\,\gamma)\rangle. \end{array}$$

Therefore, taking $e' = e[\psi\mapsto k]$, by Lem. 2.6 we conclude:

$$\begin{array}{rcl} [\![\mu\psi.[\alpha]\mu\beta.[\gamma]M]\!]\, e\, k & = & \Phi_{e'}(e'\,\alpha) \\ & = & [\![\mu\psi.([\gamma]M)[\alpha/\beta]]\!]\, e\, k. \end{array}$$

□

3 The filter domain

In this section we will build a $\lambda\mu$-model in the category of ω-algebraic lattices. The model is obtained in Sect. 3.1 by means of standard domain theoretic techniques, following the construction in [52]; we exploit the fact that compact points of any ω-algebraic lattice can be described by means of a suitable *intersection type theory* (recalled in Sect. 3.2), to get a description of the model as a filter-model in Sect. 3.3. This provides us with a semantically justified definition of intersection types (actually of three kinds, to describe the domains R, D and C,

respectively, that form the model) and of their pre-orders that we shall use in Sect. 4 for the type assignment system.

The treatment of Sect. 3.1 is introductory and can be skipped by readers who are familiar with domain theory, but for Prop. 3.1 and 3.2, which are referred to in the paper. A fuller treatment of these topics can be found for example in [3], Ch. 1-3 and 7. The developments in Sect. 3.2 and Sect. 3.3 are inspired by [17, 22] and [1]; in particular, we have used [27] in Sect. 3.3; we borrow the terminology of 'intersection type theory' from [18], where intersection type systems and filter models are treated in full detail in Part III.

3.1 A domain theoretic solution of continuation domain equations

Complete lattices are partial orders $(X, \sqsubseteq)$, closed under meet $\bigsqcap Z$ (greatest lower bound) and join $\bigsqcup Z$ (smallest upper bound) of arbitrary subsets $Z \subseteq X$. Observing that $\bigsqcap Z = \bigsqcup\{ x \in X \mid \forall z \in Z\,[x \sqsubseteq z]\}$ and $\bigsqcup Z = \bigsqcap\{ x \in X \mid \forall z \in Z\,[z \sqsubseteq x]\}$, we have that if X is closed under arbitrary meets (joins) it is likewise under arbitrary joins (meets). Furthermore, in X there exist $\bot = \bigsqcup\emptyset$ and $\top = \bigsqcap\emptyset$, which are the bottom and top elements of X with respect to $\sqsubseteq$, respectively.

A subset $Z \subseteq X$ is *directed* if for any finite subset $V \subseteq Z$ there exists $z \in Z$ which is an upper bound of V. In particular, directed subsets are always non-empty. An element $e \in X$ is *compact* if, whenever $e \sqsubseteq \bigsqcup Z$ for some directed $Z \subseteq X$, there exists $z \in Z$ such that $e \sqsubseteq z$; we write $\mathcal{K}(X)$ for the set of compact elements of X. For $x \in X$ we define $\mathcal{K}(x) = \{e \in \mathcal{K}(X) \mid e \sqsubseteq x\}$; since directed sets are non-empty, $\bot \in \mathcal{K}(X)$ and hence $\bot \in \mathcal{K}(x)$, for all $x \in X$. A complete lattice X is *algebraic* if $\mathcal{K}(x)$ is directed for any $x \in X$ and $x = \bigsqcup \mathcal{K}(x)$; X is *ω-algebraic* if it is algebraic and the subset $\mathcal{K}(X)$ is countable.

A function $f : X \to Y$ of ω-algebraic lattices is *Scott-continuous* if and only if it preserves directed sups, namely $f(\bigsqcup Z) = \bigsqcup_{z \in Z} f(z)$ whenever $Z \subseteq X$ is directed. By algebraicity, any continuous function with domain X is fully determined by its restriction to $\mathcal{K}(X)$, that is, given a monotonic function $g : \mathcal{K}(X) \to Y$ there exists a unique continuous function $\widehat{g} : X \to Y$ that coincides with g over $\mathcal{K}(X)$, namely $\widehat{g}(x) = \bigsqcup g(\mathcal{K}(x))$; $\widehat{g}$ is called the *continuous extension* of g. The category **ω-AlgL** has ω-algebraic lattices as objects and Scott-continuous maps as morphisms. As such **ω-AlgL** is a full subcategory of the category of domains, but not of the category of (complete) lattices, since morphisms do not preserve arbitrary joins. In this paper we use the word *domain* as synonym of ω-algebraic lattice.

If X, Y are domains, then the component-wise ordered cartesian product $X \times Y$ and the pointwise ordered set $[X \to Y]$ of Scott-continuous functions from X to Y are both domains. In particular, if $f, g \in [X \to Y]$ then the function $(f \sqcup g)(x) = f(x) \sqcup g(x)$ is the join of f and g, that are always *compatible* since they have an upper bound. If Z is an ω-algebraic lattice then $[X \times Y \to Z] \simeq [X \to [Y \to Z]]$ is a natural isomorphism, and therefore the category **ω-AlgL** is cartesian closed.

$\mathcal{K}(X \times Y) = \mathcal{K}(X) \times \mathcal{K}(Y)$; $\mathcal{K}[X \to Y]$ is the set of finite joins of *step functions* $(a \Rightarrow b)$ where $a \in \mathcal{K}(X)$, $b \in \mathcal{K}(Y)$, defined by

$$(a \Rightarrow b)(x) \;=\; \begin{cases} b & (\text{if } a \sqsubseteq x) \\ \bot & (\text{otherwise}) \end{cases}$$

An infinite sequence $(X_n)_{n \in \mathbb{N}}$ of domains is *projective* if for all n the continuous functions $e_n : X_n \to X_{n+1}$ and $p_n : X_{n+1} \to X_n$ exist, called *embedding-projection pairs*, that satisfy $p_n \circ e_n = \mathsf{id}_{X_n}$ and $e_n \circ p_n \leq \mathsf{id}_{X_{n+1}}$, where $\leq$ is the pointwise ordering and id_V is the identity function on V. The *inverse limit* of the projective chain $(X_n)_{n \in \mathbb{N}}$ is the set $X_\infty = \lim_{\leftarrow} X_n$ which is defined as the set of all vectors $\vec{x} \in \Pi_n X_n$ such that $x_i = p_i(x_{i+1})$ for all i, ordered component wise. Moreover,

for all n there exists an embedding-projection pair $e_{n,\infty} : X_n \to X_\infty$ and $p_{n,\infty} : X_\infty \to X_n$ such that $p_{n,\infty}(\vec{x}) = x_n$ for all $\vec{x} \in X_\infty$: for details see for example [3], Ch. 7.

Property 3.1 The inverse limit $X_\infty = \lim_{\leftarrow} X_n$ of a sequence $(X_n)_{n\in\mathbb{N}}$ of domains is itself a domain such that

$$\mathcal{K}(X_\infty) \;=\; \bigcup_n \{ e_{n,\infty}(x) \mid x \in \mathcal{K}(X_n) \}.$$

Proof That X_∞ is a domain follows by the fact that each X_n is, for all n, and that for all $x \in X_n$ there exists $\vec{x} = e_{n,\infty}(x)$ such that $x_n = p_{n,\infty}(\vec{x}) = x$.

Writing $\bigcup_n e_{n,\infty}\mathcal{K}(X_n)$ for the right-hand side of the above equation, consider a directed subset $Z \subseteq X_\infty$. For any n, if $x \in \mathcal{K}(X_n)$ then $e_{n,\infty}(x) \sqsubseteq \bigsqcup Z$ implies

$$x \;=\; p_{n,\infty} \circ e_{n,\infty}(x) \;\;\sqsubseteq\;\; p_{n,\infty}(\bigsqcup Z) \;=\; \bigsqcup p_{n,\infty}(Z)$$

by the fact that $(e_{n,\infty}, p_{n,\infty})$ is an embedding-projection pair, and the continuity of $p_{n,\infty}$. By assumption there exists $z \in Z$ such that $x \sqsubseteq p_{n,\infty}(z)$, and therefore

$$e_{n,\infty}(x) \;\sqsubseteq\; (e_{n,\infty} \circ p_{n,\infty})(z) \;\;\sqsubseteq\;\; z \in Z,$$

hence $e_{n,\infty}(x) \in \mathcal{K}(X_\infty)$, by the arbitrary choice of Z. This proves $\mathcal{K}(X_\infty) \supseteq \bigcup_n e_{n,\infty}\mathcal{K}(X_n)$.

To see the converse inclusion, take $\vec{x} \in \mathcal{K}(X_\infty)$. We claim that $x_n = p_{n,\infty}(\vec{x}) \in \mathcal{K}(X_n)$, for any n. Indeed, if $U \subseteq X_n$ is directed and such that $x_n \sqsubseteq \sqcup U$, consider the set

$$V \;=\; \{ \vec{y} \in X_\infty \mid \forall m \neq n\, [y_m = x_m \;\&\; \exists u \in U\, [\; y_n = u\,]\,] \}$$

Then V is directed and $\vec{x} \sqsubseteq \sqcup V$. From the hypothesis $\vec{x} \in \mathcal{K}(X_\infty)$ we know that there exists $\vec{y} \in V$ such that $\vec{x} \sqsubseteq \vec{y}$, and so there exists $u = y_n \in U$ such that $x_n \sqsubseteq u$, establishing the claim. From this and the first part of this proof it follows that

$$\{ e_{n,\infty}(p_{n,\infty}(\vec{x})) \}_{n\in\mathbb{N}} \;=\; \{ e_{n,\infty}(x_n) \}_{n\in\mathbb{N}} \;\subseteq\; \bigcup_n e_{n,\infty}\mathcal{K}(X_n)$$

is a chain of elements in $\mathcal{K}(X_\infty)$, and by construction $\vec{x} = \bigsqcup_n e_{n,\infty}(x_n)$; since $\vec{x} \in \mathcal{K}(X_\infty)$ we can conclude that $\vec{x} = e_{n_0,\infty}(x_{n_0})$ for some n_0, so that $\mathcal{K}(X_\infty) \subseteq \bigcup_n e_{n,\infty}\mathcal{K}(X_n)$ as desired. □

We now consider the construction in [52] for the particular case of ω-**AlgL**. Let R be some fixed domain, dubbed the domain of *results* (for the sake of solving the continuation domain equations in a non-trivial way it suffices to take $R = \{\bot, \top\}$ with $\bot \sqsubset \top$, the two-point lattice). Now define the following sequences of domains:

$$\begin{array}{lcl} C_0 & = & \{\bot\} \\ D_n & = & [C_n \to R] \\ C_{n+1} & = & D_n \times C_n \end{array}$$

where $\{\bot\}$ is the trivial lattice such that $\bot = \top$. Observe that $D_0 = [C_0 \to R] \simeq R$ and $D_0 \simeq D_0 \times \{\bot\} = C_1$ and so $D_1 = [C_1 \to R] \simeq [R \to R]$. By unravelling the definition of C_n and D_n we obtain:

$$C_n \;=\; [C_{n-1} \to R] \times [C_{n-2} \to R] \times \cdots \times [C_0 \to R] \times C_0.$$

In [52] Thm. 3.1, it is proved that these sequences are projective, so that $D = \lim_{\leftarrow} D_n$ and $C = \lim_{\leftarrow} C_n$ are the initial/final solution of the continuation domain equations such that $R \simeq D_0$. By Prop. 3.1 we know that, up to the embeddings of each D_n into D and of each C_n into C, the compact points of D and C are the union of the compacts of the D_n and C_n,

respectively:

$$\mathcal{K}(D) = \bigcup_n \mathcal{K}(D_n), \qquad \mathcal{K}(C) = \bigcup_n \mathcal{K}(C_n). \tag{1}$$

In particular, $\mathcal{K}(R) = \mathcal{K}(D_0) \subseteq \mathcal{K}(D)$. Since $C \simeq D \times C$, C can be seen as the infinite product $\Pi_n D = D \times D \cdots$; also, $\mathcal{K}(C)$ is a proper subset of the product $\Pi_n \mathcal{K}(D) = \mathcal{K}(D) \times \mathcal{K}(D) \times \cdots$.

Property 3.2 The compact points in $C = \lim_{\leftarrow} C_n$ are those infinite tuples in $\Pi_n \mathcal{K}(D)$ whose components are all equal to $\perp$ but for a finite number of cases:

$$\mathcal{K}(C) \;=\; \{\langle d_1, d_2, \ldots\rangle \in \Pi_n \mathcal{K}(D) \mid \exists i\, \forall j \geq i\, [d_j = \perp]\}.$$

Proof Let

$$K \;=\; \{\langle d_1, d_2, \ldots\rangle \in \Pi_n \mathcal{K}(D) \mid \exists i\, \forall j \geq i\, [d_j = \perp]\}$$

Since $C = \Pi_n D$ is pointwise ordered, we have $K \subseteq \mathcal{K}(C)$, so it suffices to show the inverse inclusion. Let $k = \langle d_1, d_2, \ldots\rangle \in \mathcal{K}(C)$ and assume, towards a contradiction, that there exist infinitely many components $\{d_i \mid i \in \mathbb{N}\}$ of k that are different from $\perp$. Set $k_j = \langle d_1, d_2, \ldots, d_j, \perp, \ldots\rangle \in K$ (the tuple definitely equal to $\perp$ after d_j, while previous components are the same as in k), then the set $\{k_j \mid j \in \mathbb{N}\}$ is directed (actually a chain) and $k = \bigsqcup_j k_j$; but $k \not\sqsubseteq k_j$ for any j, hence $k \notin \mathcal{K}(C)$, a contradiction. □

Another way to see this proposition is to observe that the C_n are finite products of the shape $[C_{n-1} \to R] \times [C_{n-2} \to R] \times \cdots \times [C_0 \to R] \times C_0$, where $C_0 = \{\perp\}$. Hence any tuple in C_n, and therefore in $\mathcal{K}(C_n)$, has the form $\langle d_1, \ldots, d_{n-2}, \perp\rangle$. Now the embedding of such a tuple into $C = \lim_{\leftarrow} C_n$ is the infinite tuple $\langle d_1, \ldots, d_{n-2}, \perp, \ldots, \perp, \ldots\rangle$ that is definitely $\perp$ after d_{n-2}, and we know that the images of compact points in the C_n's are exactly the elements of $\mathcal{K}(C)$.

3.2 Intersection type theories and the filter construction

Intersection types form the 'domain logic' of ω-algebraic lattices in the sense of [1]. This means that for each domain X in ω-**AlgL** there exists a countable language $\mathcal{L}_X$ of intersection types together with an appropriate pre-order $\leq_X$ such that $(\mathcal{L}_X, \leq_X)$ is the Lindenbaum algebra of the compact points $\mathcal{K}(X)$ of X, *i.e.* an axiomatic, and hence finitary presentation of the structure $\mathcal{K}^{op}(X) = (\mathcal{K}(X), \sqsubseteq^{op})$ where $\sqsubseteq^{op}$ is just the inverse of the partial order $\sqsubseteq$ of X.

To understand this, first observe that $\mathcal{K}(X)$ is closed under binary joins. Indeed, for any $e_1, e_2 \in \mathcal{K}(X)$ if $e_1 \sqcup e_2 = \bigsqcup\{e_1, e_2\} \sqsubseteq \bigsqcup Z$ for some directed $Z \subseteq X$ then $e_1, e_2 \sqsubseteq \bigsqcup Z$, which implies that there exist $z_1, z_2 \in Z$ such that $e_1 \sqsubseteq z_1$ and $e_2 \sqsubseteq z_2$. By directness of Z there exists some $z_3 \in Z$ such that $z_1, z_2 \sqsubseteq z_3$, hence $e_1 \sqcup e_2 \sqsubseteq z_1 \sqcup z_2 \sqsubseteq z_3$. Thereby the structure $(\mathcal{K}(X), \sqsubseteq)$ is a sup-semilattice (a poset closed under finite joins), hence its dual $\mathcal{K}^{op}(X)$ is an inf-semilattice (a poset closed under finite meets), whose meet operator $\sqcap^{op}$ coincides with the join $\sqcup$ over $\mathcal{K}(X)$.

By algebraicity X, is generated by $\mathcal{K}(X)$ in the sense that $(X, \sqsubseteq)$ is isomorphic to the poset $(\mathsf{Idl}(\mathcal{K}(X)), \subseteq)$, where $\mathsf{Idl}(\mathcal{K}(X))$, the set of ideals over $\mathcal{K}(X)$, consists of directed and downward closed subsets of $\mathcal{K}(X)$. It turns out that the compact elements of $\mathsf{Idl}(\mathcal{K}(X))$ are just the images $\downarrow e = \mathcal{K}(e)$ of the elements $e \in \mathcal{K}(X)$. Dually, $(X, \sqsubseteq)$ is isomorphic to the poset $(\mathsf{Filt}(\mathcal{K}^{op}(X)), \subseteq)$, where $\mathsf{Filt}(\mathcal{K}^{op}(X))$ is the set of *filters* over $\mathcal{K}^{op}(X)$, that are non-empty subsets of $\mathcal{K}(X)$ which are upward closed with respect to $\sqsubseteq^{op}$ and closed under $\sqcap^{op}$.

Therefore filters over $\mathcal{K}^{op}(X)$ give rise to the algebraic lattice $(\mathsf{Filt}(\mathcal{K}^{op}(X)), \subseteq)$, whose compact elements are $\uparrow^{op} e = \{e' \in \mathcal{K}(X) \mid e \sqsubseteq^{op} e'\}$, called the *principal filters*. In summary, we have

the isomorphisms in the category ω-**AlgL**:

$$X \;\simeq\; \mathsf{Idl}(\mathcal{K}(X)) \;\simeq\; \mathsf{Filt}(\mathcal{K}^{op}(X)).$$

The fact that $\mathcal{K}(X)$ is a countable set allows for a finitary (syntactic) presentation of $X \simeq \mathsf{Filt}(\mathcal{K}^{op}(X))$ itself by introducing a language of types denoting the elements of $\mathcal{K}(X)$ and axioms and rules defining a pre-order over types whose intended meaning is $\sqsubseteq^{op}$.

An *intersection type language* $\mathcal{L}$ is a set of expressions closed under the binary operation $\wedge$ and including the constant ω. A pre-order $\leq$ is defined over this set, making $\wedge$ into the meet and ω into the top element, as formally stated in the next definition.

Definition 3.3 (Intersection Type Language and Theory) *i*) A denumerable set of type expressions $\mathcal{L}$ is called an *intersection type language* if there exists a constant $\omega \in \mathcal{L}$ and $\mathcal{L}$ is closed under the binary operator $\sigma\wedge\tau$, called *type intersection*.

ii) An *intersection type theory* $\mathcal{T}$ *over* $\mathcal{L}$ (where $\mathcal{L}$ is an intersection type language) is an axiomatic presentation of a pre-order $\leq_{\mathcal{T}}$ over types in $\mathcal{L}$ validating the following axioms and rules:

$$\frac{}{\sigma\wedge\tau \leq_{\mathcal{T}} \sigma} \qquad \frac{}{\sigma\wedge\tau \leq_{\mathcal{T}} \tau} \qquad \frac{}{\sigma \leq_{\mathcal{T}} \omega} \qquad \frac{\rho \leq_{\mathcal{T}} \sigma \quad \rho \leq_{\mathcal{T}} \tau}{\rho \leq_{\mathcal{T}} \sigma\wedge\tau}$$

iii) We abbreviate $\sigma \leq_{\mathcal{T}} \tau \leq_{\mathcal{T}} \sigma$ by $\sigma \sim_{\mathcal{T}} \tau$ and write $[\sigma]_{\mathcal{T}}$ for the equivalence class of σ with respect to $\sim_{\mathcal{T}}$. The subscript $\mathcal{T}$ will be omitted when no ambiguity is possible.

The type $\sigma\wedge\tau$ is called an intersection type in the literature. The reason for this is that as a type of λ-terms it is interpreted as the intersection of the interpretations of σ and τ in set theoretic models of the λ-calculus. This is rather unfortunate in the present setting, where we shall speak of filters and of their intersections. To avoid confusion, we speak of 'type intersections' when we refer to expressions of the shape $\sigma\wedge\tau$, reserving the word 'intersection' to the set theoretic operation.

Given an intersection type theory $\mathcal{T}$ over a language $\mathcal{L}$ that axiomatises the pre-order $\leq_{\mathcal{T}}$, the quotient $\mathcal{L}/_{\leq_{\mathcal{T}}}$ is an inf-semilattice. We will now establish a sufficient condition for $\leq_{\mathcal{T}}$ to be isomorphic to $\mathcal{K}^{op}(X)$ for some $X \in \omega$-**AlgL**.

Lemma 3.4 Let $\mathcal{T}$ be an intersection type theory over $\mathcal{L}$ and $\leq_{\mathcal{T}}$ the relative pre-order. Let $(X,\sqsubseteq)$ be a domain and $\Theta : \mathcal{L} \to \mathcal{K}(X)$ an order-reversing surjective mapping, i.e. *such that for all $\sigma,\tau \in \mathcal{L}$: $\sigma \leq_{\mathcal{T}} \tau \;\Rightarrow\; \Theta(\tau) \sqsubseteq \Theta(\sigma)$. Then $\mathcal{L}/_{\leq_{\mathcal{T}}} \simeq \mathcal{K}^{op}(X)$ as inf-semi-lattices.*

Proof Let $\Theta' : \mathcal{L}/_{\leq_{\mathcal{T}}} \to \mathcal{K}^{op}(X)$ be defined by $\Theta'([\sigma]) = \Theta(\sigma)$. If $[\sigma] = [\tau]$, then $\sigma \sim_{\mathcal{T}} \tau$, so by assumption $\Theta(\sigma) \sqsupseteq \Theta(\tau) \sqsupseteq \Theta(\sigma)$, which implies $\Theta(\sigma) = \Theta(\tau)$. This implies that Θ' is well defined and that Θ preserves and reflects $\leq_{\mathcal{T}}$ with respect to $\sqsubseteq^{op}$, and that Θ' is a bijection, since Θ is surjective. Finally, $\Theta'([\sigma\wedge\tau]) = \Theta(\sigma\wedge\tau) = \Theta(\sigma) \sqcup \Theta(\tau) = \Theta(\sigma) \sqcap^{op} \Theta(\tau)$. In particular, $\Theta'([\omega]) = \Theta(\omega) = \bot$. □

Under the hypotheses of the last lemma we have $X \simeq \mathsf{Filt}(\mathcal{K}^{op}(X)) \simeq \mathsf{Filt}(\mathcal{L}/_{\leq_{\mathcal{T}}})$; nonetheless, we consider the more convenient isomorphism of X with the set of filters over the pre-order $(\mathcal{L},\leq_{\mathcal{T}})$, which we call *formal filters*.

Definition 3.5 (Formal Filters) *i*) A *formal filter* with respect to an intersection type theory $\mathcal{T}$ over $\mathcal{L}$ is a subset $f \subseteq \mathcal{L}$ such that:

$$\frac{}{\omega \in f} \qquad \frac{\sigma \in f \quad \sigma \leq_{\mathcal{T}} \tau}{\tau \in f} \qquad \frac{\sigma \in f \quad \tau \in f}{\sigma\wedge\tau \in f}$$

$\mathcal{F}(\mathcal{T})$ is the set of formal filters induced by the theory and we let f, g range over $\mathcal{F}(\mathcal{T})$.

ii) The filter $\uparrow_{\mathcal{T}} \sigma = \{\tau \in \mathcal{L} \mid \sigma \leq_{\mathcal{T}} \tau\}$ is called *principal* and we write $\mathcal{F}_p(\mathcal{T})$ for the set of principal filters.

We recall some properties of formal filters and of the poset $(\mathcal{F}(\mathcal{T}), \subseteq)$. Since these are easily established or well known from the literature, we just state them or provide short arguments. The following is a list of some useful facts that follow immediately by definition.

Fact 3.6 *i*) $\sigma \leq_{\mathcal{T}} \tau$ if and only if $\uparrow_{\mathcal{T}} \tau \subseteq \uparrow_{\mathcal{T}} \sigma$.

ii) $\uparrow_{\mathcal{T}} \sigma \sqcup \uparrow_{\mathcal{T}} \tau = \uparrow_{\mathcal{T}} \sigma \wedge \tau$.

iii) If $\sigma \in f$, then $\uparrow_{\mathcal{T}} \sigma \subseteq f$.

iv) $f = \bigcup_{\sigma \in f} \uparrow_{\mathcal{T}} \sigma$.

v) For any $\mathcal{G} \subseteq \mathcal{F}(\mathcal{T})$, $\bigcap \mathcal{G} \in \mathcal{F}(\mathcal{T})$.

From Fact 3.6 (*v*) follows that the poset $(\mathcal{F}(\mathcal{T}), \subseteq)$ is a complete lattice, with set-theoretic intersection as (arbitrary) meet. Notice that, for $\mathcal{G} \subseteq \mathcal{F}(\mathcal{T})$, the join

$$\bigsqcup \mathcal{G} \;=\; \bigcap \{f \in \mathcal{F}(\mathcal{T}) \mid \forall g \in \mathcal{G}\, [g \subseteq f]\}$$

includes $\bigcup \mathcal{G}$ but does not coincide with it in general, since $\bigcup \mathcal{G}$ is not necessarily closed under $\wedge$. However, since $\bigcup \mathcal{G}$ is upper-closed with respect to $\leq_{\mathcal{T}}$, to get an explicit characterisation of $\bigsqcup \mathcal{G}$ it is enough to close $\bigcup \mathcal{G}$ under finite type intersections:

$$\bigsqcup \mathcal{G} \;=\; \{\sigma \mid \exists n, \sigma_i\ (i \in \underline{n}^{3})\, [\forall i \leq n\, [\sigma_i \in \bigcup \mathcal{G}]\ \&\ \sigma \sim_{\mathcal{T}} \sigma_1 \wedge \cdots \wedge \sigma_n]\}$$

On the other hand, if $\mathcal{G}$ is directed with respect to $\subseteq$, then $\sqcup \mathcal{G} = \bigcup \mathcal{G}$. In fact, if $\sigma_i \in \bigcup \mathcal{G}$ $(i \in \underline{n})$ we have that $\uparrow_{\mathcal{T}} \sigma_i \subseteq g_i$, for certain $g_i \in \mathcal{G}$ $(i \in \underline{n})$, by Fact 3.6 (*iii*). By directness there exists $g' \in \mathcal{G}$ such that $g_1 \cup \cdots \cup g_n \subseteq g'$, and hence the same holds for $\uparrow_{\mathcal{T}} \sigma_1 \cup \cdots \cup \uparrow_{\mathcal{T}} \sigma_n$. Then $\sigma_i \in g'$ $(i \in \underline{n})$ and so $\sigma_1 \wedge \cdots \wedge \sigma_n \in g' \subseteq \bigcup \mathcal{G}$, since g' is a formal filter.

The next lemma is not referenced explicitly in the paper, but is used throughout the rest of this section; it can be considered folklore in the theory of filter λ-models.

Lemma 3.7 *$(\mathcal{F}(\mathcal{T}), \subseteq)$ is an ω-algebraic lattice with top $\uparrow_{\mathcal{T}} \omega$ and compacts $\mathcal{K}(\mathcal{F}(\mathcal{T})) = \mathcal{F}_p(\mathcal{T})$.*

Proof From the discussion above we know that $(\mathcal{F}(\mathcal{T}), \subseteq)$ is a complete lattice, so it remains to show that it is ω-algebraic.

Let $\uparrow_{\mathcal{T}} \sigma \subseteq \bigsqcup \mathcal{G}$ for some directed $\mathcal{G} \subseteq \mathcal{F}(\mathcal{T})$. Then $\bigsqcup \mathcal{G} = \bigcup \mathcal{G}$ so that $\sigma \in \uparrow_{\mathcal{T}} \sigma \subseteq \bigcup \mathcal{G}$. Therefore there exists $g \in \mathcal{G}$ such that $\sigma \in g$ which implies $\uparrow_{\mathcal{T}} \sigma \subseteq g$ by 3.6 (*iii*). Hence $\mathcal{F}_p(\mathcal{T}) \subseteq \mathcal{K}(\mathcal{F}(\mathcal{T}))$. By 3.6 (*iv*) we have $f = \bigcup_{\sigma \in f} \uparrow_{\mathcal{T}} \sigma$ for any $f \in \mathcal{F}(\mathcal{T})$. Let $\uparrow_{\mathcal{T}} \sigma_i \subseteq f$ for certain $\sigma_i \in f$ $(i \in \underline{n})$; using 3.6 (*ii*) repeatedly we have $\uparrow_{\mathcal{T}} \sigma_1 \sqcup \cdots \sqcup \uparrow_{\mathcal{T}} \sigma_n = \uparrow_{\mathcal{T}} (\sigma_1 \wedge \cdots \wedge \sigma_n)$. On the other hand, $\sigma_1 \wedge \cdots \wedge \sigma_n \in f$ since f is a formal filter, and $\sigma_1 \wedge \cdots \wedge \sigma_n \leq_{\mathcal{T}} \sigma_i$ for all $i \in \underline{n}$, which by 3.6 (*i*) implies that $\uparrow_{\mathcal{T}} \sigma_i \subseteq \uparrow_{\mathcal{T}} (\sigma_1 \wedge \cdots \wedge \sigma_n)$, *i.e.* $\{\uparrow_{\mathcal{T}} \sigma \mid \sigma \in f\}$ is directed.

Now if $f \in \mathcal{K}(\mathcal{F}(\mathcal{T}))$, then $f \subseteq \uparrow_{\mathcal{T}} \sigma$ for some $\sigma \in f$; by 3.6 (*iii*) we conclude $f = \uparrow_{\mathcal{T}} \sigma$. Therefore $\mathcal{K}(\mathcal{F}(\mathcal{T})) \subseteq \mathcal{F}_p(\mathcal{T})$ and hence $\mathcal{K}(\mathcal{F}(\mathcal{T})) = \mathcal{F}_p(\mathcal{T})$ by the above. By this and 3.6 (*iv*) we conclude that $(\mathcal{F}(\mathcal{T}), \subseteq)$ is algebraic, and in fact is ω-algebraic because $\mathcal{L}$ is countable and the map $\sigma \mapsto \uparrow_{\mathcal{T}} \sigma$ from $\mathcal{L}$ to $\mathcal{F}_p(\mathcal{T})$ is obviously onto. □

Property 3.8 *Let $\mathcal{T}$ be an intersection type theory over $\mathcal{L}$, X a domain and $\Theta : \mathcal{L} \to \mathcal{K}(X)$ a mapping that satisfies the hypotheses of Lem. 3.4. Then $\mathcal{F}(\mathcal{T}) \simeq \mathsf{Filt}(\mathcal{K}^{op}(X)) \simeq X$.*

[3] We write $\underline{n}$ for the set $\{1, \ldots, n\}$ and $a_i \in V$ $(i \in \underline{n})$ for $a_1 \in V, \ldots, a_n \in V$.

Proof If $f \in \mathcal{F}(\mathcal{T})$ then by Lem. 3.6, $\{[\sigma] \mid \sigma \in f\}$ is a filter over $\mathcal{L}/_{\leq_\mathcal{T}}$; vice versa, if f is a filter over $\mathcal{L}/_{\leq_\mathcal{T}}$ then $\bigcup f = \{\sigma \mid [\sigma] \in f\}$ is a formal filter; therefore $\mathcal{F}(\mathcal{T}) \simeq \mathsf{Filt}(\mathcal{L}/_{\leq_\mathcal{T}})$. On the other hand, by Lem. 3.4 and the hypothesis, we have $\mathcal{L}/_{\leq_\mathcal{T}} \simeq \mathcal{K}^{op}(X)$ via the mapping $\Theta'([\sigma]) = \Theta(\sigma)$, so that the desired isomorphism $\mathcal{F}(\mathcal{T}) \simeq \mathsf{Filt}(\mathcal{K}^{op}(X))$ is given by

$$\lambda f. \bigsqcup\{\Theta'([\sigma]) \mid \sigma \in f\} \;=\; \lambda f. \bigsqcup\{\Theta(\sigma) \mid \sigma \in f\}. \qquad \square$$

Because of Prop. 3.8 and essentially following [17], we ignore the distinction between formal filters over a pre-order and filters over the ordered quotient and we shall work with the simpler formal filters, henceforth just called filters.

3.3 A filter domain solution to the continuation domain equations

Given an arbitrary domain R as in Def. 2.1, we fix the initial and final solution D, C of the continuation equations in the category ω-**AlgL**. For $A = R, D, C$, we will now define the languages $\mathcal{L}_A$ and the theories $\mathcal{T}_A$, inducing the pre-orders $\leq_{\mathcal{T}_A}$, which we write as $\leq_A$.

Definition 3.9 Take $R \in \omega$-**AlgL**, ordered by $\sqsubseteq_R$, with bottom $\bot$ and join $\sqcup$.

i) The intersection type language $\mathcal{L}_R$ is defined by the grammar:

$$\rho \;::=\; v_a \mid \omega \mid \rho \wedge \rho \quad (a \in \mathcal{K}(R))$$

We let v range over the set $\{v_a \mid a \in \mathcal{K}(R)\}$.

ii) $\mathcal{T}_R$ is the smallest intersection type theory axiomatising the pre-order $\leq_R$ such that (where $\sim_R = \leq_R \cap \leq_R^{op}$):

$$\frac{}{v_\bot \sim_R \omega} \qquad \frac{}{v_{a \sqcup b} \sim_R v_a \wedge v_b}$$

iii) The mapping $\Theta_R : \mathcal{L}_R \to \mathcal{K}(R)$ is defined by:

$$\begin{array}{rcl} \Theta_R(v_a) & = & a \\ \Theta_R(\omega) & = & \bot \\ \Theta_R(\rho_1 \wedge \rho_2) & = & \Theta_R(\rho_1) \sqcup \Theta_R(\rho_2) \end{array}$$

Observe that $\wedge$ is the meet with respect to $\leq_R$.

The following property, that states the relation between $\sqsubseteq_R$ and $\leq_R$, holds naturally:

Lemma 3.10 *i)* $v_a \leq_R v_b \Leftrightarrow b \sqsubseteq_R a$.

ii) $\rho \leq_R \rho' \Leftrightarrow \Theta_R(\rho') \sqsubseteq_R \Theta_R(\rho)$.

Proof In the following we remove all the subscripts R for notational simplicity.

i) If $v_a \leq v_b$ then either $b = \bot$, so that $\bot \sqsubseteq a$, or $v_a \sim v_a \wedge v_b$, since $\wedge$ is the meet with respect to $\leq$, which is an intersection type theory. By definition we have $v_{a \sqcup b} \sim v_a \wedge v_b$, so it follows that $v_a \sim v_{a \sqcup b}$, which implies $a = a \sqcup b$, so $b \sqsubseteq a$.

Vice versa, if $b \sqsubseteq a$ then $a = a \sqcup b$ and we have $v_a = v_{a \sqcup b} \sim v_a \wedge v_b$ from which we conclude $v_a \leq v_b$.

ii) When we, consistently with the theory $\leq$, identify $v_\bot$ and ω, then $\rho = v_{a_1} \wedge \cdots \wedge v_{a_h}$, for some $a_i \in \mathcal{K}(R)$ $(i \in \underline{h})$; notice that $v_{a_1} \wedge \cdots \wedge v_{a_h} \sim v_{a_1 \sqcup \cdots \sqcup a_h}$. Likewise, $\rho' = v_{b_1} \wedge \cdots \wedge v_{b_k}$, then by part (*i*) we have:

$$\rho \;\sim\; v_{a_1 \sqcup \cdots \sqcup a_h} \;\leq\; v_{b_1 \sqcup \cdots \sqcup b_k} \;\sim\; \rho' \;\Leftrightarrow\; b_1 \sqcup \cdots \sqcup b_k \;\sqsubseteq\; a_1 \sqcup \cdots \sqcup a_h$$

The result follows from $\Theta_R(\rho') = b_1 \sqcup \cdots \sqcup b_k$ and $\Theta_R(\rho) = a_1 \sqcup \cdots \sqcup a_h$. $\square$

The following corollary is the converse of Prop. 3.8.

Corollary 3.11 There exists an intersection type theory $\mathcal{T}_R$ such that $\mathcal{F}_R \simeq R$.

Proof Let $\mathcal{T}_R$ and Θ_R be defined as in Def. 3.9. Now Θ_R is surjective since $\Theta_R(v_a) = a$ for all $a \in \mathcal{K}(R)$ and, by Lem. 3.10 (*ii*), it satisfies the hypotheses of Lem. 3.4. We conclude that $\mathcal{F}_R \simeq R$ by Prop. 3.8. □

Remark 3.12 By Prop. 3.8, the isomorphism $\mathcal{F}_R \simeq R$ is given by the map $r \mapsto \bigsqcup\{\Theta_R(\rho) \mid \rho \in r\}$; as observed in the proof of Lem. 3.10 (*ii*), for any $\rho \in \mathcal{L}_R$ there exists $a \in \mathcal{K}(R)$ such that $\rho \sim_R v_a$, therefore the filter r is mapped isomorphically to $\bigsqcup\{\Theta_R(v_a) \mid v_a \in r\} = \bigsqcup\{a \mid v_a \in r\}$ by Def. 3.9. In case $r \in \mathcal{K}(\mathcal{F}_R)$ then by Lem. 3.7 and the previous remarks $r = \uparrow_R \rho = \uparrow_R v_a$ for some ρ and a, and its image in R is just a.

Definition 3.13 (Type theories $\mathcal{T}_D$ and $\mathcal{T}_C$) *i)* $\mathcal{L}_D$ and $\mathcal{L}_C$ are the intersection type languages defined by the grammar:

$$\begin{array}{lrcl} \mathcal{L}_D: & \delta & ::= & \rho \mid \kappa{\to}\rho \mid \omega \mid \delta{\wedge}\delta \quad (\rho \in \mathcal{L}_R) \\ \mathcal{L}_C: & \kappa & ::= & \delta{\times}\kappa \mid \omega \mid \kappa{\wedge}\kappa \end{array}$$

We let δ range over $\mathcal{L}_D$, and κ over $\mathcal{L}_C$, and σ, τ over $\mathcal{L}_D \cup \mathcal{L}_C$.

ii) We define $\wedge_{i\in I}\sigma_i$ through:

$$\begin{array}{rcl} \wedge_{i\in\emptyset}\sigma_i & \triangleq & \omega \\ \wedge_{i\in I}\sigma_i & \triangleq & \sigma_p \wedge (\wedge_{i\in I\setminus p}\sigma_i) \quad (p \in I) \end{array}$$

iii) The theories $\mathcal{T}_D$ and $\mathcal{T}_C$ are the least intersection type theories closed under the following axioms and rules, inducing the pre-orders $\leq_D$ and $\leq_C$ over $\mathcal{L}_D$ and $\mathcal{L}_C$ respectively:

$$\frac{\rho_1 \leq_R \rho_2}{\rho_1 \leq_D \rho_2} \qquad \frac{}{\omega \leq_D \omega{\to}\omega} \qquad \frac{}{v \leq_D \omega{\to}v} \qquad \frac{}{\omega{\to}v \leq_D v} \qquad \frac{}{\omega \leq_C \omega{\times}\omega}$$

$$\frac{}{(\kappa{\to}\delta_1){\wedge}(\kappa{\to}\delta_2) \leq_D \kappa{\to}(\delta_1{\wedge}\delta_2)} \qquad \frac{}{(\delta_1{\times}\kappa_1){\wedge}(\delta_2{\times}\kappa_2) \leq_C (\delta_1{\wedge}\delta_2){\times}(\kappa_1{\wedge}\kappa_2)}$$

$$\frac{\kappa_2 \leq_C \kappa_1 \quad \rho_1 \leq_R \rho_2}{\kappa_1{\to}\rho_1 \leq_D \kappa_2{\to}\rho_2} \qquad \frac{\delta_1 \leq_D \delta_2 \quad \kappa_1 \leq_C \kappa_2}{\delta_1{\times}\kappa_1 \leq_C \delta_2{\times}\kappa_2}$$

As usual, we define $\sigma \sim_A \tau$ if and only if $\sigma \leq_A \tau \leq_A \sigma$, for $A = C, D$.

It is straightforward to show that both $(\sigma{\wedge}\tau){\wedge}\rho \sim_A \sigma{\wedge}(\tau{\wedge}\rho)$ and $\sigma{\wedge}\tau \sim_A \tau{\wedge}\sigma$, so the type constructor $\wedge$ is associative and commutative, and we will write $\sigma{\wedge}\tau{\wedge}\rho$ rather than $(\sigma{\wedge}\tau){\wedge}\rho$. Thereby the definition of $\wedge_{i\in I}\sigma_i$ does not depend on the order in which p is chosen from I.

The pre-order $\leq_D$ is the usual one on arrow types in that the arrow is contra-variant in the first argument and co-variant in the second one. The pre-order $\leq_C$ on product types is co-variant in both arguments, and is the component-wise pre-order. As immediate consequence of Def. 3.13 we have that $\omega \sim_D \omega{\to}\omega$, $\omega \sim_C \omega{\times}\omega$, $(\kappa{\to}\delta_1){\wedge}(\kappa{\to}\delta_2) \sim_D \kappa{\to}(\delta_1{\wedge}\delta_2)$, and $(\delta_1{\times}\kappa_1){\wedge}(\delta_2{\times}\kappa_2) \sim_C (\delta_1{\wedge}\delta_2){\times}(\kappa_1{\wedge}\kappa_2)$.

The equation $\omega \sim_D \omega{\to}\omega$ together with $v \sim_D \omega{\to}v$ are typical of filter models that are extensional λ-models. The equation $\omega \sim_C \omega{\times}\omega$ allows for a finite representation of compact elements in C, that otherwise should be described by infinite expressions of the form $\delta_1{\times}\cdots{\times}\delta_k{\times}\omega{\times}\omega{\times}\cdots$ (see points (*i*) and (*ii*) of Lem. 3.14 below).

We have also

$$\omega \;\sim_D\; \omega{\to}\omega \;\leq_D\; \kappa{\to}\omega \;\leq_D\; \omega,$$

which implies that $\kappa{\to}\rho \sim_D \omega$ if and only if $\rho \sim_D \omega$.

Lemma 3.14 *i)* $\forall \kappa \in \mathcal{L}_C\ \exists \delta_i \in \mathcal{L}_D\ (i \in \underline{n})\,[\kappa \sim_C \delta_1 {\times} \cdots {\times} \delta_n {\times} \omega]$.
ii) $\delta_1 {\times} \cdots {\times} \delta_h {\times} \omega \leq_C \delta'_1 {\times} \cdots {\times} \delta'_k {\times} \omega \Leftrightarrow k \leq h$ & $\forall i \leq h\,[\delta_i \leq_D \delta'_i]$.
iii) $\forall \delta \in \mathcal{L}_C\ \exists n > 0, \kappa_i \in \mathcal{L}_C,\ \rho_i \in \mathcal{L}_R\ (i \in \underline{n})\,[\delta \sim_D \wedge_{\underline{n}}(\kappa_i {\to} \rho_i)]$.
iv) If I, J are finite and non-empty sets of indices and $\rho_i \not\sim_D \omega$ for all $i \in I$ then:

$$\wedge_{j\in J}(\kappa'_j {\to} \rho'_j) \leq_D \wedge_{i\in I}(\kappa_i {\to} \rho_i) \iff \forall i \in I\, \exists J_i \subseteq J\,[J_i \neq \emptyset \ \&\ \kappa_i \leq_C \wedge_{j\in J_i} \kappa_j \ \&\ \wedge_{j\in J_j} \rho_j \leq_R \rho_i]$$

Proof By induction on the structure of types and derivations in the theories $\mathcal{T}_C$ and $\mathcal{T}_D$. □

Note that the equivalence $\upsilon \sim_D \omega{\to}\upsilon$ is necessary to show Lem. 3.14 (*iii*).

The proofs of parts (*i*) and (*ii*) of the last lemma are straightforward; they should be compared however with Prop. 3.2, to understand how compact points in C are represented by types in $\mathcal{L}_C$. Parts (*iii*) and (*iv*) are characteristic of *extended abstract type structures* (EATS; see for example [3] Sect. 3.3); in particular the latter implies:

$$\wedge_{j\in J}(\kappa'_j {\to} \rho'_j) \leq_D \kappa {\to} \rho \ \Rightarrow\ \wedge\{\rho'_j \mid \kappa \leq_C \kappa'_j\} \leq_R \rho.$$

See [17] Lem. 2.4 (ii) or [3] Prop. 3.3.18.

The next step is to define the mappings $\Theta_D : \mathcal{L}_D \to \mathcal{K}(D)$ and $\Theta_C : \mathcal{L}_C \to \mathcal{K}(C)$ such that both satisfy the hypotheses of Lem. 3.4. In doing that we shall exploit Equation 1 in Sect. 3.1, by introducing a stratification of $\mathcal{L}_D$ and $\mathcal{L}_C$, thereby extending [27]. First we define the *rank* of a type in $\mathcal{L}_D$ and $\mathcal{L}_C$ inductively as follows:

$$\begin{array}{rcl} rk(\rho) = rk(\omega) & = & 0 \\ rk(\sigma{\wedge}\tau) & = & max\{rk(\sigma), rk(\tau)\} \\ rk(\delta{\times}\kappa) & = & max\{rk(\delta), rk(\kappa)\} + 1 \\ rk(\kappa{\to}\rho) & = & rk(\kappa) + 1. \end{array}$$

Then we define $\mathcal{L}_{A_n} = \{\sigma \in \mathcal{L}_A \mid rk(\sigma) \leq n\}$ for $A = D, C$. By this we have that if $n \leq m$ then $\mathcal{L}_{A_n} \subseteq \mathcal{L}_{A_m}$ and that $\mathcal{L}_A = \bigcup_n \mathcal{L}_{A_n}$.

Definition 3.15 The mappings $\Theta_{C_n} : \mathcal{L}_{C_n} \to \mathcal{K}(C_n)$ and $\Theta_{D_n} : \mathcal{L}_{D_n} \to \mathcal{K}(D_n)$ are defined by mutual induction through:

$$\begin{array}{rcl} \Theta_{C_0}(\kappa) & = & \bot \\ \Theta_{D_n}(\upsilon) & = & (\bot \Rightarrow \Theta_R(\upsilon)) = \lambda\!\!\lambda\_.\, \Theta_R(\upsilon) \\ \Theta_{D_n}(\kappa{\to}\rho) & = & (\Theta_{C_n}(\kappa) \Rightarrow \Theta_R(\rho)) \\ \Theta_{C_{n+1}}(\delta{\times}\kappa) & = & \langle \Theta_{D_n}(\delta), \Theta_{C_n}(\kappa) \rangle \end{array}$$

And, for $A_n = C_n, D_n$:

$$\begin{array}{rcl} \Theta_{A_n}(\omega) & = & \bot \\ \Theta_{A_n}(\sigma{\wedge}\tau) & = & \Theta_{A_n}(\sigma) \sqcup \Theta_{A_n}(\tau) \end{array}$$

The following lemma states that the mappings Θ_{C_n} and Θ_{D_n} are well defined, which is necessary since $\mathcal{L}_{C_n} \subseteq \mathcal{L}_{C_m}$ and $\mathcal{L}_{D_n} \subseteq \mathcal{L}_{D_m}$ when $n \leq m$.

Lemma 3.16 *For all $\kappa \in \mathcal{L}_C$ and $\delta \in \mathcal{L}_D$, if $rk(\kappa) \leq m$ and $rk(\delta) \leq n$ then*

$$\Theta_{C_m}(\kappa) = \Theta_{C_{rk(\kappa)}}(\kappa) \ \text{ and } \ \Theta_{D_n}(\delta) = \Theta_{D_{rk(\delta)}}(\delta).$$

Proof By easy induction over $m - rk(\kappa)$ and $n - rk(\delta)$, respectively. Consider the case of $\delta\times\kappa\to\rho$; then $rk(\delta\times\kappa\to\rho) = p+2$ where $p = max\{rk(\delta), rk(\kappa)\}$ and

$$\begin{array}{rcl}\Theta_{D_n}(\delta\times\kappa\to\rho) & = & (\Theta_{C_n}(\delta\times\kappa)\Rightarrow\Theta_R(\rho)) \\ & = & (\langle\Theta_{D_{n-1}}(\delta),\Theta_{C_{n-1}}(\kappa)\rangle\Rightarrow\Theta_R(\rho)).\end{array}$$

If $p+2\leq n$ then $rk(\delta)\leq p < p+1\leq n-1$ and $rk(\kappa)\leq p < p+1\leq n-1$, so that by induction: $\Theta_{D_{n-1}}(\delta) = \Theta_{D_{rk_{(\delta)}}}(\delta)$ and $\Theta_{C_{n-1}}(\kappa) = \Theta_{C_{rk_{(\kappa)}}}(\kappa)$. Then:

$$\begin{array}{rcl}(\langle\Theta_{D_{n-1}}(\delta),\Theta_{C_{n-1}}(\kappa)\rangle\Rightarrow\Theta_R(\rho)) & = & (\langle\Theta_{D_{rk_{(\delta)}}}(\delta),\Theta_{C_{rk_{(\kappa)}}}(\kappa)\rangle\Rightarrow\Theta_R(\rho)) \\ & = & \Theta_{D_{p+2}}(\delta\times\kappa\to\rho).\end{array}$$

□

For $A = D, C$, let $\leq_{A_n}$ be the pre-order $\leq_A$ restricted to $\mathcal{L}_{A_n}$.

Lemma 3.17 For every n, the mappings Θ_{C_n} and Θ_{D_n} are surjective and order reversing with respect to $\leq_{C_n}$ and $\leq_{D_n}$ respectively, i.e. *they satisfy the hypotheses of Lem. 3.4.*

Proof By induction on the definition of Θ_{C_n} and Θ_{D_n}.

$(n=0)$: The language $\mathcal{L}_{C_0}$ is generated by the constant ω and the connectives $\times$ and $\wedge$, so all types in $\mathcal{L}_{C_0}$ are equated by $\sim C_0$. Then the thesis holds for Θ_{C_0}, since $C_0 = \{\bot\}$. On the other hand, since the only way for a type in $\mathcal{L}_D$ to be of rank greater than 0 is to include an arrow, $\mathcal{L}_{D_0}$ is generated by the constants ω and v_a for $a\in\mathcal{K}(R)$ and the connective $\wedge$ so that $\mathcal{L}_{D_0} = \mathcal{L}_R$. Besides, the isomorphism $D_0\simeq R$ is given by the mapping $\lambda\!\!\lambda\_.r\mapsto r$, which is the continuous extension of the mapping $(\bot\Rightarrow a)\mapsto a$ from $\mathcal{K}(D_0)$ to $\mathcal{K}(R)$. Then $\Theta_{D_0}(a) = (\bot\Rightarrow a)\mapsto a = \Theta_R(v_a)$, where the last mapping is an isomorphism of ordered sets, hence order preserving and respecting. We conclude that $\Theta_{D_0} = \Theta_R$ up to the isomorphism $\mathcal{K}(D_0)\simeq\mathcal{K}(R)$, hence satisfies the hypotheses of Lem. 3.4 by Lem. 3.10 (*ii*).

$(n>0)$: If $\kappa\in\mathcal{L}_{C_n}$ then:

$$\Theta_{C_n}(\kappa) = \begin{cases}\langle\Theta_{D_{n-1}}(\delta),\Theta_{C_{n-1}}(\kappa)\rangle & (\text{if } \kappa = \delta'\times\kappa') \\ \Theta_{C_n}(\kappa_1)\sqcup\Theta_{C_n}(\kappa_2) & (\text{if } \kappa = \kappa_1\wedge\kappa_2) \\ \bot & (\text{if } \kappa = \omega).\end{cases}$$

For $\kappa = \delta'\times\kappa'$ we have $rk(\kappa)\leq n$ implies $rk(\delta'), rk(\kappa')\leq n-1$ by definition of rk; hence $\delta'\in\mathcal{L}_{D_{n-1}}$ and $\kappa'\in\mathcal{L}_{C_{n-1}}$. By induction, $\Theta_{D_{n-1}}:\mathcal{L}_{D_{n-1}}\to\mathcal{K}(D_{n-1})$ and $\Theta_{C_{n-1}}:\mathcal{L}_{C_{n-1}}\to\mathcal{K}(C_{n-1})$ are onto and order reversing. Since $\mathcal{K}(C_n) = \mathcal{K}(D_{n-1})\times\mathcal{K}(C_{n-1})$, by induction Θ_{C_n} is onto and order reversing. If $\kappa = \kappa_1\wedge\kappa_2$ then for any $\kappa_3\in\mathcal{L}_{C_n}$:

$$\begin{array}{rcll}\Theta_{C_n}(\kappa_1)\sqcup\Theta_{C_n}(\kappa_2)\sqsubseteq\Theta_{C_n}(\kappa_3) & \Leftrightarrow & \Theta_{C_n}(\kappa_i)\sqsubseteq\Theta_{C_n}(\kappa_3) & (i=1,2) \\ & \Leftrightarrow & \kappa_3\leq_{C_n}\kappa_i & (\text{by a subordinate induction on } \kappa) \\ & \Leftrightarrow & \kappa_3\leq_{C_n}\kappa_1\wedge\kappa_2. & \end{array}$$

Finally, the case $\kappa = \omega$ is obvious as $\bot = \langle\bot,\bot\rangle$ is the bottom in $\mathcal{K}(C_n)$, while $\omega\sim_{C_n}\omega\times\omega$ is the top in $(\mathcal{L}_{C_n},\leq_{C_n})$.

If $\delta\in\mathcal{L}_{D_n}$ then:

$$\Theta_{D_n}(\delta) = \begin{cases}(\bot\Rightarrow\Theta_R(\rho)) & (\text{if } \delta = \rho\in\mathcal{L}_R) \\ (\Theta_{C_n}(\kappa)\Rightarrow\Theta_R(\rho)) & (\text{if } \delta = \kappa\to\rho) \\ \Theta_{D_n}(\delta_1)\sqcup\Theta_{D_n}(\delta_2) & (\text{if } \delta = \delta_1\wedge\delta_2) \\ \bot & (\text{if } \delta = \omega)\end{cases}$$

By construction

$$\begin{array}{rcl}\mathcal{K}(D_n) & = & \mathcal{K}([C_n \to R]) \\ & = & \{\sqcup_{i\in I}(k_i \Rightarrow r_i) \mid I \text{ is finite } \& \, \forall i \in I\,[k_i \in \mathcal{K}(C_n) \,\&\, r_i \in \mathcal{K}(R)\,]\}.\end{array}$$

We know from the above that Θ_{C_n} is surjective (since both $\Theta_{D_{n-1}}$ and $\Theta_{C_{n-1}}$ are), while Θ_R is surjective by definition. Let $k_i = \Theta_{C_n}(\kappa_i)$ and $r_i = \Theta_R(\rho_i)$, then $\sqcup_{i\in I}(k_i \Rightarrow r_i) = \Theta_{D_n}(\wedge_{i\in I}\kappa_i \to \rho_i)$; since also $\Theta_{D_n}(\omega) = \bot = (\bot \Rightarrow \bot) = \Theta_{D_n}(\omega\to\omega)$, we conclude that Θ_{D_n} is surjective.

To see that Θ_{D_n} is order reversing, note that $(k \Rightarrow r) \sqsubseteq f$ for $f \in [C_n \to R]$ if and only if $r \sqsubseteq f(k)$, which is trivially the case if $r = \bot$. Since $\bot = \Theta_R(\omega)$ and also $\bot = (k \Rightarrow \bot) = \Theta_{D_n}(\kappa\to\omega)$ for any k and κ, while $\kappa\to\omega \sim D_n\omega \geq_{D_n} \delta$ for any $\delta \in \mathcal{L}_{D_n}$, the thesis trivially holds if $r = \bot$.

Suppose that $r \neq \bot$. Since $(\sqcup_{i\in I}(k_i \Rightarrow r_i))\,(x) \;=\; \sqcup_{j\in J} r_j$ for $J = \{j \in I \mid k_j \sqsubseteq x\}$, we have

$$\begin{array}{rcl}(k \Rightarrow r) \sqsubseteq \sqcup_{i\in I}(k_i \Rightarrow r_i) & \iff & r \sqsubseteq \sqcup_{i\in I}(k_i \Rightarrow r_i)(k) \\ & \iff & \exists J \subseteq I\,[r \sqsubseteq \sqcup_{j\in J} r_j \,\&\, \sqcup_{j\in J} k_j \sqsubseteq k]\end{array}$$

By subjectivity of Θ_{C_n} and Θ_R we know that there exist κ,ρ, such that $\Theta_{C_n}(\kappa) = k$ and $\Theta_{C_n}(\kappa_i) = k_i$, and κ_i,ρ_i such that $\Theta_R(\rho) = r, \Theta_R(\rho_i) = r_i$, for every $i \in I$. Therefore,

$$\begin{array}{rcll}\Theta_{D_n}(\kappa\to\rho) & = & (\Theta_{C_n}(\kappa) \Rightarrow \Theta_R(\rho)) \sqsubseteq \sqcup_{i\in I}(\Theta_{C_n}(\kappa_i) \Rightarrow \Theta_R(\rho_i)) & \\ & \iff & \exists J \subseteq I\,[\Theta_R(\rho) \sqsubseteq \sqcup_{j\in J}\Theta_R(\rho_j) \,\&\, \sqcup_{j\in J}\Theta_{C_n}(\kappa_j) \sqsubseteq \Theta_{C_n}(\kappa)\,] & \\ & \iff & \exists J \subseteq I\,[\Theta_R(\rho) \sqsubseteq \Theta_R(\wedge_{j\in J}\rho_j) \,\&\, \Theta_{C_n}(\wedge_{j\in J}\kappa_j) \sqsubseteq \Theta_{C_n}(\kappa)\,] & \\ & \iff & \exists J \subseteq I\,[\wedge_{j\in J}\rho_j \leq_R \rho \,\&\, \kappa \leq_{C_n} \wedge_{j\in J}\rho_j\,] & \\ & \iff & \wedge_{i\in I}(\kappa_i\to\rho_i) \leq_{D_n} \kappa\to\rho & \text{(by 3.14 (}iv\text{))}\end{array}$$

where we use that both Θ_{C_n} (as proved above) and Θ_R (by Lem. 3.10 (*ii*)) are order reversing, and that 3.14 (*iv*) applies because $\Theta_R(\rho) \neq \bot$ if and only if $\rho \neq_R \omega$ by 3.10 (*ii*). The general case

$$\Theta_{D_n}(\wedge_{i\in I}\kappa_i\to\rho_i) \;=\; \sqcup_{i\in I}\Theta_{D_n}(\kappa_i\to\rho_i) \;\sqsubseteq\; \Theta_{D_n}(\wedge_{j\in J}\kappa'_j\to\rho'_j)$$

now follows, since this is equivalent to

$$\begin{array}{rcl}\Theta_{D_n}(\kappa_i\to\rho_i) & = & (\Theta_{C_n}(\kappa_i) \Rightarrow \Theta_R(\rho_i)) \\ & \sqsubseteq & \Theta_{D_n}(\wedge_{j\in J}\kappa'_j\to\rho'_j) \\ & = & \sqcup_{j\in J}(\Theta_{C_n}(\kappa'_j) \Rightarrow \Theta_R(\rho'_j))\end{array}$$

for all $i \in I$. □

Definition 3.18 The mappings $\Theta_D : \mathcal{L}_D \to \mathcal{K}(D)$ and $\Theta_C : \mathcal{L}_C \to \mathcal{K}(C)$ are defined by

$$\begin{array}{rcl}\Theta_D(\delta) & = & \Theta_{D_{rk_{(\delta)}}}(\delta) \\ \Theta_C(\kappa) & = & \Theta_{C_{rk_{(\kappa)}}}(\kappa)\end{array}$$

Remark 3.19 By Lem. 3.16 and of the definition of *rk*, $\Theta_D(\kappa\to\rho) = (\Theta_C(\kappa) \Rightarrow \Theta_R(\rho))$ and similarly $\Theta_C(\delta\times\kappa) = \langle\Theta_D(\delta), \Theta_C(\kappa)\rangle$. In general all the equations in Def. 3.15 concerning the mappings Θ_{A_n} do hold for the respective maps Θ_A.

Lemma 3.20 The mappings Θ_D and Θ_C are surjective and order reversing, i.e. *satisfy the hypotheses of Lem. 3.4.*

Proof First observe that if $n \geq rk(\delta)$ then $\Theta_D(\delta) = \Theta_{D_{rk_{(\delta)}}}(\delta) = \Theta_{D_n}(\delta)$ by Lem. 3.16, and similarly for Θ_C. Now if $d \in \mathcal{K}(D)$ then by Equation (1) we have $\mathcal{K}(D) = \bigcup_n \mathcal{K}(D_n)$, so that there exists n such that $d \in \mathcal{K}(D_n)$. By Lem. 3.17 Θ_{D_n} is surjective, hence there exists $\delta \in \mathcal{L}_{D_n}$ such

that $\Theta_{D_n}(\delta) = d$. Then $rk(\delta) \leq n$ and $\Theta_D(\delta) = \Theta_{D_n}(\delta) = d$, by the above remark. Hence Θ_D is surjective.

On the other hand, if $\delta_1 \leq_D \delta_2$ then $\delta_1 \leq_{D_n} \delta_2$ for any $n \geq max\{rk(\delta_1), rk(\delta_2)\}$; by Lem. 3.17 and the above remark we conclude that

$$\Theta_D(\delta_1) \;=\; \Theta_{D_n}(\delta_1) \;\sqsupseteq\; \Theta_{D_n}(\delta_2) \;=\; \Theta_D(\delta_1),$$

which establishes that Θ_D is order reversing. The proof concerning Θ_C is similar. □

Theorem 3.21 *For $A = R, D, C$, the filter domain $\mathcal{F}_A$ is isomorphic to A.*

Proof That $\mathcal{F}_R \simeq R$ is stated in Cor. 3.11. By Lem. 3.20 Θ_D and Θ_C satisfy Lem. 3.4, hence we conclude by Prop. 3.8. □

Thm. 3.21 implies that $(\mathcal{F}_R, \mathcal{F}_D, \mathcal{F}_C)$ is a $\lambda\mu$-model. However, it is a rather implicit description of the model on which we base the construction of the intersection type assignment system in the next section. To get a better picture relating term and type interpretation, below we will show how functional application and the operation of adding an element of $\mathcal{F}_D$ in front of a continuation in $\mathcal{F}_C$ are defined in this model; this provides us with a more explicit description of the isomorphisms relating $\mathcal{F}_D$ and $\mathcal{F}_C$.

In the following, we let d and k range over filters in $\mathcal{F}_D$ and $\mathcal{F}_C$, respectively; notice that above they were used for elements of C and D. Since no confusion is possible, and since a clear link exists between these concepts, we permit ourselves a little overloading in notation.

Definition 3.22 For $d \in \mathcal{F}_D$ and $k \in \mathcal{F}_C$ we define:

$$\begin{array}{rcl} d \cdot k & \triangleq & \uparrow_D \{\rho \in \mathcal{L}_R \mid \exists \kappa \to \rho \in d\,[\kappa \in k]\} \\ d :: k & \triangleq & \uparrow_C \{\wedge_{i\in I} \delta_i \times \kappa_i \in \mathcal{L}_C \mid \forall i \in I\,[\delta_i \in d \;\&\; \kappa_i \in k]\} \end{array}$$

The upward closure $\uparrow_D$ in the definition of $d \cdot k$ is redundant, since we can show that the set $\{\wedge_{i\in I} \delta_i \times \kappa_i \in \mathcal{L}_C \mid \forall i \in I\,[\delta_i \in d \;\&\; \kappa_i \in k]\}$ is a filter. We have added $\uparrow_D$ to simplify proofs; in fact any set of types $\uparrow A$ is clearly closed under $\sim$. A similar remark holds for $\uparrow_C$ in the definition of $d :: k$, where we have to include ω. Alternatively one could stipulate the usual convention that $\wedge_{i\in I} \delta_i \times \kappa_i$ is syntactically the same as ω when $I = \emptyset$.

Lemma 3.23 *$d \cdot k \in \mathcal{F}_R$ and $d :: k \in \mathcal{F}_C$, for any $d \in \mathcal{F}_D$ and $k \in \mathcal{F}_C$. Moreover, the mappings '$\cdot$' and '::' are continuous in both their arguments.*

Proof The proof that $d \cdot k$ is well defined and continuous is essentially the same as that with EATS (see for example [3] Sect. 3.3). The set $d :: k$ is a filter by definition. By definition unfolding we have that

$$d :: k \;=\; (\bigcup_{\delta\in d} \uparrow_D \delta) :: (\bigcup_{\kappa\in k} \uparrow_C \kappa) \;=\; \bigcup_{\delta\in d, \kappa\in k} \uparrow_D \delta :: \uparrow_C \kappa,$$

hence '::' is continuous. □

In the particular case of $[\mathcal{F}_C \to \mathcal{F}_R]$, step functions (see Sect. 3.1) take the form $(\uparrow_C \kappa \Rightarrow \uparrow_R \rho)$. Indeed for $k \in \mathcal{F}_C$ we have that $\uparrow_C \kappa \subseteq k$ if and only if $\kappa \in k$, so that we have:

$$\begin{array}{rcl} (\uparrow_C \kappa \Rightarrow \uparrow_R \rho)(k) & = & \left\{ \begin{array}{ll} \uparrow_R \rho & (\text{if } \kappa \in k) \\ \uparrow_R \omega & (\text{ otherwise}) \end{array} \right. \\ & = & \uparrow_D (\kappa \to \rho) \cdot k \end{array}$$

Thus arrow types represent step functions. Similarly, the product of domains $X \times Y$ ordered component-wise is a domain such that $\mathcal{K}(X \times Y) = \mathcal{K}(X) \times \mathcal{K}(Y)$. In case of $\mathcal{F}_D \times \mathcal{F}_C$ compact points are of the shape $\langle \uparrow_D \delta, \uparrow_C \kappa \rangle$, which corresponds to the filter $\uparrow_C \delta \times \kappa \in \mathcal{F}_C$. This justifies the following definition:

Definition 3.24 We define the following maps:

$$\begin{array}{lll} F : \mathcal{F}_D \to [\mathcal{F}_C \to \mathcal{F}_R] & F\,d\,k & = d \cdot k \\ G : [\mathcal{F}_C \to \mathcal{F}_R] \to \mathcal{F}_D & G\,f & = \uparrow_D \{ \wedge_{i \in I} \kappa_i \to \rho_i \in \mathcal{L}_D \mid \forall i \in I\, [\rho_i \in f(\uparrow \kappa_i)] \} \\ H : \mathcal{F}_C \to (\mathcal{F}_D \times \mathcal{F}_C) & H\,k & = \langle \{ \delta \in \mathcal{L}_D \mid \delta \times \kappa \in k \}, \{ \kappa \in \mathcal{L}_C \mid \delta \times \kappa \in k \} \rangle \\ K : (\mathcal{F}_D \times \mathcal{F}_C) \to \mathcal{F}_C & K\langle d, k \rangle & = d :: k \end{array}$$

Remark 3.25 As expected form the claim that step functions in $[\mathcal{F}_C \to \mathcal{F}_R]$ are represented by arrow types in $\mathcal{L}_D$, for any $\kappa \to \rho \in \mathcal{L}_D$ we have $G(\uparrow_C \kappa \Rightarrow \uparrow_R \rho) = \uparrow_D (\kappa \to \rho)$. Indeed, $\kappa \to \rho \leq_D \kappa' \to \rho'$ if and only if $\kappa' \leq_C \kappa$ and $\rho \leq_R \rho'$, *i.e.* $\uparrow_C \kappa \subseteq \uparrow_C \kappa'$ and $\uparrow_R \rho' \subseteq \uparrow_R \rho$, if and only if $\rho' \in \uparrow_R \rho = (\uparrow_C \kappa \Rightarrow \uparrow_R \rho)(\uparrow_C \kappa')$. Similarly, the type $\delta \times \kappa \in \mathcal{L}_C$ represents pairs in $\mathcal{F}_D \times \mathcal{F}_C$ via K, *i.e.* $K\langle \uparrow_D \delta, \uparrow_C \kappa \rangle = \uparrow_D \delta :: \uparrow_C \kappa = \uparrow_C (\delta \times \kappa)$.

When no ambiguity is possible, we will write $\uparrow \rho$ for $\uparrow_R \rho$, and similarly for $\uparrow_D$ and $\uparrow_C$.

Lemma 3.26 The functions F, G and H, K are well defined and monotonic with respect to subset inclusion.

Proof By Lem. 3.23, F and K are well defined and continuous, hence monotonic.

For all $f \in [\mathcal{F}_C \to \mathcal{F}_R]$, by definition the set $G\,f$ is a filter over $(\mathcal{L}_D, \leq_D)$; we check that G is monotonic. Observe that $\wedge_{i \in I} \kappa_i \to \rho_i \in G\,f$ if and only if $\sqcup_{i \in I} (\uparrow \kappa_i \Rightarrow \uparrow \rho_i) \sqsubseteq f$; on the other hand, if $f \sqsubseteq g$ then $\sqcup_{i \in I} (\uparrow \kappa_i \Rightarrow \uparrow \rho_i) \sqsubseteq f$ implies $\sqcup_{i \in I} (\uparrow \kappa_i \Rightarrow \uparrow \rho_i) \sqsubseteq g$, so $\wedge_{i \in I} \kappa_i \to \rho_i \in G\,f$ implies $\wedge_{i \in I} \kappa_i \to \rho_i \in G\,g$.

The function H is evidently monotonic with respect to $\subseteq$. We check that it is well defined, *i.e.* that both

$$\begin{array}{lll} d' & = & \{ \delta \in \mathcal{L}_D \mid \delta \times \kappa \in k \} \quad \text{and} \\ k' & = & \{ \kappa \in \mathcal{L}_C \mid \delta \times \kappa \in k \} \end{array}$$

are filters whenever k is one. Let δ_1, $\delta_2 \in d'$, then there exist κ_1, κ_2 such that $\delta_1 \times \kappa_1$, $\delta_2 \times \kappa_2 \in k$ (and hence κ_1, $\kappa_2 \in k'$). Since k is a filter, we have $\delta_1 \times \kappa_1 \wedge \delta_2 \times \kappa_2 \in k$; also, $\delta_1 \times \kappa_1 \wedge \delta_2 \times \kappa_2 \sim_C (\delta_1 \wedge \delta_2) \times (\kappa_1 \wedge \kappa_2)$ implies $(\delta_1 \wedge \delta_2) \times (\kappa_1 \wedge \kappa_2) \in k$, as k, being a filter, is closed under meets and $\sim_C$. We conclude that $\delta_1 \wedge \delta_2 \in d'$; similarly, we can reason that $\kappa_1 \wedge \kappa_2 \in k'$.

The same reasoning shows that both d' and k' are upward closed sets with respect to $\leq_D$ and $\leq_C$, respectively. □

We can show that the following isomorphisms exist:

Theorem 3.27 *$\mathcal{F}_D \simeq [\mathcal{F}_C \to \mathcal{F}_R]$ via F with inverse G, and $\mathcal{F}_C \simeq \mathcal{F}_D \times \mathcal{F}_C$ via H with inverse K.*

Proof Since any monotonic function of posets that is invertible is an isomorphism, by Lem. 3.26 it suffices to show that $G = F^{-1}$ and $K = H^{-1}$.

$$\begin{array}{lll} i)\ (F \circ G)\, f\, k & = & F(\uparrow \{ \wedge_{i \in I} \kappa_i \to \rho_i \in \mathcal{L}_D \mid \forall i \in I\, [\rho_i \in f(\uparrow \kappa_i)\,] \})\, k] \\ & = & \uparrow \{ \wedge_{i \in I} \kappa_i \to \rho_i \in \mathcal{L}_D \mid \forall i \in I\, [\rho_i \in f(\uparrow \kappa_i)\,] \cdot k] \\ & = & \uparrow \{ \rho \mid \exists \kappa \in k\, [\rho \in f(\uparrow \kappa)\,] = \bigcup_{\kappa \in k} f(\uparrow \kappa)\,] \\ & = & \sqcup_{\uparrow \kappa \subseteq k} f(\uparrow \kappa) \qquad \text{(since } \{ \uparrow \kappa \mid \kappa \in k \} \text{ is directed)} \\ & = & f(k) \qquad \text{(by continuity of } f) \end{array}$$

hence $(F \circ G)\, f = f$.

ii) $\begin{array}{rcl}(G \circ F)\, d &=& G(\lambda\!\!\lambda k \in \mathcal{F}_C .\, d \cdot k) \\ &=& \uparrow\{\wedge_{i\in I}\, \kappa_i{\to}\rho_i \mid \forall i \in I\, [\rho_i \in d \cdot \uparrow\kappa_i]\} \\ &=& \uparrow\{\wedge_{i\in I}\, \kappa_i{\to}\rho_i \mid \forall i \in I\, \exists \kappa'_i\, [\kappa_i \leq_C \kappa'_i \,\&\, \kappa'_i{\to}\rho_i \in d]\} \;=\; d\end{array}$

where $\lambda\!\!\lambda$ represents semantic abstraction. In the last equation, the inclusion $\supseteq$ is obvious, while the inclusion $\subseteq$ follows by the fact that if $\kappa_i \leq_C \kappa'_i$ then $\kappa'_i{\to}\rho_i \leq_D \kappa_i{\to}\rho_i$, hence $\kappa'_i{\to}\rho_i \in d$ implies $\kappa_i{\to}\rho_i \in d$ for all $i \in I$, which in turn implies that $\wedge_{i\in I}\, \kappa_i{\to}\rho_i \in d$.

iii) $\begin{array}{rcl}(H \circ K)\langle d, k\rangle &=& H(d :: k) \\ &=& \langle\{\delta \in \mathcal{L}_D \mid \delta{\times}\kappa \in d :: k\}, \{\kappa \in \mathcal{L}_C \mid \delta{\times}\kappa \in d :: k\}\rangle \;=\; \langle d, k\rangle\end{array}$

by observing that $\delta{\times}\kappa \in d :: k$ if and only if $\delta \in d$ and $\kappa \in k$, and that if $\kappa' \in \uparrow\omega \subseteq d :: k$ then $\kappa' \sim_C \omega{\times}\omega$ and obviously $\omega \in d$ and $\omega \in k$.

iv) $\begin{array}{rcll}(K \circ H)\, k &=& \{\delta \in \mathcal{L}_D \mid \delta{\times}\kappa \in k\} :: \{\kappa \in \mathcal{L}_C \mid \delta{\times}\kappa \in k\} & \\ &=& \uparrow\{\wedge_{i\in I}\, \delta_i{\times}\kappa_i \mid \forall i \in I\, \exists \delta'_i, \kappa'_i\, [d_i{\times}\kappa'_i, \delta'_i{\times}\kappa_i \in k]\} & \\ &=& k & (\text{since } k \in \mathcal{F}_C)\end{array}$ □

Remark 3.28 As observed in Remark 2.2, a $\lambda\mu$-model is an extensional λ-model. Thm. 3.1 in [52] states that the initial/final solution of the continuation domain equations is isomorphic to the domain $R_\infty \simeq [R_\infty \to R_\infty]$, *i.e.* Scott's D_∞ λ-model obtained as inverse limit of a chain where $D_0 = R$.

To see this from the point of view of the intersection type theory, consider the extension $\mathcal{L}_\lambda = \cdots \mid \delta{\to}\delta$ of $\mathcal{L}_D$. Let $\mathcal{T}_\lambda$ be the theory obtained by adding to $\mathcal{T}_D$ the equation $\delta{\times}\kappa{\to}\rho = \delta{\to}\kappa{\to}\rho$. Then in the intersection type theory $\mathcal{T}_\lambda$, the following rules are derivable:

$$\frac{}{(\delta{\to}\delta_1){\wedge}(\delta{\to}\delta_2) \leq_\lambda \delta{\to}(\delta_1{\wedge}\delta_2)} \qquad \frac{\delta'_1 \leq_\lambda \delta_1 \quad \delta_2 \leq_\lambda \delta'_2}{\delta_1{\to}\delta_2 \leq_\lambda \delta'_1{\to}\delta'_2}$$

By this, $\mathcal{T}_\lambda$ is a *natural equated* intersection type theory in terms of [2], and hence $\mathcal{F}^\lambda \simeq [\mathcal{F}^\lambda \to \mathcal{F}^\lambda]$ where $\mathcal{F}^\lambda$ is the set of filters generated by the pre-order $\leq_\lambda$ (see [2], Cor. 28 (4)).

4 An intersection type system

Let $\mathcal{M} = (R, D, C)$ be a $\lambda\mu$-model, where D, C are initial solutions of the continuation domain equations (we say then that $\mathcal{M}$ is *initial*). In this section, using the fact that $\mathcal{M}$ is isomorphic to the filter model $\mathcal{F} = (\mathcal{F}_R, \mathcal{F}_D, \mathcal{F}_C)$, as established by Thm. 3.21 and 3.27, we will define a type assignment system such that the statement $M : \delta$ (or $\mathtt{c} : \kappa$) is derivable, under appropriate assumptions about the variables and names in it, if and only if $[\![M]\!]^{\mathcal{F}} e \in \uparrow_D \delta$ (or $[\![\mathtt{c}]\!]^{\mathcal{F}} e \in \uparrow_C \kappa$) for all environments e respecting those assumptions.

Thereby an interpretation of types can be defined such that $[\![\sigma]\!]^{\mathcal{F}} = \uparrow_A \sigma$ for $A = D, C$. Since filters are upward closed sets of types, we have that $[\![T]\!]^{\mathcal{F}} e \in \uparrow_A \sigma$ if and only if $\sigma \in [\![T]\!]^{\mathcal{F}} e$, and we obtain that the denotation of a term/command is just the set of types that can be inferred for it in the assignment system.

4.1 Type assignment

We now give some preliminary definitions for our type system.

Definition 4.1 (Bases, Name Contexts, and Judgements) *i)* A *basis* is a finite mapping from term variables to types in $\mathcal{T}_D$, written as a finite set $\Gamma = \{x_1{:}\delta_1, \ldots, x_n{:}\delta_n\}$ where the term

variables x_i are pairwise distinct.

ii) A *name context* (or *context*) is a finite mapping from names to types in $\mathcal{T}_C$, written as a finite set $\Delta = \{\alpha_1{:}\kappa_1, \ldots, \alpha_m{:}\kappa_m\}$ where the continuation variables α_i are pairwise distinct.

iii) We write $\Gamma, x{:}\delta$ for the basis $\Gamma \cup \{x{:}\delta\}$, and assume that either x does not occur in Γ or $x{:}\delta \in \Gamma$, and similarly for $\alpha{:}\kappa, \Delta$.

iv) We write $\Gamma\backslash x$ for $\Gamma\backslash\{x{:}\Gamma(x)\}$ and $\Delta\backslash\alpha$ for $\Delta\backslash\{\alpha{:}\Delta(\alpha)\}$.

v) Let Γ be a basis and Δ a name context. We define

$$\begin{array}{rcl} dom(\Gamma) & \triangleq & \{x \mid \exists\delta\,[x{:}\delta \in \Gamma]\} \\ dom(\Delta) & \triangleq & \{\alpha \mid \exists\kappa\,[\alpha{:}\kappa \in \Delta]\} \end{array}$$

and write $x \notin \Gamma$ ($\alpha \notin \Delta$) if $x \notin dom(\Gamma)$ ($\alpha \notin dom(\Delta)$).

vi) A *judgement* is an expression of the form $\Gamma \vdash M : \delta \mid \Delta$ or $\Gamma \vdash \mathtt{c} : \kappa \mid \Delta$ where Γ is a basis and Δ is a name context. M and $\mathtt{c}$ are the *subjects* and δ and κ the *predicates*.

We will occasionally allow ourselves some freedom when writing basis and contexts, and also consider $\Gamma, x{:}\omega$ a basis and $\alpha{:}\omega, \Delta$ a context.

Judgements are in appearance very similar to Parigot's (see Sect. 7), apart from the obvious difference in the language of types; in fact, there exists a relation between Parigot's system and the one presented here, which will be treated in detail in Sect. 7. Since bases and contexts are sets, the order in which variable and name assumptions are listed is immaterial.

We will occasionally treat basis and contexts as total functions by using the following notation:

$$\Gamma(x) \;=\; \begin{cases} \delta & (\text{if } x{:}\delta \in \Gamma) \\ \omega & (\text{otherwise}) \end{cases} \qquad \Delta(\alpha) \;=\; \begin{cases} \kappa & (\text{if } \alpha{:}\kappa \in \Delta) \\ \omega & (\text{otherwise}) \end{cases}$$

Notice that then $\Gamma\backslash x$ corresponds to the function update $\Gamma[x := \omega]$ and $\Delta\backslash\alpha$ to $\Delta[\alpha := \omega]$.

Definition 4.2 (Intersection type system for $\lambda\mu$) We define intersection type assignment for $\lambda\mu$ through the following sets of inference rules:

(Type rules):

$$(Ax): \frac{}{\Gamma, x{:}\delta \vdash x : \delta \mid \Delta} \qquad (Abs): \frac{\Gamma \vdash M : \kappa \to \rho \mid \Delta}{\Gamma\backslash x \vdash \lambda x.M : \delta \times \kappa \to \rho \mid \Delta}\ (\Gamma(x) = \delta)$$

$$(App): \frac{\Gamma \vdash M : \delta \times \kappa \to \rho \mid \Delta \quad \Gamma \vdash N : \delta \mid \Delta}{\Gamma \vdash MN : \kappa \to \rho \mid \Delta}$$

$$(Cmd): \frac{\Gamma \vdash M : \delta \mid \Delta}{\Gamma \vdash [\alpha]M : \delta \times \kappa \mid \Delta}\ (\Delta(\alpha) = \kappa) \qquad (\mu): \frac{\Gamma \vdash \mathtt{c} : (\kappa' \to \rho) \times \kappa' \mid \Delta}{\Gamma \vdash \mu\alpha.\mathtt{c} : \kappa \to \rho \mid \Delta\backslash\alpha}\ (\Delta(\alpha) = \kappa)$$

(Logical rules):

$$(\wedge): \frac{\Gamma \vdash T : \sigma \mid \Delta \quad \Gamma \vdash T : \tau \mid \Delta}{\Gamma \vdash T : \sigma \wedge \tau \mid \Delta} \qquad (\omega): \frac{}{\Gamma \vdash T : \omega \mid \Delta} \qquad (\leq): \frac{\Gamma \vdash T : \sigma \mid \Delta \quad \sigma \leq \tau}{\Gamma \vdash T : \tau \mid \Delta}$$

We will write $\Gamma \vdash T : \sigma \mid \Delta$ if there exists a derivation built using the above rules that has this judgement in the bottom line, and $\mathcal{D} :: \Gamma \vdash T : \sigma \mid \Delta$ if we want to name that derivation.

As mentioned above, we extend Barendregt's convention to judgements $\Gamma \vdash T : \sigma \mid \Delta$ by seeing the variables that occur in Γ and names in Δ as binding occurrences over T as well; in particular, we will assume that no variable in Γ and no name in Δ is bound in T.

To understand these rules we can think of types as properties of term denotations in the initial model $\mathcal{M} = (R, D, C)$. In particular, if $\sigma \in \mathcal{L}_A$ then σ denotes a subset $[\![\sigma]\!]^{\mathcal{M}} \subseteq A$, for $A = R, D, C$. The judgement $\Gamma \vdash T : \sigma \mid \Delta$ is then interpreted as the claim that $[\![T]\!]^{\mathcal{M}} e \in [\![\sigma]\!]^{\mathcal{M}}$

whenever $e\,x \in [\![\Gamma(x)]\!]$ and $e\,\alpha \in [\![\Delta(\alpha)]\!]$ for all x and α (the formal definitions will be given in Sect. 4.2).

The *logical* rules, which are familiar from intersection type systems for the standard λ-calculus, just state that types are sets: ω is the largest set which coincides with the domain of interpretation itself, the pre-order is subset inclusion, and $\sigma \wedge \tau$ is the set theoretic intersection of σ and τ. Note that the subject in the conclusion of a logical rule is the same as in the premises. Moreover, remark that we use here the term 'logical' in the sense of Abramsky's domain logic, not in the sense of (propositional) logic of any kind. In particular, intersection is *not* conjunction, both in systems for the λ-calculus and in the present one.

The *type rules* are syntax directed; they have been obtained from the equations in Def. 2.3 by representing the left-hand side of the equation in the conclusion and the right-hand side in the premises of the corresponding rule:

- (Abs): This rule corresponds to the equation $[\![\lambda x.M]\!]^D\, e\, \langle d, k\rangle = [\![M]\!]^D\, e[x \mapsto d]\, k$, where $[\![\cdot]\!]^D$ is short for $[\![\cdot]\!]^{\mathcal{M}}_{\mathcal{D}}$. It states that $\lambda x.M$ is a function of continuations $\langle d, k\rangle$, whose values are those of M where x is interpreted by d, and applied to continuation k. On the other hand, the arrow types from $\mathcal{L}_D$ represent properties of functions: a property of $\lambda x.M$ is then a type $\delta \times \kappa \to \rho$ (the conclusion of the rule) so that whenever $\delta \times \kappa$ is a property of $\langle d, k\rangle$, *i.e.* $d \in [\![\delta]\!]^{\mathcal{M}}$ and $k \in [\![\kappa]\!]^{\mathcal{M}}$, ρ is a property of the result. But since the result is $[\![M]\!]^D\ e[x \mapsto d]\ k$, it suffices to prove that M has the property $\kappa \to \rho$ whenever x is interpreted by d, which is represented by the assumption $x{:}\delta$ in the premise of (Abs).
- (App): Dually, this rule comes from the equation $[\![MN]\!]^D\, e\, k = [\![M]\!]^D\, e\, \langle [\![N]\!]^D e, k\rangle$. For the application MN to have the property $\kappa \to \rho$ (as in the conclusion) it suffices that if applied to a continuation $k \in [\![\kappa]\!]^{\mathcal{M}}$ it yields a value with property ρ. By the equation, such a value is computed by putting $[\![N]\!]^D e$ before k in the continuation passed to $[\![M]\!]^D e$. Therefore, for the conclusion to hold it suffices to prove that N has type δ and M type $\delta \times \kappa \to \rho$.
- (Cmd): This rule is based on the equation $[\![[\alpha]M]\!]^C e = \langle [\![M]\!]^D e, e\,\alpha\rangle$, which states that the meaning of a command $[\alpha]M$ is a continuation $\langle d, k\rangle$ where d is the meaning of M and $k = e\,\alpha$. For $\langle d, k\rangle$ to have the property $\delta \times \kappa$ (as in the conclusion) we have to check that M has the property δ whenever α denotes the continuation k with property κ. Since the assumptions about the environment are in the contexts Δ in case of names, this is represented by the side condition $\Delta(\alpha) = \kappa$ of the rule.
- (μ): This rule is the more involved case, which corresponds to the equation $[\![\mu\alpha.\mathtt{c}]\!]^D\, e\, k = d\, k'$, where $\langle d, k'\rangle = [\![\mathtt{c}]\!]^C e[\alpha \mapsto k]$. This states that $[\![\mu\alpha.\mathtt{c}]\!]^D e$ is the function that, when applied to a continuation k yields the value of the application of the first component d to the second component k' of a different continuation $\langle d, k'\rangle$, which however depends on k, because it is computed by $\mathtt{c}$ whenever α is sent to k. Now the result $d\,k'$ will have the property ρ if for some κ' both $k' \in [\![\kappa']\!]^{\mathcal{M}}$ and $d \in [\![\kappa' \to \rho]\!]^{\mathcal{M}}$. Therefore, to type $\mu\alpha.\mathtt{c}$ by $\kappa \to \rho$ (as in the conclusion) we have to ensure that the continuation represented by $\mathtt{c}$ has the property $(\kappa' \to \rho) \times \kappa'$, whenever $\alpha{:}\kappa$ occurs in the context (as in the premise).

Remark 4.3 Note how rules (App) and (Abs) are actually instances of the familiar rules for application and λ-abstraction in the simply typed λ-calculus. In fact, $\delta \times \kappa \to \rho \in \mathcal{L}_D$ is equivalent to $\delta \to (\kappa \to \rho) \in \mathcal{L}_\lambda$ so that, if we admitted types of $\mathcal{L}_\lambda$, the following rules would be admissible:

$$(\to E):\ \frac{\Gamma \vdash M : \delta \to (\kappa \to \rho) \mid \Delta \quad \Gamma \vdash N : \delta \mid \Delta}{\Gamma \vdash MN : \kappa \to \rho \mid \Delta} \qquad (\to I):\ \frac{\Gamma, x{:}\delta \vdash M : \kappa \to \rho \mid \Delta}{\Gamma \vdash \lambda x.M : \delta \to (\kappa \to \rho) \mid \Delta}$$

Remark 4.4 Rule (Cmd) is equivalent to the following two:

$$(Cmd_1):\ \frac{\Gamma \vdash M:\delta \mid \Delta}{\Gamma \vdash [\alpha]M:\delta\times\kappa \mid \alpha{:}\kappa,\Delta} \qquad (Cmd_2):\ \frac{\Gamma \vdash M:\delta \mid \Delta}{\Gamma \vdash [\alpha]M:\delta\times\omega \mid \Delta}\ (\Delta(\alpha)=\omega)$$

By definition, the context $\alpha{:}\kappa,\Delta$ in the conclusion of (Cmd_1) is only legal when either $\alpha \notin dom(\Delta)$ or $\alpha{:}\kappa \in \Delta$.

The need of (Cmd_2) will become apparent when proving the admissibility of the strengthening rule (Lem. 4.7) and the completeness of the type assignment. For the moment we observe that with (Cmd_1) the conclusion of the shape $\Gamma \vdash [\alpha]M:\delta\times\omega \mid \Delta$ would be derivable from $\Gamma \vdash M:\delta \mid \Delta$ only if $\alpha{:}\omega \in \Delta$; note that $\Delta(\alpha)=\omega$ does not require that $\alpha \in dom(\Delta)$. On the other hand, (Cmd_2) allows the implicit typing of α by ω even if $\alpha \notin dom(\Delta)$. Not having rule (Cmd_2) (that is a particular case of (Cmd)) would introduce an asymmetry with respect to the typing with ω of the term variable x in a basis Γ, since we can conclude $\Gamma \vdash x:\omega \mid \Delta$ either by rule (Ax) (in which case $x{:}\omega \in \Gamma$ is required), or by rule (ω), where $x \notin \Gamma$ is allowed.

With the above proviso, in the proofs we shall often consider the rules (Cmd_1) and (Cmd_2) instances of (Cmd) without explicit mention.

The admissibility of the following rules will be useful:

Lemma 4.5 (Admissible rules) *The following rules are admissible:*

$$(Wk):\ \frac{\Gamma \vdash T:\sigma \mid \Delta}{\Gamma' \vdash T:\sigma \mid \Delta'}\ (\Gamma \subseteq \Gamma' \,\&\, \Delta \subseteq \Delta')$$

$$(Th):\ \frac{\Gamma \vdash T:\sigma \mid \Delta}{\Gamma' \vdash T:\sigma \mid \Delta'}\ (\Gamma' \supseteq \{x{:}\delta \in \Gamma \mid x \in fv(T)\},\ \Delta' \supseteq \{\alpha{:}\kappa \in \Delta \mid \alpha \in fn(T)\})$$

$$(\wedge_g):\ \frac{\Gamma \vdash T:\sigma_i \mid \Delta \quad (\forall i \in I)}{\Gamma \vdash T:\wedge_{i\in I}\sigma_i \mid \Delta}$$

Proof Easy. □

Notice that, by our interpretation of Barendregt's convention, the variables in Γ' and names in Δ' are not bound in T.

In presence of the subtyping $(\leq)$ we can have a further form of weakening, namely by weakening the types in the assumptions. We first extend the operator $\wedge$ and the pre-orders $\leq_D$ and $\leq_C$ to bases and contexts.

Definition 4.6 *i*) For bases Γ_1,Γ_2 we define the basis $\Gamma_1\wedge\Gamma_2$ by:

$$\begin{aligned}\Gamma_1\wedge\Gamma_2 \ \triangleq\ & \{x{:}\Gamma_1(x)\wedge\Gamma_2(x) \mid x \in dom(\Gamma_1)\cap dom(\Gamma_2)\}\\ & \cup \{x{:}\delta \in \Gamma_1 \mid x \notin dom(\Gamma_2)\}\\ & \cup \{x{:}\delta \in \Gamma_2 \mid x \notin dom(\Gamma_1)\}\end{aligned}$$

For contexts Δ_1,Δ_2, we define the context $\Delta_1\wedge\Delta_2$ similarly.

ii) We extend the relations $\leq_D$ and $\leq_C$ to bases and contexts respectively by:

$$\begin{aligned}\Gamma_1 \leq_D \Gamma_2 \ &\triangleq\ \forall x \in \textsc{Var}[\Gamma_1(x) \leq_D \Gamma_2(x)]\\ \Delta_1 \leq_C \Delta_2 \ &\triangleq\ \forall \alpha \in \textsc{Name}[\Delta_1(\alpha) \leq_C \Delta_2(\alpha)]\end{aligned}$$

Note that, if Γ_1, Γ_2 are well-formed bases then so is $\Gamma_1\wedge\Gamma_2$, and if Δ_1, Δ_2 are well-formed contexts, then so is $\Delta_1\wedge\Delta_2$. Also, $dom(\Gamma_1\wedge\Gamma_2)=dom(\Gamma_1)\cup dom(\Gamma_2)$ and $dom(\Delta_1\wedge\Delta_2)=dom(\Delta_1)\cup dom(\Delta_2)$. Therefore $\Gamma_1\wedge\Gamma_2$ is often called 'union of bases' in the literature.

The relations $\Gamma_1 \leq_D \Gamma_2$ and $\Delta_1 \leq_C \Delta_2$ are the pointwise extensions of the relations $\leq_D$ and $\leq_C$ over types; note that the quantifications are not restricted to the domains of the bases nor of the contexts.

Another immediate consequence of the definition is that $\Gamma_1 \wedge \Gamma_2 \leq_D \Gamma_i$ and $\Delta_1 \wedge \Delta_2 \leq_C \Delta_i$, for $i = 1,2$. However if $\Gamma_1 \leq_D \Gamma_2$ then $dom(\Gamma_1)$ and $dom(\Gamma_2)$ are unrelated in general, since we have for example $\{x{:}\omega, y{:}\delta_1 \wedge \delta_2\} \leq_D \{z{:}\omega, y{:}\delta_1\}$. Therefore $\Gamma_1 \leq_D \Gamma_2$ does not imply that Γ_1 and $\Gamma_1 \wedge \Gamma_2$ are the equal, as one perhaps would expect; this is however without consequence since in this case $\Gamma_1 \leq_D \Gamma_1 \wedge \Gamma_2$, so that using the admissibility of strengthening to be shown below, one can prove that all the typings obtainable by means of either basis, can be obtained by the other one. A similar remark holds for contexts.

Now we are in place to prove the admissibility of strengthening:

Lemma 4.7 (Admissibility of Strengthening) *The following rule is admissible:*

$$(St): \ \frac{\Gamma \vdash T : \sigma \mid \Delta}{\Gamma' \vdash T : \sigma \mid \Delta'} \ (\Gamma' \leq_D \Gamma \ \& \ \Delta' \leq_C \Delta)$$

Proof By straightforward induction over the structure of derivations. □

It is straightforward to show that $\Gamma \subseteq \Gamma'$ implies $\Gamma' \leq \Gamma$ (and $\Delta \subseteq \Delta'$ implies $\Delta' \leq \Delta$), so rule (St) contains rule (Wk).

The following lemma describes the set of types that can be assigned to a term or a command.

Lemma 4.8 If $\mathcal{D} :: \Gamma \vdash T : \sigma \mid \Delta$, then either $\sigma \sim_A \omega$ or there exist sub-derivations $\mathcal{D}_i :: \Gamma \vdash T : \sigma_i \mid \Delta$ of $\mathcal{D}$ $(i \in \underline{n})$, such that $\wedge_{i=1}^n \sigma_i \leq_A \sigma$ and the last rule of each $\mathcal{D}_i$ is a type rule.

Proof By straightforward induction over the structure of derivations. □

This particular property will be of use in many of the proofs below, where we reason by induction over the structure of derivations and allows us to always assume that a type rule was applied last, and not treat the logical rules.

Another way to state the above result is the following:

Lemma 4.9 (Generation Lemma) *Let $\delta \not\sim_D \omega$ and $\kappa \not\sim_C \omega$:*

$$\begin{array}{rcl}
\Gamma \vdash x : \delta \mid \Delta & \Leftrightarrow & \exists\, x{:}\delta \in \Gamma\, [\delta' \leq_D \delta] \\
\Gamma \vdash \lambda x.M : \delta \mid \Delta & \Leftrightarrow & \exists I \forall i \in I\ \exists \delta_i, \kappa_i, \rho_i\, [\Gamma, x{:}\delta_i \vdash M : \kappa_i \to \rho_i \mid \Delta \ \& \ \wedge_I \delta_i \times \kappa_i \to \rho_i \leq_D \delta] \\
\Gamma \vdash MN : \delta \mid \Delta & \Leftrightarrow & \\
 & & \exists I \forall i \in I\ \exists \delta_i, \kappa_i, \rho_i\, [\Gamma \vdash M : \delta_i \times \kappa_i \to \rho_i \mid \Delta \ \& \ \Gamma \vdash N : \delta_i \mid \Delta \ \& \ \wedge_I \kappa_i \to \rho_i \leq_D \delta] \\
\Gamma \vdash \mu\alpha.\mathtt{C} : \delta \mid \Delta & \Leftrightarrow & \exists I \forall i \in I\ \exists \kappa_i, \rho_i, \kappa_i'\, [\Gamma \vdash \mathtt{C} : (\kappa_i \to \rho_i) \times \kappa_i \mid \alpha{:}\kappa_i', \Delta \ \& \ \wedge_I \kappa_i' \to \rho_i \leq_D \delta] \\
\Gamma \vdash [\alpha]M : \kappa \mid \Delta & \Leftrightarrow & \exists \kappa', I \forall i \in I\ \exists \delta_i\, [\Gamma \vdash M : \delta_i \mid \Delta \ \& \ \Delta(\alpha) = \kappa' \ \& \ \wedge_I \delta_i \times \kappa' \leq_C \kappa]
\end{array}$$

Proof The proof is standard. As an illustration we just show one of the cases in detail.

$(\Leftarrow)$: Assume that for all $i \in I$ $\Gamma \vdash M : \delta_i \mid \Delta$ and $\Delta(\alpha) = \kappa' \leq_C \kappa_i$, and that $\wedge_{i \in I} \delta_i \times \kappa_i \leq_C \kappa$. Then we can construct:

$$\dfrac{\dfrac{\dfrac{\boxed{}}{\Gamma \vdash M : \delta_i \mid \Delta}\ (Cmd)}{\Gamma \vdash [\alpha]M : \delta_i \times \kappa' \mid \Delta}\ (\forall i \in I)}{\dfrac{\Gamma \vdash [\alpha]M : \wedge_{i \in I} \delta_i \times \kappa_i \mid \Delta}{\Gamma \vdash [\alpha]M : \kappa \mid \Delta}\ (\leq)}\ (\wedge_g)$$

$(\Rightarrow)$: Let $\Gamma \vdash [\alpha]M : \kappa \mid \Delta$; the derivation ends with either:

(Cmd): Then the derivation is shaped like:

$$\dfrac{\Gamma \vdash M : \delta \mid \Delta}{\Gamma \vdash [\alpha]M : \delta \times \kappa'' \mid \Delta}\ (\Delta(\alpha) = \kappa'')$$

with $\kappa = \delta \times \kappa''$; take $\kappa' = \kappa''$, $I = \{1\}$, $\delta_1 = \delta$.

$(\wedge)$: Then the derivation is shaped like:

$$\dfrac{\Gamma \vdash T : \kappa_1 \mid \Delta \qquad \Gamma \vdash T : \kappa_2 \mid \Delta}{\Gamma \vdash T : \kappa_1 \wedge \kappa_2 \mid \Delta}$$

By induction, there exist κ'_1, κ'_2, I_1, I_2, such that

$$\forall i \in I_1\ \exists \delta_i^1 \left[\, \Gamma \vdash M : \delta_i^1 \mid \Delta \ \&\ \Delta(\alpha) = \kappa'_1 \ \&\ \textstyle\bigwedge_{I_1} \delta_i \times \kappa'_1 \leq_C \kappa_1 \,\right]\ \&$$
$$\forall i \in I_2\ \exists \delta_i^2 \left[\, \Gamma \vdash M : \delta_i^2 \mid \Delta \ \&\ \Delta(\alpha) = \kappa'_2 \ \&\ \textstyle\bigwedge_{I_2} \delta_i^2 \times \kappa'_2 \leq_C \kappa_2 \,\right]$$

Then necessarily $\kappa'_1 = \kappa'_2$. Take $\kappa' = \kappa'_1 = \kappa'_2$, $I = I_1 \cup I_2$, then for all δ_i with $i \in I$ we have $\Gamma \vdash M : \delta_i \mid \Delta$, and $\bigwedge_I \delta_i \times \kappa' = (\bigwedge_{I_1} \delta_i^1 \times \kappa'_1) \wedge (\bigwedge_{I_2} \delta_i^2 \times \kappa'_2) \leq_C \kappa_1 \wedge \kappa_2$.

$(\leq)$: Then the derivation is shaped like:

$$\dfrac{\Gamma \vdash T : \kappa_1 \mid \Delta \qquad \kappa_1 \leq \kappa_2}{\Gamma \vdash T : \kappa_2 \mid \Delta}$$

By induction, there exist κ', I, such that

$$\forall i \in I\ \exists \delta_i \left[\, \Gamma \vdash M : \delta_i \mid \Delta \ \&\ \Delta(\alpha) = \kappa' \ \&\ \textstyle\bigwedge_I \delta_i \times \kappa' \leq_C \kappa_1 \,\right]$$

Notice that then also $\bigwedge_I \delta_i \times \kappa' \leq_C \kappa_2$. □

4.2 Type interpretation and soundness

In this section we will formally define the type interpretation and thereby the interpretation of typing judgements. As anticipated above in the informal discussion of the system, the meaning of a type will be a subset of the domain of interpretation.

In definitions and statements below we relate types to a $\lambda\mu$-model $\mathcal{M} = (R, D, C)$, silently assuming that the language $\mathcal{L}_R$ includes a constant v_a for every $a \in \mathcal{K}(R)$.

Definition 4.10 (Type interpretation) Let $\mathcal{M} = (R, D, C)$ be a $\lambda\mu$-model. For $A = R, D, C$ we define the interpretation $[\![\cdot]\!]^{\mathcal{M},A} : \mathcal{L}_A \to \mathcal{P}(A)$ (written $[\![\cdot]\!]^A$ when $\mathcal{M}$ is understood) as follows:

$$\begin{array}{rcl} [\![v_a]\!]^R & = & \uparrow_R a = \{r \in R \mid a \sqsubseteq r\} \\ [\![\delta \times \kappa]\!]^C & = & [\![\delta]\!]^D \times [\![\kappa]\!]^C \\ [\![\kappa \to \rho]\!]^D & = & \{d \in D \mid \forall k \in [\![\kappa]\!]^C\, [\, dk \in [\![\rho]\!]^R \,]\} \\ [\![v_a]\!]^D = [\![\omega \to v_a]\!]^D & = & \{d \in D \mid \forall k \in C\, [\, dk \in [\![v_a]\!]^R \,]\} \end{array}$$

and

$$\begin{array}{rcl} [\![\omega]\!]^A & = & A \\ [\![\sigma_1 \wedge \sigma_2]\!]^A & = & [\![\sigma_1]\!]^A \cap [\![\sigma_2]\!]^A \end{array}$$

Remark 4.11 The last definition is a special case with respect to the natural adaptation of the intersection type interpretation as subsets of a λ-model, in that we fix the interpretation of

the type constants v_a. This is consistent with the approach of constructing types from the solution of the continuation domain equations, and is the intended interpretation throughout this paper. In particular, it implies that the language $\mathcal{L}_R$ depends on the chosen domain of results R, and that the interpretation of a type is always a principal filter of either R, D, or C, according to its kind, as is proven in the next lemma.

This choice poses no limitations. If we postulate that there exist denumerably many constants $v_0, v_1, \ldots$ in $\mathcal{L}_R$, then we can generalise the definition of type interpretation in a straightforward way to $[\![\sigma]\!]^A_\eta$ (relative to the *type environment* η, a mapping of type constants such that $\eta(v_i) \subseteq R$ for all i) by defining $[\![v_i]\!]^R_\eta = \eta(v_i)$ as the base case of the inductive definition. Then the above definition is recovered by considering an arbitrary exhaustive enumeration of the compacts $a_0, a_1, \ldots = \mathcal{K}(R)$ (possibly with repetitions; this enumeration exists since R is ω-algebraic) and defining the interpretation of type constants through $\eta_0(v_i) = \uparrow_R a_i$.

There exists a close relation between the interpretation of types and the maps Θ_A (see Def. 3.18), that is made explicit in the following lemma.

Lemma 4.12 For $A = R, D, C$ and any $\sigma \in \mathcal{L}_A$, we have that $[\![\sigma]\!]^A = \uparrow_A \Theta_A(\sigma)$.

Proof By induction over the structure of types and by cases on A.

$(\sigma \equiv \omega)$: Then $\Theta_A(\omega) = \bot$ and $\uparrow_A \bot = A = [\![\omega]\!]^A$.

$(\sigma \equiv \sigma_1 \wedge \sigma_2)$: By induction, $[\![\sigma_i]\!]^A = \uparrow_A \Theta_A(\sigma_i)$ and by Rem. 3.19, $\Theta_A(\sigma_1 \wedge \sigma_2) = \Theta_A(\sigma_1) \sqcup \Theta_A(\sigma_2)$. We have $\uparrow_A(\Theta_A(\sigma_1) \sqcup \Theta_A(\sigma_2)) = \uparrow_A \Theta_A(\sigma_1) \cap \uparrow_A \Theta_A(\sigma_2)$, and therefore $[\![\sigma_1 \wedge \sigma_2]\!]^A = [\![\sigma_1]\!]^A \cap [\![\sigma_2]\!]^A = \uparrow_A \Theta_A(\sigma_1 \wedge \sigma_2)$.

$(\sigma \equiv v_a)$: $(A = R)$: This follows immediately from $\Theta_R(v_a) = a$ and $[\![v_a]\!]^R = \uparrow_R a$.

$(A = D)$: Then $\Theta_D(v_a) = (\bot \Rightarrow a)$; but for any $d \in D = [C \to R]$, by definition of step functions, $(\bot \Rightarrow a) \sqsubseteq d$ if and only if $a \sqsubseteq d\, k$ for all $k \in C$, that is if and only if $d\, k \in [\![v_a]\!]^R$ for the above; it follows that $\Theta_D(v_a) \sqsubseteq d$ if and only if $d \in [\![v_a]\!]^D$ as desired.

$(\sigma \equiv \delta \times \kappa)$: Then $\Theta_C(\delta \times \kappa) = \langle \Theta_D(\delta), \Theta_C(\kappa) \rangle$ by Rem. 3.19, and for any $\langle d, k \rangle \in C = D \times C$ we have:

$$\begin{array}{rcll} \langle \Theta_D(\delta), \Theta_C(\kappa) \rangle \sqsubseteq \langle d, k \rangle & \iff & \Theta_D(\delta) \sqsubseteq d \;\&\; \Theta_C(\kappa) \sqsubseteq k & \text{(by definition of order over } D \times C) \\ & \iff & d \in [\![\delta]\!]^D \;\&\; k \in [\![\kappa]\!]^C & \text{(by ind.)} \\ & \iff & \langle d, k \rangle \in [\![\delta]\!]^D \times [\![\kappa]\!]^C & \text{(by Def. 4.10)} \\ & = & [\![\delta \times \kappa]\!]^C & \end{array}$$

$(\sigma \equiv \kappa \to \rho)$: Then $\Theta_D(\kappa \to \rho) = (\Theta_C(\kappa) \Rightarrow \Theta_R(\rho))$ by Rem. 3.19; for any $d \in D = [C \to R]$ we have:

$$\begin{array}{rcll} (\Theta_C(\kappa) \Rightarrow \Theta_R(\rho)) \sqsubseteq d & \iff & \forall k \in C\, [\Theta_C(\kappa) \sqsubseteq k \Rightarrow \Theta_R(\rho) \sqsubseteq d\, k] & \text{(by definition of } (\cdot \Rightarrow \cdot)) \\ & \iff & \forall k \in C\, [k \in [\![\kappa]\!]^C \Rightarrow d\, k \in [\![\rho]\!]^R] & \text{(IH)} \\ & \iff & d \in [\![\kappa \to \rho]\!]^D & \text{(by Def. 4.10)} \end{array}$$

□

Corollary 4.13 For $A = R, D, C$, if $\sigma, \tau \in \mathcal{L}_A$ then $\sigma \leq_A \tau \iff [\![\sigma]\!]^A \subseteq [\![\tau]\!]^A$.

Proof

$$\begin{array}{rcll} \sigma \leq_A \tau & \iff & \Theta_A(\sigma) \sqsupseteq \Theta_A(\tau) & \text{(by Lem. 3.10 (}ii\text{) and 3.20)} \\ & \iff & \uparrow \Theta_A(\sigma) \subseteq \uparrow \Theta_A(\tau) & \text{(by Lem. 4.12)} \\ & \iff & [\![\sigma]\!]^A \subseteq [\![\tau]\!]^A & \end{array}$$

□

We will now define satisfiability for typing judgements with respect to a $\lambda\mu$-model.

Definition 4.14 (Satisfiability) Let $\mathcal{M} = (R, D, C)$ be a $\lambda\mu$-model. We define semantic satisfiability through:

$$\begin{array}{lll} e \models_{\mathcal{M}} \Gamma;\Delta & \Leftrightarrow & \forall x\,[e\,x \in [\![\Gamma(x)]\!]^{\mathcal{D}}_{\mathcal{M}}] \;\&\; \forall \alpha\,[e\,\alpha \in [\![\Delta(\alpha)]\!]^{\mathcal{C}}_{\mathcal{M}}] \\ \Gamma \models_{\mathcal{M}} M : \delta \mid \Delta & \Leftrightarrow & \forall e\,[e \models_{\mathcal{M}} \Gamma;\Delta \Rightarrow [\![M]\!]^{\mathcal{D}}_{\mathcal{M}}\, e \in [\![\delta]\!]^{\mathcal{D}}_{\mathcal{M}}] \\ \Gamma \models_{\mathcal{M}} \mathtt{C} : \kappa \mid \Delta & \Leftrightarrow & \forall e\,[e \models_{\mathcal{M}} \Gamma;\Delta \Rightarrow [\![\mathtt{C}]\!]^{\mathcal{C}}_{\mathcal{M}}\, e \in [\![\kappa]\!]^{\mathcal{C}}_{\mathcal{M}}] \end{array}$$

We will write $\models$ for $\models_{\mathcal{M}}$ when $\mathcal{M}$ is understood.

Remark 4.15 Continuing the discussion in Rem. 4.11, we note that we do not consider here the concept of *validity*, namely satisfiability with respect to any $\lambda\mu$-model $\mathcal{M}$, since we model both the language $\mathcal{L}_R$ and the pre-order $\leq_R$ *after* R, which is the particular domain of results of $\mathcal{M}$.

As a matter of fact, we could define validity as follows: first we fix the type theory $\mathcal{T}_R$ for the language $\mathcal{L}_R$ with denumerably many type constants v_i; then the satisfiability notion should be relativised to *both* term *and* type environments e and η, asking that the latter is a model of the theory $\mathcal{T}_R$, in the sense that whenever $\rho \leq_R \rho'$, it holds that $\eta(\rho) \subseteq \eta(\rho')$.

However, we will not consider such a general formulation, as it would involve an unnecessary complication of the theory developed here.

The next result states the soundness of the typing system. Note that, although the construction of the system has been made by having an initial model in mind, the soundness theorem holds for any model.

Theorem 4.16 (Soundness of type assignment) *Let $\mathcal{M}$ be a $\lambda\mu$-model. If $\Gamma \vdash T : \sigma \mid \Delta$, then $\Gamma \models T : \sigma \mid \Delta$.*

Proof By induction on the structure of derivations. (We will drop the super and subscripts on the interpretation function.)

- (Ax): Then $T \equiv x$ and $\sigma = \delta$; let $e \models \Gamma, x{:}\delta; \Delta$, then $e\,x \in [\![\delta]\!]$. Hence, by Def. 2.3, we get $[\![x]\!]\, e \in [\![\delta]\!]$.
- (Abs): Then $T \equiv \lambda x.N$ and there exist δ, κ and ρ such that $\sigma = \delta \times \kappa \to \rho$ and $\Gamma, x{:}\delta \vdash N : \kappa \to \rho \mid \Delta$. By definition, $[\![\lambda x.N]\!]\, e\, k = [\![N]\!]\, e[x \mapsto d]\, k'$, where $k = \langle d, k' \rangle$; also, $e \models \Gamma;\Delta$ and $d \in [\![\delta]\!]$ if and only if $e[x \mapsto d] \models \Gamma, x{:}\delta; \Delta$. By induction, for any $e \models \Gamma;\Delta$ and $d \in [\![\delta]\!]$, $\Gamma, x{:}\delta \models N : \kappa \to \rho \mid \Delta$, so $[\![N]\!]\, e[x \mapsto d] \in [\![\kappa \to \rho]\!]$, so $[\![N]\!]\, e[x \mapsto d]\; k \in [\![\rho]\!]$ for any $k \in [\![\kappa]\!]$. So $[\![\lambda x.N]\!]\, e\, \langle d, k \rangle \in [\![\rho]\!]$, so $[\![\lambda x.N]\!]\, e \in [\![\delta \times \kappa \to \rho]\!]$, and we conclude $\Gamma \models \lambda x.N : \delta \times \kappa \to \rho \mid \Delta$.
- (App): Then $T \equiv PQ$ and there exist δ, κ and ρ such that $\sigma = \kappa \to \rho$, $\Gamma \vdash P : \delta \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash Q : \delta \mid \Delta$. By definition, $[\![PQ]\!]\, e\, k = [\![P]\!]\, e\, \langle [\![Q]\!]\, e, k \rangle$. Let $e \models \Gamma;\Delta$. By induction, $\Gamma \models P : \delta \times \kappa \to \rho \mid \Delta$ and $\Gamma \models Q : \delta \mid \Delta$, so $[\![P]\!]\, e \in [\![\delta \times \kappa \to \rho]\!]$ and $[\![Q]\!]\, e \in [\![\delta]\!]$; in particular, for any $\langle d, k' \rangle \in [\![\delta]\!] \times [\![\kappa]\!] = [\![\delta \times \kappa]\!]$, we have $[\![P]\!]\, e\, \langle d, k' \rangle \in [\![\rho]\!]^R$, so $[\![PQ]\!]\, e \in [\![\kappa \to \rho]\!]$, and thereby $\Gamma \models PQ : \kappa \to \rho \mid \Delta$.
- (Cmd): Then $T \equiv [\alpha]N$ and there exist δ and $\kappa = \Delta(\alpha)$ such that $\sigma = \delta \times \kappa$ and $\Gamma \vdash N : \delta \mid \Delta$. By induction we have that $\Gamma \models N : \delta \mid \Delta$, so that for any $e \models \Gamma;\Delta$ we have that $[\![N]\!] e \in [\![\delta]\!]$. But $e \models \Gamma;\Delta$ implies that $e\,\alpha \in [\![\Delta(\alpha)]\!] = [\![\kappa]\!]$, hence $[\![[\alpha]N]\!] e = \langle [\![N]\!] e, e\,\alpha \rangle \in [\![\delta]\!] \times [\![\kappa]\!] = [\![\delta \times \kappa]\!]$ as desired. Then $\Gamma \models [\alpha]N : \delta \times \kappa \mid \Delta$ by the arbitrary choice of e.
- (μ): Then $T \equiv \mu\alpha.\mathtt{C}$, and there exist κ, κ' and ρ such that $\sigma = \kappa \to \rho$ and $\Gamma \vdash \mathtt{C} : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta$. By definition, $[\![\mu\alpha.\mathtt{C}]\!]\, e\, k = d\, k'$, where $\langle d, k' \rangle = [\![\mathtt{C}]\!]\; e[\alpha \mapsto k]$. Let $e \models \Gamma;\Delta$, and $k \in [\![\kappa]\!]$, then $e[\alpha \mapsto k] \models \Gamma; \alpha{:}\kappa, \Delta$. By induction, $\Gamma \models \mathtt{C} : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta$, so $[\![\mathtt{C}]\!]\; e[\alpha \mapsto k] \in [\![(\kappa' \to \rho) \times \kappa']\!]$. Let $[\![\mathtt{C}]\!]\; e[\alpha \mapsto k] = p$, then $\pi_1\, p \in [\![\kappa' \to \rho]\!]$ and $\pi_2\, p \in [\![\kappa']\!]$, and $\pi_1\, p\, (\pi_2\, P) \in$

$[\![\rho]\!]$, so $[\![\mu\alpha.\mathtt{C}]\!]\, e\, k \in [\![\rho]\!]$, for any $k \in [\![\kappa]\!]$, so $[\![\mu\alpha.\mathtt{C}]\!]\, e \in [\![\kappa{\to}\rho]\!]$, and therefore $\Gamma \models \mu\alpha.\mathtt{C} : \kappa{\to}\rho \mid \alpha{:}\kappa, \Delta$.

$(\wedge)$: By induction and the interpretation of an intersection type.

(ω): Immediate by the definition of interpretation of ω.

$(\leq)$: By induction and Cor. 4.13. □

We will now show that we can interpret a term or command by the set of types that can be given to it (Thm. 4.19). Towards that result, we first make the denotation of terms and commands and the interpretations of types in the filter model $\mathcal{F}$ explicit.

Lemma 4.17 The following equations hold:

$$\begin{aligned}
[\![\lambda x.M]\!]^{\mathcal{F}} e &= \uparrow_D \{ \wedge_{i\in I}(\delta_i{\times}\kappa_i{\to}\rho_i) \in \mathcal{L}_D \mid \forall i \in I\, [\kappa_i{\to}\rho_i \in [\![M]\!]^{\mathcal{F}} e[x{\mapsto}(\uparrow_D \delta_i)]\,]\,\} \\
[\![MN]\!]^{\mathcal{F}} e &= \uparrow_D \{ \wedge_{i\in I}\kappa_i{\to}\rho_i \in \mathcal{L}_D \mid \forall i \in I\ \exists \delta_i \in [\![N]\!]^{\mathcal{F}} e, \kappa_i \in \mathcal{L}_C\, [\delta_i{\times}\kappa_i{\to}\rho_i \in [\![M]\!]^{\mathcal{F}} e]\,\} \\
[\![\mu\alpha.\mathtt{C}]\!]^{\mathcal{F}} e &= \uparrow_D \{ \wedge_{i\in I}\kappa_i{\to}\rho_i \in \mathcal{L}_D \mid \forall i \in I\ \exists \kappa'_i \in \mathcal{L}_C\, [(\kappa'_i{\to}\rho_i){\times}\kappa'_i \in [\![\mathtt{C}]\!]^{\mathcal{F}} e[\alpha{\mapsto}(\uparrow_C \kappa_i)]\,]\,\} \\
[\![[\alpha]M]\!]^{\mathcal{F}} e &= \uparrow_C \{ \wedge_{i\in I}\delta_i{\times}\kappa_i \in \mathcal{L}_C \mid \forall i \in I\, [\delta_i \in [\![M]\!]^{\mathcal{F}} e \;\&\; \kappa_i \in e(\alpha)\,]\,\}
\end{aligned}$$

Proof By unravelling definitions. For example:

$$\begin{aligned}
[\![\lambda x.M]\!]^{\mathcal{F}^D} e &= \lambda\!\!\lambda d{::}k \in \mathcal{F}_C\,.\,[\![M]\!]^{\mathcal{F}}\, e[x{\mapsto}d]\ k \\
&= \bigcup \{ \sqcup_{i\in I}(\uparrow_C \delta_i{\times}\kappa_i \Rightarrow \uparrow_R \rho_i) \mid \uparrow_R \rho_i \subseteq [\![M]\!]^{\mathcal{F}}\, e[x{\mapsto}\uparrow_D \delta_i]\ (\uparrow_C \kappa_i) \} \\
&= \uparrow_D \{ \wedge_{i\in I}(\delta_i{\times}\kappa_i{\to}\rho_i) \in \mathcal{L}_D \mid \forall i \in I\, [\kappa_i{\to}\rho_i \in [\![M]\!]^{\mathcal{F}} e[x{\mapsto}(\uparrow_D \delta_i)]\,]\,\}
\end{aligned}$$

using the fact that $\lambda\!\!\lambda d{::}k \in \mathcal{F}_C\,.\,[\![M]\!]^{\mathcal{F}}\, e[x{\mapsto}d]\ k$ is continuous, hence it is the *sup* of finite joins of the step functions $(\uparrow_C \delta_i{\times}\kappa_i \Rightarrow \uparrow_R \rho_i)$. Observe that $d{::}k = \uparrow_C \{ \wedge_{i\in I}\delta_i{\times}\kappa_i \mid \forall i \in I\, [\delta_i \in d \;\&\; \kappa_i \in k]\,\}$, that the set $\{ \sqcup_{i\in I}(\uparrow_C \delta_i{\times}\kappa_i \Rightarrow \uparrow_R \rho_i) \mid \ldots \}$ is directed and hence its join is its union, and finally that $\uparrow_R \rho_i \subseteq [\![M]\!]^{\mathcal{F}}\, e[x{\mapsto}\uparrow_D \delta_i]\ (\uparrow_C \kappa_i)$ if and only if $\kappa_i{\to}\rho_i \in [\![M]\!]^{\mathcal{F}} e[x{\mapsto}(\uparrow_D \delta_i)]$. □

Lemma 4.18 For $A = R, D, C$: if $\sigma \in \mathcal{L}_A$ then, up to the isomorphisms $\mathcal{F}_R \simeq R$, $\mathcal{F}_D{\times}\mathcal{F}_C \simeq \mathcal{F}_C$ and $[\mathcal{F}_C \to \mathcal{F}_R] \simeq \mathcal{F}_D$, we have:

i) $\Theta_{\mathcal{F}_A}(\sigma) = \uparrow_A \sigma$,

ii) $[\![\sigma]\!]^{\mathcal{F}} = \{ a \in \mathcal{F}_A \mid \sigma \in a \}$.

Proof Recall that the isomorphism $\mathcal{F}_R \simeq R$ is established by Prop. 3.8 and described in detail in Rem. 3.12, and that $K : \mathcal{F}_D{\times}\mathcal{F}_C \to \mathcal{F}_C$ and $G : [\mathcal{F}_C \to \mathcal{F}_R] \to \mathcal{F}_D$ the are isomorphisms of Def. 3.24 and Thm. 3.27. Now to prove part (*i*) we proceed by induction over the structure of types.

$(\sigma \equiv \omega)$: Then $\Theta_{\mathcal{F}_A}(\omega) = \bot_{\mathcal{F}_A} = \uparrow_A \omega$.

$(\sigma \equiv \sigma{\wedge}\tau)$:
$$\begin{aligned}
\Theta_{\mathcal{F}_A}(\sigma{\wedge}\tau) &= \Theta_{\mathcal{F}_A}(\sigma) \sqcup \Theta_{\mathcal{F}_A}(\tau) && \text{(by Def. 3.15 and Rem. 3.19)} \\
&= \uparrow_A \sigma \sqcup \uparrow_A \tau && \text{(IH)} \\
&= \uparrow_A (\sigma{\wedge}\tau)
\end{aligned}$$

$(\sigma \equiv v_a)$: By Rem. 3.12, under the isomorphism $\mathcal{F}_R \simeq R$ the compact point $a \in \mathcal{K}(R)$ is the image of $\uparrow_R v_a$; hence, up to isomorphism, we have that $\Theta_{\mathcal{F}_R}(v_a) = a = \uparrow_R v_a$.

$(\sigma \equiv \delta{\times}\kappa)$:
$$\begin{aligned}
\Theta_{\mathcal{F}_C}(\delta{\times}\kappa) &= \langle \Theta_{\mathcal{F}_D}(\delta), \Theta_{\mathcal{F}_C}(\kappa) \rangle && \text{(by Def. 3.15 and Rem. 3.19)} \\
&= \langle \uparrow_D \delta, \uparrow_C \kappa \rangle && \text{(IH)} \\
&= (\uparrow_D \delta) :: (\uparrow_C \kappa) && \text{(up to the iso } K\text{)} \\
&= \uparrow_C (\delta{\times}\kappa) && \text{(by Rem. 3.25)}
\end{aligned}$$

$$\begin{array}{lrcll}(\sigma \equiv \kappa{\to}\rho)\colon & \Theta_{\mathcal{F}_D}(\kappa{\to}\rho) & = & (\Theta_{\mathcal{F}_C}(\kappa) \Rightarrow \Theta_{\mathcal{F}_R}(\rho)) & \text{(by Def. 3.15 and Rem. 3.19)}\\ & & = & (\uparrow_C \kappa \Rightarrow \uparrow_R \rho) & \text{(IH)}\\ & & = & \uparrow_D(\kappa{\to}\rho) & \text{(up to the iso } G \text{ and by Rem. 3.25)}\end{array}$$

To prove part (*ii*), observe that for any $\sigma \in \mathcal{L}_A$ and $a \in \mathcal{F}_A$, $\uparrow_A \sigma = \{\tau \in \mathcal{L}_A \mid \sigma \leq_A \tau\} \subseteq a$ if and only $\sigma \in a$. Now by Lem. 4.12 we know that $[\![\sigma]\!]^{\mathcal{F}} = \uparrow_{\mathcal{F}_A} \Theta_{\mathcal{F}_A}(\sigma)$, which, by part (*i*) of this lemma, implies $[\![\sigma]\!]^{\mathcal{F}} = \uparrow_{\mathcal{F}_A}(\uparrow_A \sigma) = \{a \in \mathcal{F}_A \mid \uparrow_A \sigma \subseteq a\}$. By the remark above we have that $\{a \in \mathcal{F}_A \mid \uparrow_A \sigma \subseteq a\} = \{a \in \mathcal{F}_A \mid \sigma \in a\}$. □

The next theorem, together with the Completeness Theorem (Thm. 4.23) that it implies, is the main result of this section, which states that the set of types that are assigned to terms and commands by the type assignment system coincides with their interpretation in the filter model. Its proof essentially depends on Lem. 4.9 and Lem. 4.17.

Theorem 4.19 *Let $A = D, C$. Given an environment $e \in \text{ENV}_{\mathcal{F}}$, then*

$$[\![T]\!]^{\mathcal{F}} e \;=\; \{\sigma \in \mathcal{L}_A \mid \exists \Gamma, \Delta [e \models_{\mathcal{F}} \Gamma; \Delta \;\&\; \Gamma \vdash T : \sigma \mid \Delta]\}.$$

Proof Because of the logical rules, the set $\{\sigma \in \mathcal{L}_A \mid \exists \Gamma, \Delta [e \models_{\mathcal{F}} \Gamma; \Delta \;\&\; \Gamma \vdash T : \sigma \mid \Delta]\}$ is a filter in $\mathcal{F}_A$, for $A = D, C$. To prove that this filter coincides with $[\![T]\!]^{\mathcal{F}} e$ we proceed by induction over the structure of terms.

- $(T \equiv x)$: Then $[\![x]\!]^{\mathcal{F}} e = e\,x$. By definition, $e \models_{\mathcal{F}} \Gamma; \emptyset \Leftrightarrow e\,x \in [\![\Gamma(x)]\!]^{\mathcal{F}}$. By Lem. 4.18 (*ii*) we know that $[\![\Gamma(x)]\!]^{\mathcal{F}} = \{d \in \mathcal{F}_D \mid \Gamma(x) \in d\}$, so that $e \models_{\mathcal{F}} \Gamma; \emptyset$ is equivalent to $\Gamma(x) \in e\,x$. On the other hand, by Lem. 4.9, we have that $\Gamma \vdash x : \delta \mid$ if and only if $\Gamma(x) \leq_D \delta$, hence $\Gamma \vdash x : \delta \mid$ if and only if $\delta \in e\,x$.
- $(T \equiv \lambda x.M)$: For $\delta \in [\![\lambda x.M]\!]^{\mathcal{F}} e$, by Lem. 4.17 there exist I such and δ_i, κ_i, and ρ_i $(i \in I)$ such that $\kappa_i{\to}\rho_i \in [\![M]\!]^{\mathcal{F}} e[x \mapsto (\uparrow_D \delta_i)]$. By induction there exist Γ_i, Δ_i $(i \in I)$ such that $e[x \mapsto (\uparrow_D \delta_i)] \models_{\mathcal{F}} \Gamma_i; \Delta_i$ and $\Gamma_i \vdash M : \kappa_i{\to}\rho_i \mid \Delta_i$. Let $\delta_i' = \Gamma_i(x)$: then by rule (Abs) we have $\Gamma_i \backslash x \vdash \lambda x.M : \delta_i' {\times} \kappa_i {\to} \rho_i \mid \Delta_i$.

  On the other hand, from $e[x \mapsto (\uparrow_D \delta_i)] \models_{\mathcal{F}} \Gamma_i; \Delta_i$ we know that $\Gamma_i(x) = \delta_i'$; since $\delta_i' \in \uparrow_D \delta_i$, also $\delta_i \leq_D \delta_i'$. Then $\delta_i {\times} \kappa_i \leq_C \delta_i' {\times} \kappa_i$ follows by the co-variance of $\times$, and $\delta_i' {\times} \kappa_i {\to} \rho_i \leq_D \delta_i {\times} \kappa_i {\to} \rho_i$ by the contra-variance of the arrow in its first argument. Hence, by applying rule $(\leq)$, for all $i \in I$, we obtain $\Gamma_i \backslash x \vdash \lambda x.M : \delta_i {\times} \kappa_i {\to} \rho_i \mid \Delta_i$.

  Take $\Gamma = \wedge_{i \in I} \Gamma_i \backslash x$ and $\Delta = \wedge_{i \in I} \Delta_i$: then $\Gamma \vdash \lambda x.M : \delta_i {\times} \kappa_i {\to} \rho_i \mid \Delta$ for all $i \in I$ by applying rule (Wk), and therefore $\Gamma \vdash \lambda x.M : \wedge_{i \in I} \delta_i {\times} \kappa_i {\to} \rho_i \mid \Delta$ by applying rule $(\wedge)$ and $\Gamma \vdash \lambda x.M : \delta \mid \Delta$ by rule $(\leq)$. Observe that, for all $i \in I$, $x \notin dom(\Gamma_i \backslash x)$ and consequently $x \notin dom(\Gamma)$, so that, for all $i \in I$, $e[x \mapsto (\uparrow_D \delta_i)] \models_{\mathcal{F}} \Gamma_i; \Delta_i$ implies $e \models_{\mathcal{F}} \Gamma_i \backslash x; \Delta_i$ and so $e \models_{\mathcal{F}} \Gamma; \Delta$ as required.

  Vice-versa, if $\Gamma \vdash \lambda x.M : \delta \mid \Delta$ and $e \models_{\mathcal{F}} \Gamma; \Delta$ then, by Lem. 4.9, there exist δ_i, κ_i, and ρ_i $(i \in I)$ such that $\Gamma, x{:}\delta_i \vdash M : \kappa_i {\to} \rho_i \mid \Delta$ and $\wedge_{i \in I} \delta_i {\times} \kappa_i {\to} \rho_i \leq_D \delta$. Observe that $e \models_{\mathcal{F}} \Gamma; \Delta$ implies $e[x \mapsto (\uparrow_D \delta_i)] \models_{\mathcal{F}} \Gamma, x{:}\delta_i; \Delta$ for all $i \in I$. By induction, $\kappa_i {\to} \rho_i \in [\![M]\!]^{\mathcal{F}} e[x \mapsto (\uparrow_D \delta_i)]$, so by Lem. 4.17 $\wedge_{i \in I}(\delta_i {\times} \kappa_i {\to} \rho_i) \in [\![\lambda x.M]\!]^{\mathcal{F}} e$. Now $[\![\lambda x.M]\!]^{\mathcal{F}} e$ is a filter and since $\wedge_{i \in I} \delta_i {\times} \kappa_i {\to} \rho_i \leq_D \delta$, we conclude that $\delta \in [\![\lambda x.M]\!]^{\mathcal{F}} e$.
- $(T \equiv MN)$: If $\delta \in [\![MN]\!]^{\mathcal{F}} e$ then, by Lem. 4.17, there exist I and κ_i, ρ_i and $\delta_i \in [\![N]\!]^{\mathcal{F}} e$ $(i \in I)$ such that $\delta_i {\times} \kappa_i {\to} \rho_i \in [\![M]\!]^{\mathcal{F}} e$ and $\wedge_{i \in I} \kappa_i {\to} \rho_i \leq_D \delta$. By induction, for all $i \in I$ there exist $\Gamma_{i,j}, \Delta_{i,j}$ for $j = 1, 2$, such that $e \models_{\mathcal{F}} \Gamma_{i,j}, \Delta_{i,j}$ and $\Gamma_{i,1} \vdash M : \delta_i {\times} \kappa_i {\to} \rho_i \mid \Delta_{i,1}$ and $\Gamma_{i,2} \vdash N : \delta_i \mid \Delta_{i,2}$. Take $\Gamma = \wedge_{i,j} \Gamma_{i,j}$ and $\Delta = \wedge_{i,j} \Delta_{i,j}$, then $\Gamma \leq_D \Gamma_{i,j}$ and $\Delta \leq_C \Delta_{i,j}$ so that $e \models_{\mathcal{F}} \Gamma; \Delta$. By applying rule (St), for all $i \in I$ we have $\Gamma \vdash M : \delta_i {\times} \kappa_i {\to} \rho_i \mid \Delta$ and $\Gamma \vdash N : \delta_i \mid \Delta$. By applying rule

(App) we have $\Gamma \vdash MN : \kappa_i \to \rho_i \mid \Delta$; we obtain $\Gamma \vdash MN : \wedge_{i\in I} \kappa_i \to \rho_i \mid \Delta$ by applying rule $(\wedge)$, and $\Gamma \vdash MN : \delta \mid \Delta$ by applying rule $(\leq)$.

Vice-versa, assume $\Gamma \vdash MN : \delta \mid \Delta$ and $e \models_{\mathcal{F}} \Gamma;\Delta$. By Lem. 4.9, there exist I and δ_i, κ_i, and ρ_i $(i \in I)$ such that $\Gamma \vdash M : \delta_i \times k_i \to \rho_i \mid \Delta$ and $\Gamma \vdash N : \delta_i \mid \Delta$ and $\wedge_{i\in I} \kappa_i \to \rho_i \leq_D \delta$. For all $i \in I$, by induction we have $\delta_i \times k_i \to \rho_i \in [\![M]\!]^{\mathcal{F}} e$ and $\delta_i \in [\![N]\!]^{\mathcal{F}} e$, and so, by Lem. 4.17, $k_i \to \rho_i \in [\![MN]\!]^{\mathcal{F}} e$. Since $[\![MN]\!]^{\mathcal{F}} e$ is a filter, we have $\wedge_{i\in I} \kappa_i \to \rho_i \in [\![MN]\!]^{\mathcal{F}} e$ and therefore $\delta \in [\![MN]\!]^{\mathcal{F}} e$.

$(T \equiv \mu\alpha.\mathtt{C})$: If $\delta \in [\![\mu\alpha.\mathtt{C}]\!]^{\mathcal{F}} e$ then, by Lem. 4.17, there exist I and κ_i, ρ_i, and κ'_i $(i \in I)$ such that $(\kappa'_i \to \rho_i) \times \kappa'_i \in [\![\mathtt{C}]\!]^{\mathcal{F}} e[\alpha \mapsto (\uparrow_C \kappa_i)]$. For all $i \in I$, by induction there exist Γ_i and Δ_i such that $e[\alpha \mapsto (\uparrow_C \kappa_i)] \models_{\mathcal{F}} \Gamma_i;\Delta_i$ and $\Gamma_i \vdash \mathtt{C} : (\kappa'_i \to \rho_i) \times \kappa'_i \mid \Delta_i$. Let $\Gamma = \wedge_{i\in I} \Gamma_i$ and, for all $i \in I$, $\kappa_i = \Delta_i(\alpha)$ and $\Delta'_i = \Delta_i \backslash \alpha$, then $\Gamma \vdash \mathtt{C} : (\kappa'_i \to \rho_i) \times \kappa'_i \mid \alpha{:}\kappa_i, \Delta'_i$ so that $\Gamma \vdash \mu\alpha.\mathtt{C} : \kappa_i \to \rho_i \mid \Delta'_i$. Take $\Delta = \wedge_{i\in I} \Delta'_i$; then $\Delta \leq_C \Delta'_i$ for all $i \in I$, so by applying rule (St) we obtain $\Gamma \vdash \mu\alpha.\mathtt{C} : \kappa_i \to \rho_i \mid \Delta$, from which we derive $\Gamma \vdash \mu\alpha.\mathtt{C} : \delta \mid \Delta$ by applying rules $(\wedge)$ and $(\leq)$. On the other hand, for all $i \in I$, since $\alpha \notin dom(\Delta'_i)$ we have $e[\alpha \mapsto (\uparrow_C \kappa_i)] \models_{\mathcal{F}} \Gamma;\Delta_i$ which implies $e \models_{\mathcal{F}} \Gamma;\Delta'_i$, so that $\Delta(\alpha) = \omega$. We conclude that $e \models_{\mathcal{F}} \Gamma;\Delta$, as desired.

Vice-versa, assume $\Gamma \vdash \mu\alpha.\mathtt{C} : \delta \mid \Delta$ and $e \models_{\mathcal{F}} \Gamma;\Delta$. Then by Lem. 4.9, there exists I and κ_i, ρ_i, and κ'_i $(i \in I)$ such that $\Gamma \vdash \mathtt{C} : (\kappa_i \to \rho_i) \times \kappa_i \mid \alpha{:}\kappa'_i, \Delta$, and $\wedge_{i\in I} \kappa'_i \to \rho_i \leq_D \delta$. But if $e \models_{\mathcal{F}} \Gamma;\Delta$ then $e[\alpha \mapsto (\uparrow_C \kappa_i)] \models_{\mathcal{F}} \Gamma;\alpha{:}\kappa'_i,\Delta$; then by induction, for all $i \in I$, $(\kappa_i \to \rho_i) \times \kappa_i \in [\![\mathtt{C}]\!]^{\mathcal{F}} e[\alpha \mapsto (\uparrow_C \kappa_i)]$. Since $\wedge_{i\in I} \kappa'_i \to \rho_i \leq_D \delta$, by Lem. 4.17, we conclude that $\delta \in [\![\mu\alpha.\mathtt{C}]\!]^{\mathcal{F}} e$.

$(T \equiv [\alpha]M)$: If $\kappa \in [\![[\alpha]M]\!]^{\mathcal{F}} e$ then by Lem. 4.17, there exist I and δ_i, κ_i, and $\delta_i \in [\![M]\!]^{\mathcal{F}} e$ $(i \in I)$ such that $\kappa_i \in e(\alpha)$ and $\wedge_{i\in I} \delta_i \times \kappa_i \leq_C \kappa$. For all $i \in I$, by induction there exist Γ_i, Δ_i such that $e \models_{\mathcal{F}} \Gamma_i;\Delta_i$ and $\Delta_i(\alpha) \leq_C \kappa_i$ and $\Gamma_i \vdash M : \delta_i \mid \Delta_i$. Let $\Gamma = \wedge_{i\in I} \Gamma_i$. Then, for all $i \in I$, $\Gamma \vdash [\alpha]M : \delta_i \times \kappa_i \mid \Delta_i$; take $\Delta = \wedge_{i\in I} \Delta_i$, then for all $i \in I$, by applying rule (St), we obtain $\Gamma \vdash [\alpha]M : \delta_i \times \kappa_i \mid \Delta$. We obtain $\Gamma \vdash [\alpha]M : \wedge_{i\in I} \delta_i \times \kappa_i \mid \Delta$ by applying rule $(\wedge)$ and then $\Gamma \vdash [\alpha]M : \kappa \mid \Delta$ by applying rule $(\leq)$. Since $e \models_{\mathcal{F}} \Gamma;\Delta_i$ and $\Delta \leq_C \Delta_i$ for all $i \in I$, we conclude that $e \models_{\mathcal{F}} \Gamma;\Delta$.

Vice-versa, if $\Gamma \vdash [\alpha]M : \kappa \mid \Delta$ and $e \models_{\mathcal{F}} \Gamma;\Delta$ then, by Lem. 4.9, there exists I and δ_i and κ_i $(i \in I)$ such that $\Gamma \vdash M : \delta_i \mid \Delta$ and $\Delta(\alpha) \leq_C \kappa_i$, and $\wedge_{i\in I} \delta_i \times \kappa_i \leq_C \kappa$. By induction $\delta_i \in [\![M]\!]^{\mathcal{F}} e$; from $e \models_{\mathcal{F}} \Gamma;\Delta$ we have that $\kappa_i \in e(\alpha)$ for all $i \in I$. Then $\wedge_{i\in I} \delta_i \times \kappa_i \in [\![[\alpha]M]\!]^{\mathcal{F}} e$ and therefore that $\kappa \in [\![[\alpha]M]\!]^{\mathcal{F}} e$ since the last set is a filter. □

Definition 4.20 Given a basis Γ and a context Δ, we define the environment $e_{\Gamma;\Delta} \in \textsc{Env}_{\mathcal{F}}$ by:

$$\begin{array}{rcl} e_{\Gamma;\Delta}(x) & \triangleq & \uparrow_D \Gamma(x) \\ e_{\Gamma;\Delta}(\alpha) & \triangleq & \uparrow_D \Delta(\alpha) \end{array}$$

Because of the definition of $\Gamma(x)$ and $\Delta(\alpha)$ for $x \in \textsc{Var}$ and $\alpha \in \textsc{Name}$, $e_{\Gamma;\Delta} \in \textsc{Env}_{\mathcal{F}}$ implies:

$$e_{\Gamma;\Delta}(x) = \begin{cases} \uparrow_D \delta & (\text{if } x{:}\delta \in \Gamma) \\ \uparrow_D \omega & (\text{otherwise}) \end{cases}$$

$$e_{\Gamma;\Delta}(\alpha) = \begin{cases} \uparrow_C \kappa & (\text{if } \alpha{:}\kappa \in \Delta) \\ \uparrow_C \omega & (\text{otherwise}) \end{cases}$$

For this environment, we can show:

Lemma 4.21 If $e_{\Gamma;\Delta} \models_{\mathcal{F}} \Gamma';\Delta'$, then $\Gamma \leq_D \Gamma'$ and $\Delta \leq_C \Delta'$.

Proof By definition, if $e_{\Gamma;\Delta} \models_{\mathcal{F}} \Gamma',\Delta'$ then:

$$\begin{array}{rcl} e_{\Gamma;\Delta}(x) & = & \uparrow_D \Gamma(x) \ \in \ [\![\Gamma'(x)]\!]^{\mathcal{F}} \\ e_{\Gamma;\Delta}(\alpha) & = & \uparrow_C \Delta(\alpha) \ \in \ [\![\Delta'(\alpha)]\!]^{\mathcal{F}} \end{array}$$

for all $x \in \textsc{Var}$ and $\alpha \in \textsc{Name}$. From Lem. 4.18 (*ii*) we know that $\uparrow_D \Gamma(x) \in [\![\Gamma'(x)]\!]^{\mathcal{F}}$ if and only if $\Gamma'(x) \in \uparrow_D \Gamma(x)$, that is $\Gamma(x) \leq_D \Gamma'(x)$. Similarly $\uparrow_C \Delta(\alpha) \in [\![\Delta'(\alpha)]\!]^{\mathcal{F}}$ if and only if $\Delta(\alpha) \leq_C \Delta'(\alpha)$. Hence $\Gamma \leq_D \Gamma'$ and $\Delta \leq_C \Delta'$. □

This last result implies that every type which is an element of the interpretation of a term is derivable for that term:

Lemma 4.22 If $\delta \in [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta}$, then $\Gamma \vdash M : \delta \mid \Delta$.

Proof
$$\begin{array}{rcll} \delta \in [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta} & \Rightarrow & \exists \Gamma', \Delta' \, [e_{\Gamma;\Delta} \models \Gamma'; \Delta' \ \& \ \Gamma' \vdash M : \delta \mid \Delta'] & \text{(Thm. 4.19)} \\ & \Rightarrow & \exists \Gamma', \Delta' \, [\Gamma \leq_D \Gamma' \ \& \ \Delta \leq_C \Delta' \ \& \ \Gamma' \vdash M : \delta \mid \Delta'] & \text{(Lem. 4.21)} \\ & \Rightarrow & \Gamma \vdash M : \delta \mid \Delta & \text{(rule } (St)\text{, Lem. 4.7)} \end{array}$$ □

This lemma would not hold without case (Cmd_2) of rule (Cmd). As explained in Rem. 4.4 we should require that $\Delta \leq_C \Delta'$ implies $dom(\Delta) \supseteq dom(\Delta')$, which is not always the case.

We can now prove the completeness theorem for our type assignment system.

Theorem 4.23 (Completeness) *Let $\mathcal{M} = (R, D, C)$ be a $\lambda\mu$-model. If $\Gamma \models_{\mathcal{M}} M : \delta \mid \Delta$, then $\Gamma \vdash M : \delta \mid \Delta$.*

Proof Let $\Gamma \models_{\mathcal{M}} M : \delta \mid \Delta$: since $\mathcal{M}$ is isomorphic to the filter model $\mathcal{F} = (\mathcal{F}_R, \mathcal{F}_D, \mathcal{F}_C)$, we have that $\Gamma \models_{\mathcal{F}} M : \delta \mid \Delta$. By construction, $e_{\Gamma;\Delta} \models_{\mathcal{F}} \Gamma; \Delta$, and therefore:

$$\begin{array}{rcll} \Gamma \models_{\mathcal{F}} M : \delta \mid \Delta & \Rightarrow & [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta} \in [\![\delta]\!]^{\mathcal{F}} & \text{(Def. 4.14)} \\ & \Rightarrow & \delta \in [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta} & \text{(Lem. 4.18 (}ii\text{))} \\ & \Rightarrow & \Gamma \vdash M : \delta \mid \Delta & \text{(Lem. 4.22)} \end{array}$$ □

5 Closure under conversion

In this section, we will show that our notion of type assignment is closed under conversion, *i.e.* is closed both under subject reduction and expansion. We will first show that this follows from the semantical results we have established in the previous section; then we show the same result via a syntactical proof. The latter is more informative about the structure of derivations in our system; also we establish the term substitution and, more importantly, the structural substitution lemmas (Lem. 5.2 and Lem. 5.3 respectively).

We begin with the abstract proof, which crucially depends on Lem. 4.22 and hence on Thm. 4.19.

Theorem 5.1 (Closure under conversion) *Let $M =_{\beta\mu} N$. If $\Gamma \vdash M : \delta \mid \Delta$, then $\Gamma \vdash N : \delta \mid \Delta$.*

Proof By Thm. 2.7, if $M =_{\beta\mu} N$ then $[\![M]\!]^{\mathcal{M}} e = [\![N]\!]^{\mathcal{M}} e$ for any model $\mathcal{M}$ and environment $e \in \textsc{Env}_{\mathcal{M}}$, which holds in particular for $\mathcal{F}$ and $e_{\Gamma;\Delta}$. So

$$\begin{array}{rcll}
\Gamma \vdash M : \delta \mid \Delta & \Rightarrow & \Gamma \models_{\mathcal{F}} M : \delta \mid \Delta & \text{(Thm. 4.16)} \\
& \Rightarrow & [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta} \in [\![\delta]\!]^{\mathcal{F}} & \text{(since } e_{\Gamma;\Delta} \models_{\mathcal{F}} \Gamma;\Delta\text{)} \\
& \Rightarrow & \delta \in [\![M]\!]^{\mathcal{F}} e_{\Gamma;\Delta} & \text{(Lem. 4.18 (}ii\text{))} \\
& \Rightarrow & \delta \in [\![N]\!]^{\mathcal{F}} e_{\Gamma;\Delta} & \text{(Thm. 2.7)} \\
& \Rightarrow & \Gamma \vdash N : \delta \mid \Delta & \text{(Lem. 4.22)}
\end{array}$$ □

To illustrate the type assignment system itself, we will now show a more 'operational' proof for the same property, by studying how reductions and expansions of the term in the conclusion (the 'subject' of the typing judgement) are reflected by transformations of its typing derivations. First we show that type assignment is closed for preforming or reversing both name and term substitution.

Lemma 5.2 (Term substitution lemma) *$\Gamma \vdash T[L/x] : \sigma \mid \Delta$ with $x \notin \Gamma$ if and only if there exists δ' such that $\Gamma, x{:}\delta' \vdash T : \sigma \mid \Delta$ and $\Gamma \vdash L : \delta' \mid \Delta$.*

Proof By induction on the definition of term substitution.

$(T \equiv x)$: Then $x[L/x] \equiv L$ and $\sigma = \delta \in \mathcal{L}_D$.

$(\Rightarrow)$: If $\Gamma \vdash x[L/x] : \delta \mid \Delta$, then $\Gamma, x{:}\delta \vdash x : \delta \mid \Delta$ by (Ax) and $\Gamma \vdash L : \delta \mid \Delta$ by assumption.

$(\Leftarrow)$: If $\Gamma, x{:}\delta' \vdash x : \delta \mid \Delta$, then $\delta' \leq_D \delta$ by Cor. 4.9. From $\Gamma \vdash L : \delta' \mid \Delta$ and rule $(\leq)$, we have $\Gamma \vdash L : \delta \mid \Delta$, so also $\Gamma \vdash x[L/x] : \delta \mid \Delta$.

$(T \equiv y \neq x)$: Then $y[L/x] \equiv y$ and $\sigma = \delta \in \mathcal{L}_D$.

$(\Rightarrow)$: By rule (Wk), $\Gamma, x{:}\omega \vdash y : \delta \mid \Delta$, and $\Gamma \vdash L : \omega \mid \Delta$ by rule (ω).

$(\Leftarrow)$: By rule (Th), since $x \notin fv(y)$.

$(T \equiv \lambda y.N)$: Let $\sigma = \delta'' \times \kappa \to \rho \in \mathcal{L}_D$.

$$\begin{array}{rcll}
& & \exists \delta' \, [\, \Gamma, x{:}\delta' \vdash \lambda y.N : \delta'' \times \kappa \to \rho \mid \Delta \;\&\; \Gamma \vdash L : \delta' \mid \Delta \,] & \text{(Cor. 4.9)} \\
& \Leftrightarrow & \exists \delta' \, [\, \Gamma, x{:}\delta', y{:}\delta'' \vdash N : \kappa \to \rho \mid \Delta \;\&\; \Gamma \vdash L : \delta' \mid \Delta \,] & \text{(by induction)} \\
& \Leftrightarrow & \Gamma, y{:}\delta'' \vdash N[L/x] : \kappa \to \rho \mid \Delta & \text{(Cor. 4.9)} \\
& \Leftrightarrow & \Gamma \vdash \lambda y.(N[L/x]) : \delta'' \times \kappa \to \rho \mid \Delta & \\
& \triangleq & \Gamma \vdash (\lambda y.N)[L/x] : \delta'' \times \kappa \to \rho \mid \Delta &
\end{array}$$

$(T \equiv PQ)$: Let $\sigma = \kappa \to \rho \in \mathcal{L}_D$; notice that $(PQ)[L/x] \equiv P[L/x]\; Q[L/x]$.

$(\Rightarrow)$: By Cor. 4.9, there exist δ'' such that both $\Gamma \vdash P[L/x] : \delta'' \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash Q[L/x] : \delta'' \mid \Delta$. Then by induction there exist δ_1, δ_2 such that:

- $\Gamma, x{:}\delta_1 \vdash P : \delta'' \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash L : \delta_1 \mid \Delta$, as well as
- $\Gamma, x{:}\delta_2 \vdash Q : \delta'' \mid \Delta$ and $\Gamma \vdash L : \delta_2 \mid \Delta$.

Take $\delta' = \delta_1 \wedge \delta_2$; then by rules (St) and (App) we get $\Gamma, x{:}\delta' \vdash PQ : \kappa \to \rho \mid \Delta$ and $\Gamma \vdash L : \delta' \mid \Delta$ by rule $(\wedge I)$.

$(\Leftarrow)$: If $\Gamma, x{:}\delta' \vdash PQ : \kappa \to \rho \mid \Delta$, then, by Cor. 4.9 there exist δ'' such that both $\Gamma, x{:}\delta' \vdash P : \delta'' \times \kappa \to \rho \mid \Delta$ and $\Gamma, x{:}\delta'' \vdash Q : \delta' \mid \Delta$. Then, by induction, $\Gamma \vdash P[L/x] : \delta'' \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash Q[L/x] : \delta'' \mid \Delta$; the result follows by rule (App).

$(T \equiv \mu\alpha.\mathtt{C})$: Let $\sigma = \kappa \to \rho \in \mathcal{L}_D$; notice that $(\mu\alpha.\mathtt{C})[L/x] \equiv \mu\alpha.\mathtt{C}[L/x]$. Now:

$$\begin{array}{rcll}
& & \exists \delta' \, [\, \Gamma, x{:}\delta' \vdash \mu\alpha.\mathtt{C} : \kappa \to \rho \mid \Delta \;\&\; \Gamma \vdash L : \delta' \mid \Delta \,] & \\
& \Leftrightarrow & \exists \delta', \kappa' \, [\, \Gamma, x{:}\delta' \vdash \mathtt{C} : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta \;\&\; \Gamma \vdash L : \delta' \mid \Delta \,] & \text{(Cor. 4.9)} \\
& \Leftrightarrow & \exists \kappa' \, [\, \Gamma \vdash \mathtt{C}[L/x] : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta \,] & \text{(by induction)} \\
& \Leftrightarrow & \Gamma \vdash \mu\alpha.\mathtt{C}[L/x] : \kappa \to \rho \mid \Delta & \text{(Cor. 4.9)}
\end{array}$$

$(T \equiv [\alpha]N)$: Let $\sigma = \delta{\times}\kappa$, and $\Delta = \alpha{:}\kappa, \Delta'$. Now:

$$\begin{array}{lll} \exists \delta' \, [\, \Gamma, x{:}\delta' \vdash [\alpha]N : \delta{\times}\kappa \mid \alpha{:}\kappa, \Delta' \;\&\; \Gamma \vdash L : \delta' \mid \alpha{:}\kappa, \Delta'\,] & & \\ \quad \Longleftrightarrow \;\; \exists \delta', \kappa' \, [\, \Gamma, x{:}\delta' \vdash N : \delta \mid \alpha{:}\kappa, \Delta' \;\&\; \Gamma \vdash L : \delta' \mid \Delta\,] & \text{(Cor. 4.9)} \\ \quad \Longleftrightarrow \;\; \exists \kappa' \, [\, \Gamma \vdash N[L/x] : \delta \mid \alpha{:}\kappa, \Delta'\,] & \text{(by induction)} \\ \quad \Longleftrightarrow \;\; \Gamma \vdash [\alpha]N[L/x] : \delta{\times}\kappa \mid \alpha{:}\kappa, \Delta' & \text{(Cor. 4.9)} \end{array}$$

Notice that $([\alpha]N)[L/x] \equiv [\alpha]N[L/x]$. □

The next lemma states how the structural substitution $T[\alpha \Leftarrow L]$ is related to the type of the name α. When $T \in \text{Trm}$ or $T \equiv [\beta]N$ with $\alpha \neq \beta$, the type of $T[\alpha \Leftarrow L]$ remains the same as that of T, which is similar to the term substitution lemma, but the context Δ used to type $T[\alpha \Leftarrow L]$ changes to Δ' which is equal to Δ but for $\Delta'(\alpha) = \delta'{\times}\Delta(\alpha)$, where δ' is a type of L (in the same basis and context). When $T \equiv [\alpha]N$ the effect of the structural substitution is more complex, and it affects also the type of T with respect to that of $T[\alpha \Leftarrow L]$. The fact that this does not invalidate the type invariance of terms with respect to structural substitution is due to the form of rule (μ) which essentially is a cut rule: indeed, the 'cut type' changes in case of T with respect to that of $T[\alpha \Leftarrow L]$, but then is hidden in the conclusion.

Lemma 5.3 (Structural substitution lemma) *Let $M, N, L \in \text{Trm}$ and $\alpha, \beta \in \text{Name}$ with $\alpha \neq \beta$, and assume that $\alpha \notin fn(L)$; then:*

i) $\Gamma \vdash M[\alpha \Leftarrow L] : \delta \mid \alpha{:}\kappa, \Delta$ *if and only if there exists* δ' *such that* $\Gamma \vdash L : \delta' \mid \Delta$ *and* $\Gamma \vdash M : \delta \mid \alpha{:}\delta'{\times}\kappa, \Delta$.

ii) $\Gamma \vdash ([\beta]N)[\alpha \Leftarrow L] : \kappa' \mid \alpha{:}k, \Delta$ *with* $\alpha \neq \beta$, *if and only if there exists* δ' *such that* $\Gamma \vdash L : \delta' \mid \Delta$ *and* $\Gamma \vdash [\beta]N : \kappa' \mid \alpha{:}\delta'{\times}\kappa, \Delta$.

iii) $\Gamma \vdash ([\alpha]N)[\alpha \Leftarrow L] : (\kappa_1{\to}\rho){\times}\kappa_2 \mid \alpha{:}\kappa_2, \Delta$ *if and only if there exists* δ' *such that* $\Gamma \vdash L : \delta' \mid \Delta$ *and* $\Gamma \vdash [\alpha]N : (\delta'{\times}\kappa_1{\to}\rho){\times}\delta'{\times}\kappa_2 \mid \alpha{:}\delta'{\times}\kappa_2, \Delta$.

Proof By simultaneous induction on the definition of structural substitution. Observe that whenever $\alpha \notin fn(T)$ for $T \equiv M$ in part (*i*) and $T \equiv ([\beta]N)$ in part (*ii*), we have that $T[\alpha \Leftarrow L] \equiv T$, so that the lemma is vacuously true by taking $\delta' = \omega$. We only show the relevant cases.

i) $(\Rightarrow)$: We can assume the last step in $\Gamma \vdash (\mu\gamma.\mathsf{C})[\alpha \Leftarrow L] : \delta \mid \alpha{:}\kappa, \Delta$ is rule (μ); then $\delta = \kappa_1{\to}\rho$ and $\Gamma \vdash \mathsf{C}[\alpha \Leftarrow L] : (\kappa_2{\to}\rho){\times}\kappa_2 \mid \gamma{:}\kappa_1, \alpha{:}\kappa, \Delta$. We now distinguish two cases:

- $(\mathsf{C} \equiv [\beta]N$ *with* $\alpha \neq \beta)$: Then *IH* (*ii*) applies, so there exists δ' such that $\Gamma \vdash L : \delta' \mid \gamma{:}\kappa_1, \Delta$ and $\Gamma \vdash [\beta]N : (\kappa_2{\to}\rho){\times}\kappa_2 \mid \gamma{:}\kappa_1, \alpha{:}\delta'{\times}\kappa, \Delta$. Then $\Gamma \vdash \mu\gamma.[\beta]N : \kappa_1{\to}\rho \mid \alpha{:}\delta'{\times}\kappa, \Delta$ follows by applying (μ). For the second part, since γ is bound in $\mu\gamma.[\beta]N$, by Barendregt's convention $\gamma \notin fn(L)$, so applying rule (Th) to $\Gamma \vdash L : \delta' \mid \gamma{:}\kappa_1, \Delta$ gives $\Gamma \vdash L : \delta' \mid \Delta$.
- $(\mathsf{C} \equiv [\alpha]N)$: Now *IH* (*iii*) applies, so there exists δ' such that $\Gamma \vdash L : \delta' \mid \gamma{:}\kappa_1, \Delta$ and $\Gamma \vdash [\alpha]N : (\delta'{\times}\kappa{\to}\rho){\times}(\delta'{\times}\kappa) \mid \gamma{:}\kappa_1, \alpha{:}\delta'{\times}\kappa, \Delta$. Then by applying rule (μ) we get $\Gamma \vdash \mu\gamma.[\alpha]N : \kappa_1{\to}\rho \mid \alpha{:}\delta'{\times}\kappa, \Delta$, and $\Gamma \vdash L : \delta' \mid \Delta$ by rule (Th).

$(\Leftarrow)$: We can assume $\Gamma \vdash \mu\alpha.[\alpha]N : \kappa_1{\to}\rho \mid \alpha{:}\delta'{\times}\kappa, \Delta$ ends with rule (μ); then $\Gamma \vdash [\alpha]N : (\kappa_2{\to}\rho){\times}\kappa_2 \mid \gamma{:}\kappa_1, \alpha{:}\delta'{\times}\kappa, \Delta$, where $\kappa_2 = \delta'{\times}\kappa$, so by *IH* (*iii*) we get the result. The case $\mu\alpha.[\beta]N$ with $\alpha \neq \beta$ is similar and easier.

ii) Note that $([\beta]N)[\alpha \Leftarrow L] \equiv [\beta](N[\alpha \Leftarrow L])$.

$(\Rightarrow)$: We can assume the derivation for $\Gamma \vdash [\beta](N[\alpha \Leftarrow L]) : \kappa' \mid \alpha{:}k, \Delta$ ends with rule (Cmd); then $\kappa' = \delta''{\times}\kappa''$, $\Delta = \beta{:}\kappa'', \Delta'$, and $\Gamma \vdash N[\alpha \Leftarrow L] : \delta'' \mid \alpha{:}k, \beta{:}\kappa'', \Delta$. Then by *IH* (*i*), there

exists δ' such that $\Gamma \vdash L : \delta' \mid \Delta$ and $\Gamma \vdash N : \delta'' \mid \alpha{:}\delta' \times \kappa, \beta{:}\kappa'', \Delta'$. By rule (Cmd) we get $\Gamma \vdash [\beta]N : \kappa' \mid \alpha{:}\delta' \times \kappa, \Delta$.

$(\Leftarrow)$: The reasoning for this part is the reverse of part $(\Rightarrow)$.

iii) We consider the two implications; note that $([\alpha]N)[\alpha \Leftarrow L] \equiv [\alpha](N[\alpha \Leftarrow L])L$.

$(\Rightarrow)$: We can assume the derivation ends with (Cmd); then $\mathcal{D}' :: \Gamma \vdash (N[\alpha \Leftarrow L])L : \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta$. Likewise, we can assume $\mathcal{D}'$ ends with rule (App):

$$\dfrac{\begin{array}{cc}\boxed{\mathcal{D}'_1} & \boxed{\mathcal{D}'_2} \\ \Gamma \vdash N[\alpha \Leftarrow L] : \delta_1 \times \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta & \Gamma \vdash L : \delta_1 \mid \alpha{:}\kappa_2, \Delta\end{array}}{\Gamma \vdash (N[\alpha \Leftarrow L])L : \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta}\ (App)$$

Since $\alpha \notin fn(L)$, $\Gamma \vdash L : \delta_1 \mid \Delta$ is derivable by applying rule (Th) to $\mathcal{D}'_2$. By $IH\,(i)$, from $\mathcal{D}'_1$ there exists δ_2 such that $\Gamma \vdash L : \delta_2 \mid \Delta$ and $\Gamma \vdash N : \delta_1 \times \kappa_1 \to \rho \mid \alpha{:}\delta_2 \times \kappa_2, \Delta$. Then also $\Gamma \vdash L : \delta_1 \wedge \delta_2 \mid \Delta$ by rule $(\wedge)$. Taking $\delta' = \delta_1 \wedge \delta_2$, we have

$$\delta' \times \kappa_2 \leq_C \delta_2 \times \kappa_2 \quad \text{and} \quad \delta_1 \times \kappa_1 \to \rho \leq_D \delta' \times \kappa_1 \to \rho.$$

Then we can construct:

$$\dfrac{\dfrac{\dfrac{\begin{array}{cc}\boxed{\phantom{\mathcal{D}'_1}} & \boxed{\phantom{\mathcal{D}'_2}} \\ \Gamma \vdash N : \delta_1 \times \kappa_1 \to \rho \mid \alpha{:}\delta_2 \times \kappa_2, \Delta & \delta_1 \times \kappa_1 \to \rho \leq_D \delta' \times \kappa_1 \to \rho\end{array}}{\Gamma \vdash N : \delta' \times \kappa_1 \to \rho \mid \alpha{:}\delta_2 \times \kappa_2, \Delta}\ (\leq)}{\Gamma \vdash N : \delta' \times \kappa_1 \to \rho \mid \alpha{:}\delta' \times \kappa_2, \Delta}\ (St)}{\Gamma \vdash [\alpha]N : (\delta' \times \kappa_1 \to \rho) \times \delta' \times \kappa_2 \mid \alpha{:}\delta' \times \kappa_2, \Delta}\ (Cmd)$$

$(\Leftarrow)$: Again, we can assume the derivation ends with rule (Cmd), then $\mathcal{D}' :: \Gamma \vdash N : \delta' \times \kappa_1 \to \rho \mid \alpha{:}\delta' \times \kappa_2, \Delta$. Then $\Gamma \vdash N[\alpha \Leftarrow L] : \delta' \times \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta$ by $IH\,(i)$. Since $\alpha \notin fn(L)$ and (as we can assume) $\alpha \notin dom(\Delta)$, we have $\Gamma \vdash L : \delta' \mid \alpha{:}\kappa_2, \Delta$ by (Wk). We can construct:

$$\dfrac{\dfrac{\dfrac{\begin{array}{cc}\boxed{\phantom{\mathcal{D}'_1}} & \boxed{\phantom{\mathcal{D}'_2}} \\ \Gamma \vdash N[\alpha \Leftarrow L] : \delta' \times \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta & \Gamma \vdash L : \delta' \mid \alpha{:}\kappa_2, \Delta\end{array}}{\Gamma \vdash (N[\alpha \Leftarrow L])L : \kappa_1 \to \rho \mid \alpha{:}\kappa_2, \Delta}\ (App)}{\Gamma \vdash [\alpha](N[\alpha \Leftarrow L])L : (\kappa_1 \to \rho) \times \kappa_2 \mid \alpha{:}\kappa_2, \Delta}\ (Cmd)$$

□

We will now show that types are preserved under expansion, the opposite of reduction.

Theorem 5.4 (Subject expansion) *If $M \to_{\beta\mu} N$, and $\Gamma \vdash N : \delta \mid \Delta$, then $\Gamma \vdash M : \delta \mid \Delta$.*

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

$((\lambda x.M)N \to M[N/x])$: If $\Gamma \vdash M[N/x] : \delta \mid \Delta$, then by Lem. 5.2 there exists a δ' such that $\Gamma, x{:}\delta' \vdash M : \delta \mid \Delta$ and $\Gamma \vdash N : \delta' \mid \Delta$; assume (without loss of generality) that $\delta = \kappa \to \rho$, then, by applying rule (Abs) to the first result we get $\Gamma \vdash \lambda x.M : \delta' \times \kappa \to \rho \mid \Delta$ and by (App) we get $\Gamma \vdash (\lambda x.M)N : \delta \mid \Delta$.

$((\mu\alpha.\mathtt{C})N \to \mu\alpha.\mathtt{C}[\alpha \Leftarrow N])$: We can assume rule (μ) was applied last for $\Gamma \vdash \mu\alpha.\mathtt{C}[\alpha \Leftarrow N] : \delta \mid \Delta$, then $\delta = \kappa \to \rho$ and there exists κ' such that $\Gamma \vdash \mathtt{C}[\alpha \Leftarrow N] : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta$. We now distinguish the following cases of $\mathtt{C}$:

($\mathtt{C} \equiv [\beta]L$, *with* $\beta \neq \alpha$): Then, by Lem. 5.3 (*ii*), there exists δ' such that $\Gamma \vdash N : \delta' \mid \Delta$, and $\Gamma \vdash [\beta]L : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\delta' \times \kappa, \Delta$. Then, by rule (μ), $\Gamma \vdash \mu\alpha.[\beta]L : \delta' \times \kappa \to \rho \mid \Delta$, and $\Gamma \vdash (\mu\alpha.[\beta]L)N : \kappa \to \rho \mid \Delta$ follows by rule (App).

($\mathtt{C} \equiv [\alpha]L$): Then, by Lem. 5.3 (*iii*), there exists δ' such that $\Gamma \vdash N : \delta' \mid \Delta$, and $\mathcal{D} :: \Gamma \vdash [\alpha]L : (\delta' \times \kappa \to \rho) \times \delta' \times \kappa \mid \alpha{:}\delta' \times \kappa, \Delta$. We can assume the last rule of $\mathcal{D}$ is (Cmd), then $\Gamma \vdash L : \delta' \times \kappa \to \rho \mid \alpha{:}\delta' \times \kappa, \Delta$. Since α is bound in $\mu\alpha.[\alpha]L$ we can assume that $\alpha \notin dom(\Delta)$, so that we can construct:

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash L : \delta' \times \kappa \to \rho \mid \alpha{:}\delta' \times \kappa, \Delta}}}{\Gamma \vdash L : \delta' \times \kappa \to \rho \mid \alpha{:}\delta' \times \kappa, \Delta}\,(Cmd)}{\Gamma \vdash [\alpha]L : (\delta' \times \kappa \to \rho) \times (\delta' \times \kappa) \mid \alpha{:}\delta' \times \kappa, \Delta}\,(\mu) \qquad \boxed{}}{\Gamma \vdash \mu\alpha.[\alpha]L : \delta' \times \kappa \to \rho \mid \Delta \qquad\qquad \Gamma \vdash N : \delta' \mid \Delta}\,(App)$$
$$\Gamma \vdash (\mu\alpha.[\alpha]L)N : \kappa \to \rho \mid \Delta$$

($\mu\alpha[\beta]\mu\gamma.[\alpha']M \to \mu\alpha.[\alpha']M[\beta/\gamma]$): For $\mathcal{D} :: \Gamma \vdash \mu\alpha.[\alpha']M[\beta/\gamma] : \delta \mid \Delta$, assume there exist κ_1, κ_2, κ_3 such that $\delta = \kappa_2 \to \rho$, and that $\mathcal{D}$ is shaped like:

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash M[\beta/\gamma] : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}}}{\Gamma \vdash M[\beta/\gamma] : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(Cmd)}{\Gamma \vdash [\alpha']M[\beta/\gamma] : (\kappa_2 \to \rho) \times \kappa_2 \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(\mu)}{\Gamma \vdash \mu\alpha.[\alpha']M[\beta/\gamma] : \kappa_1 \to \rho \mid \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(\mu)$$

Since M can contain β as well, this means that there exist κ^1, κ^2 with $\kappa_3 = \kappa^1 \wedge \kappa^2$, such that κ^1 is an intersection of the types used for the 'original' β, and κ^2 for those inserted by the substitution. Then we have $\Gamma \vdash M : \kappa \mid \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa^1, \gamma{:}\kappa^2, \Delta$ as well, and, by weakening, also $\Gamma \vdash M : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \gamma{:}\kappa_3, \Delta$. We can now construct:

$$\dfrac{\dfrac{\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash M : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \gamma{:}\kappa_3, \Delta}}}{\Gamma \vdash M : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \gamma{:}\kappa_3, \Delta}\,(Cmd)}{\Gamma \vdash [\alpha']M : (\kappa_2 \to \rho) \times \kappa_2 \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \gamma{:}\kappa_3, \Delta}\,(\mu)}{\Gamma \vdash \mu\gamma.[\alpha']M : k_3 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(Cmd)}{\Gamma \vdash [\beta]\mu\gamma.[\alpha']M : (\kappa_3 \to \rho) \times \kappa_3 \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(\mu)}{\Gamma \vdash \mu\alpha.[\beta]\mu\gamma.[\alpha']M : \kappa_1 \to \rho \mid \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta}\,(\mu)$$

which establishes the result. □

We will now prove the counterpart of the previous theorem, and show that types are preserved under reduction.

Theorem 5.5 (Subject reduction) *If $M \to_{\beta\mu} N$, and $\Gamma \vdash M : \delta \mid \Delta$, then $\Gamma \vdash N : \delta \mid \Delta$.*

Proof By considering the three reduction rules.

($(\lambda x.M)N \to M[N/x]$): We can assume the derivation for $\Gamma \vdash (\lambda x.M)N : \delta \mid \Delta$ ends with rule (App), then $\delta = \kappa \to \rho$ and there exists δ' such that $\Gamma \vdash \lambda x.M : \delta' \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash N : \delta' \mid \Delta$. Likewise, we can assume the first ends with rule (Abs) then $\Gamma, x{:}\delta' \vdash M : \kappa \to \rho \mid \Delta$. Then, by Lem. 5.2, we have $\Gamma \vdash M[N/x] : \kappa \to \rho \mid \Delta$.

($(\mu\alpha.\mathtt{C})N \to \mu\alpha.\mathtt{C}[\alpha \Leftarrow N]$): We can assume the derivation for $\Gamma \vdash (\mu\alpha.\mathtt{C})N : \delta \mid \Delta$ ends with rule (App), then $\delta = \kappa \to \rho$ and there exists δ' such that $\Gamma \vdash \mu\alpha.\mathtt{C} : \delta' \times \kappa \to \rho \mid \Delta$ and $\Gamma \vdash N : \delta' \mid \Delta$. We can assume the first ends with rule with rule (μ), then there exists κ' such that $\Gamma \vdash \mathtt{C} : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\delta' \times \kappa, \Delta$. As in the proof of Thm. 5.4, we distinguish:

($\mathtt{C} \equiv [\beta]L$, *with* $\beta \neq \alpha$): Then, by Lem. 5.3 (*ii*), $\Gamma \vdash [\beta]L[\alpha \Leftarrow N] : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\kappa, \Delta$, and $\Gamma \vdash \mu\alpha.[\beta]L[\alpha \Leftarrow N] : \kappa \to \rho \mid \Delta$ follows by rule (μ).

($\mathtt{C} \equiv [\alpha]L$): We can assume the derivation for $\Gamma \vdash [\alpha]L : (\kappa' \to \rho) \times \kappa' \mid \alpha{:}\delta' \times \kappa, \Delta$ ends with (Cmd). Then $\kappa' = \delta' \times \kappa$, and $\Gamma \vdash ([\alpha]L)[\alpha \Leftarrow N] : (\kappa \to \rho) \times \kappa \mid \alpha{:}\kappa, \Delta$ by Lem. 5.3 (*iii*); we apply rule (μ) to derive $\Gamma \vdash \mu\alpha.([\alpha]L)[\alpha \Leftarrow N] : \kappa \to \rho \mid \Delta$. Notice that $\mu\alpha.([\alpha]L)[\alpha \Leftarrow N] \equiv (\mu\alpha.[\alpha])(L[\alpha \Leftarrow N])N$.

($\mu\alpha.[\beta]\mu\gamma.[\alpha']M \to \mu\alpha.[\alpha'](M[\beta/\gamma])$): If $\Gamma \vdash \mu\alpha.[\alpha']\mu\gamma.[\delta]M : \delta \mid \Delta$, then we can assume there exist $\rho, \kappa_1, \kappa_2, \kappa_3$ such that $\Gamma \vdash M : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \gamma{:}\kappa_3, \Delta$ and $\delta = \kappa \to \rho$. Then, obviously, also $\Gamma \vdash M[\beta/\gamma] : \kappa_2 \to \rho \mid \alpha{:}\kappa_1, \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta$, and applying rule $(\times)$ and (μ) to this derivation gives $\Gamma \vdash \mu\alpha.[\alpha'](M[\beta/\gamma]) : \delta \mid \alpha'{:}\kappa_2, \beta{:}\kappa_3, \Delta$. □

We end this section with two examples.

Example 5.6 As stated by the last results, we now show that we can assign to $(\lambda xyz.xz(yz))(\lambda ab.a)$ any type that is assignable to $\lambda ba.a$ since $(\lambda xyz.xz(yz))(\lambda ab.a) \to^* \lambda ba.a$. We first derive a type for $\lambda ba.a$.

$$\dfrac{\dfrac{\dfrac{}{a{:}\kappa \to \rho, b{:}\omega \vdash a : \kappa \to \rho \mid}\,(Ax)}{b{:}\omega \vdash \lambda a.a : (\kappa \to \rho) \times \kappa \to \rho \mid}\,(Abs)}{\vdash \lambda ba.a : \omega \times (\kappa \to \rho) \times \kappa \to \rho \mid}\,(Abs)$$

Let $\Gamma = x{:}(\kappa \to \rho) \times \omega \times \kappa \to \rho, y{:}\omega, z{:}\kappa \to \rho$, then we can derive:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : (\kappa \to \rho) \times \omega \times \kappa \to \rho \mid}\,(Ax) \quad \dfrac{}{\Gamma \vdash z : \kappa \to \rho \mid}\,(Ax)}{\Gamma \vdash xz : \omega \times \kappa \to \rho \mid}\,(App) \quad \dfrac{}{\Gamma \vdash yz : \omega \mid}\,(\omega)}{\Gamma \vdash xz(yz) : \kappa \to \rho \mid}\,(App)}{\Gamma \backslash z \vdash \lambda z.xz(yz) : (\kappa \to \rho) \times \kappa \to \rho \mid}\,(Abs)}{\Gamma \backslash z \backslash y \vdash \lambda yz.xz(yz) : \omega \times (\kappa \to \rho) \times \kappa \to \rho \mid}\,(Abs) \quad \dfrac{\dfrac{\dfrac{}{a{:}\kappa \to \rho, b{:}\omega \vdash a : \kappa \to \rho \mid}\,(Ax)}{a{:}\kappa \to \rho \vdash \lambda b.a : \omega \times \kappa \to \rho \mid}\,(Abs)}{\vdash \lambda ab.a : (\kappa \to \rho) \times \omega \times \kappa \to \rho \mid}\,(Abs)}{\vdash (\lambda xyz.xz(yz))(\lambda ab.a) : \omega \times (\kappa \to \rho) \times \kappa \to \rho \mid}\,(App)$$

Example 5.7 Consider the reduction $(\mu\alpha.[\alpha]x)x \to \mu\alpha.([\alpha]x)[\alpha \Leftarrow x] \equiv \mu\alpha.[\alpha]xx$. This last term is not a proof term in the sense of Parigot, but of interest here since typing the self application xx is a characteristic of intersection type systems. Let $\delta \in \mathcal{L}_D$ be arbitrary. Now we have:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \wedge (\delta \times \kappa \to \rho) \mid \alpha{:}\delta \times \kappa}\,(Ax)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \times \kappa \to \rho \mid \alpha{:}\delta \times \kappa}\,(\leq)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash [\alpha]x : (\delta \times \kappa \to \rho) \times (\delta \times \kappa) \mid \alpha{:}\delta \times \kappa}\,(Cmd)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash \mu\alpha.[\alpha]x : \delta \times \kappa \to \rho \mid}\,(\mu) \quad \dfrac{\dfrac{\dfrac{}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \wedge (\delta \times \kappa \to \rho) \mid \alpha{:}\delta \times \kappa}\,(Ax)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \mid \alpha{:}\delta \times \kappa}\,(\leq)}{\vdots}}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash (\mu\alpha.[\alpha]x)x : \kappa \to \rho \mid}\,(App)$$

as well as:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \wedge (\delta \times \kappa \to \rho) \mid \alpha{:}\kappa}\,(Ax)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \times \kappa \to \rho \mid \alpha{:}\kappa}\,(\leq) \quad \dfrac{\dfrac{}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \wedge (\delta \times \kappa \to \rho) \mid \alpha{:}\kappa}\,(Ax)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash x : \delta \mid \alpha{:}\kappa}\,(\leq)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash xx : \kappa \to \rho \mid \alpha{:}\kappa}\,(App)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash [\alpha]xx : (\kappa \to \rho) \times \kappa \mid \alpha{:}\kappa}\,(Cmd)}{x{:}\delta \wedge (\delta \times \kappa \to \rho) \vdash \mu\alpha.[\alpha]xx : \kappa \to \rho \mid}\,(\mu)$$

Observe that the 'cut type' in the first derivation, $\delta \times \kappa$ (appearing twice in the type $(\delta \times \kappa \to \rho) \times (\delta \times \kappa)$ of the premise of rule (μ)), differs from the cut type κ in $(\kappa \to \rho) \times \kappa$ occurring in the

premise of (μ) of the second derivation; indeed the latter is of a smaller size than the former.

A similar but simpler derivation can be obtained from the previous one in the case of the reduction $(\mu\alpha.[\alpha]x)y \to \mu\alpha.([\alpha]x)[\alpha \Leftarrow y] \equiv \mu\alpha.[\alpha]xy$, where there is no self-application: indeed we have that $x{:}\delta\times\kappa\to\rho, y{:}\delta \vdash (\mu\alpha.[\alpha]x)y : \kappa\to\rho \mid$ and $x{:}\delta\times\kappa\to\rho, y{:}\delta \vdash \mu\alpha.[\alpha]xy : \kappa\to\rho \mid$ are derivable in a very similar manner.

6 Characterisation of Strong Normalisation

One of the main results for $\lambda\mu$, proved in [47], states that all $\lambda\mu$-terms that correspond to proofs of second-order natural deduction are strongly normalising; the reverse of this property does not hold for Parigot's system, since there, for example, not all terms in normal form are typeable.

The full characterisation of strong normalisation (M is strongly normalising if and only if M is typeable in a given system) for the λ-calculus is a property that is shown for various intersection systems (see [18], Sect. 17.2 and the references there). So it is a natural question whether there exists a similar characterisation of strongly normalising $\lambda\mu$-terms in the present context, by suitably restricting the system in Sect. 4. We answer this question here; the proof is a revised version of that in [12], obtained by a simplified type syntax and by just restricting the full system, instead of considering one of its variants.

To simplify the technical treatment we shall ignore the structural reduction rule (ren); in fact the set of strongly formalisable terms remains the same, no matter whether this rule is considered or not.

We establish the relation between our result to the one for Parigot's system in the next section.

6.1 The restricted type system

For the untyped λ-calculus, the characterisation of strong normalisation states that a λ-term is strongly normalisable if and only if it is typeable in a restricted system of intersection types, where ω is not admitted as a type and consequently the rule (ω) is not part of the system. Alas a straightforward extension of this result does not hold for the $\lambda\mu$-calculus, at least with respect to the system presented in this paper. This is due to the fact that the natural interpretation of a type $\kappa = \delta_1\times\cdots\delta_k\times\omega$ (for $k > 0$) is the set of continuations whose leading k elements are in the denotations of $\delta_1,\ldots,\delta_k$; since continuations are infinite tuples, the ending ω represents the lack of information about the remaining infinite part. Therefore these occurrences of ω cannot be simply deleted without substantially changing the semantics of the type system and questioning the soundness of its rules.

We solve the problem of restricting the type assignment system to the extent of typing strongly normalising terms here by defining a particular subset of the type language. There the type ω is allowed only in certain harmless positions such that its meaning becomes just the universe of terms we are looking for, *i.e.* the strongly normalising ones. This amounts to restricting the sets of types $\mathcal{L}_D$ and $\mathcal{L}_C$ to those having ω only as the final part of product types. We shall then suitably modify the standard interpretation of intersection types, adapting Tait's computability argument.

For what concerns the atomic types, a single constant type v suffices for our purposes. Therefore restricted types are of two sorts instead of three.

Definition 6.1 (Restricted Types and Pre-order) *i*) The sets $\mathcal{L}^{\mathrm{R}}_D$ of *(restricted) term types* and $\mathcal{L}^{\mathrm{R}}_C$ of *(restricted) continuation types* are defined inductively by the following grammars, where υ is a type constant:

$$\begin{array}{lrcl} \mathcal{L}^{\mathrm{R}}_D: & \delta & ::= & \kappa\to\upsilon \mid \delta\wedge\delta \\ \mathcal{L}^{\mathrm{R}}_C: & \kappa & ::= & \omega \mid \delta\times\kappa \mid \kappa\wedge\kappa \end{array}$$

ii) We define the set $\mathcal{L}^{\mathrm{R}}$ of *(restricted) types* as $\mathcal{L}^{\mathrm{R}} = \mathcal{L}^{\mathrm{R}}_D \cup \mathcal{L}^{\mathrm{R}}_C$ and the relations $\leq^{\mathrm{R}}_A$ over $\mathcal{L}^{\mathrm{R}}_A$ (for $A = D,C$) as the pre-order induced by the least intersection type theories $\mathcal{T}^{\mathrm{R}}_A$ such that:

$$\frac{}{\kappa \leq^{\mathrm{R}}_C \omega} \qquad \frac{}{(\delta_1\times\kappa_1)\wedge(\delta_2\times\kappa_2) \leq^{\mathrm{R}}_C (\delta_1\wedge\delta_2)\times(\kappa_1\wedge\kappa_2)} \qquad \frac{\delta_1 \leq^{\mathrm{R}}_D \delta_2 \quad \kappa_1 \leq^{\mathrm{R}}_C \kappa_2}{\delta_1\times\kappa_1 \leq^{\mathrm{R}}_C \delta_2\times\kappa_2} \qquad \frac{\kappa_2 \leq^{\mathrm{R}}_C \kappa_1}{\kappa_1\to\upsilon \leq^{\mathrm{R}}_D \kappa_2\to\upsilon}$$

$$\frac{}{\sigma\wedge\tau \leq^{\mathrm{R}}_A \sigma} \qquad \frac{}{\sigma\wedge\tau \leq^{\mathrm{R}}_A \tau} \qquad \frac{\sigma \leq^{\mathrm{R}}_A \tau_1 \quad \sigma \leq^{\mathrm{R}}_A \tau_2}{\sigma \leq^{\mathrm{R}}_A \tau_1\wedge\tau_2} \qquad (A = D,C)$$

iii) We define the *length* of a continuation type, $|\cdot| : \mathcal{L}^{\mathrm{R}}_C \to \mathbb{N}$, as follows:

$$\begin{array}{rcl} |\omega| & = & 0 \\ |\delta\times\kappa| & = & 1 + |\kappa| \\ |\kappa_1\wedge\kappa_2| & = & max(\,|\kappa_1|\,,|\kappa_2|\,) \end{array}$$

By definition, we have that $\mathcal{L}^{\mathrm{R}}_D \subseteq \mathcal{L}_D$ and $\mathcal{L}^{\mathrm{R}}_C \subseteq \mathcal{L}_C$. All the rules axiomatising $\leq^{\mathrm{R}}_A$ are instances of the rules axiomatising $\leq_D$ and $\leq_C$ in Def. 3.9 and 3.13, hence $\leq^{\mathrm{R}}_A \subseteq \leq_A$ for $A = D,C$; in other words, the theories $\mathcal{T}_A$ can be seen as extensions of the respective theories $\mathcal{T}^{\mathrm{R}}_A$. It is natural to ask whether $\mathcal{T}_A$ (for $A = D,C$) is conservative with respect to $\mathcal{T}^{\mathrm{R}}_A$. This is not obvious: in a derivation of $\sigma \leq_A \tau$ in the formal theory $\mathcal{T}_A$, even if $\sigma,\tau \in \mathcal{L}^{\mathrm{R}}_A$, one could have used a type $\sigma' \notin \mathcal{L}^{\mathrm{R}}_A$ and the transitivity rule with premises $\sigma \leq_A \sigma' \leq_A \tau$, which cannot be derived in the formal presentation of $\leq^{\mathrm{R}}_A$. For example, consider the inequalities:

$$\delta_1\times\delta_2\times\omega \;\leq_C\; \delta_1\times\omega\times\omega \;=_C\; \delta_1\times\omega,$$

where the axiom $\omega\times\omega =_C \omega$ is used (see Def. 3.13). If $\delta_1,\delta_2 \in \mathcal{L}^{\mathrm{R}}_D$ then both $\delta_1\times\delta_2\times\omega$ and $\delta_1\times\omega$ are in $\mathcal{L}^{\mathrm{R}}_C$, but $\delta_1\times\omega\times\omega \notin \mathcal{L}^{\mathrm{R}}_C$ because $\omega \notin \mathcal{L}^{\mathrm{R}}_D$.

However this is not a counterexample, since $\delta_2\times\omega \leq^{\mathrm{R}} \omega$ which implies that $\delta_1\times\delta_2\times\omega \leq^{\mathrm{R}} \delta_1\times\omega$ is derivable in $\mathcal{T}^{\mathrm{R}}_C$. As a matter of fact, we can show that $\sigma \leq^{\mathrm{R}} \tau$ if and only if $\sigma \leq \tau$ for any $\sigma,\tau \in \mathcal{L}^{\mathrm{R}}$ by a semantic argument.

Lemma 6.2 Take $R = \{\bot \sqsubseteq \top\}$ and set $\Theta_R(\upsilon) = \top$. Then for all $\sigma,\tau \in \mathcal{L}^{\mathrm{R}}_A$, where $A = D,C$, if $\Theta_A(\tau) \sqsubseteq \Theta_A(\sigma)$ then $\sigma \leq^{\mathrm{R}}_A \tau$.

Proof By induction over the structure of types.

$(\sigma,\tau \in \mathcal{L}^{\mathrm{R}}_D)$: Let $\sigma = \delta$, and $\tau = \delta'$. Then $\delta = \wedge_{i\in I}(\kappa_i\to\upsilon)$ and $\delta' = \wedge_{j\in J}(\kappa'_j\to\upsilon)$; hence:

$$\begin{array}{lll} \Theta_D(\delta') = \sqcup_{j\in J}(\Theta_C(\kappa'_j)\Rightarrow\top) \sqsubseteq \sqcup_{i\in I}(\Theta_C(\kappa_i)\Rightarrow\top) = \Theta_D(\delta) & \Rightarrow & \text{(by hypothesis)} \\ \forall j\in J\ \exists i_j \in I\,[\Theta_C(\kappa_{i_j}) \sqsubseteq \Theta_C(\kappa'_j)] & \Rightarrow & (*) \\ \forall j\in J\ \exists i_j \in I\,[\kappa'_j \leq^{\mathrm{R}}_C \kappa_{i_j}] & \Rightarrow & \text{(IH)} \\ \forall j\in J\ \exists i_j \in I\,[\kappa_{i_j}\to\upsilon \leq^{\mathrm{R}}_D \kappa'_j\to\upsilon] & \Rightarrow & \\ \delta \leq^{\mathrm{R}} \wedge_{j\in J}\kappa_{i_j}\to\upsilon \leq^{\mathrm{R}}_D \wedge_{j\in J}(\kappa'_j\to\upsilon) = \delta' & & \end{array}$$

where (*) follows by contraposition: if for some $j_0 \in J$ we had $\Theta_C(\kappa_i) \not\sqsubseteq \Theta_C(\kappa'_{j_0})$ for all $i \in I$ then $\sqcup_{j\in J}(\Theta_C(\kappa'_j)\Rightarrow\top)(\Theta_C(\kappa'_{j_0})) = \top$ and $\sqcup_{i\in I}(\Theta_C(\kappa_i)\Rightarrow\top)(\Theta_C(\kappa'_{j_0})) = \bot$.

$(\sigma,\tau \in \mathcal{L}^{\mathrm{R}}_C)$: Let $\sigma = \kappa$, $\tau = \kappa'$. Since the ordering over $C = D\times D\times\cdots$ is component-wise,

$$\langle d_1, d_2, \ldots \rangle \sqcup \langle d'_1, d'_2, \ldots \rangle \;=\; \langle d_1 \sqcup d'_1, d_2 \sqcup d'_2, \ldots \rangle,$$

and $\Theta_C(\kappa \wedge \kappa') = \Theta_C(\kappa) \sqcup \Theta_C(\kappa')$, we can assume that $\kappa = \delta_1 \times \cdots \times \delta_n \times \omega$ and $\kappa' = \delta'_1 \times \cdots \times \delta'_m \times \omega$. Hence the hypothesis $\Theta_C(\kappa) \sqsubseteq \Theta_C(\kappa')$ reads as

$$\langle \Theta_D(\delta_1), \ldots, \Theta_D(\delta_n), \bot, \ldots \rangle \;\sqsubseteq\; \langle \Theta_D(\delta'_1), \ldots, \Theta_D(\delta'_m), \bot, \ldots \rangle$$

where $\bot, \ldots$ stands for infinitely many $\bot$s; then $\Theta_D(\delta_1) \sqsubseteq \Theta_D(\delta'_i)$ for all $i \leq \min(m,n)$. It is easy to see that $\Theta_D(\delta) \neq \bot$ for any $\delta \in \mathcal{L}^{\mathrm{R}}_D$, and therefore $n \leq m$. Then by induction, $\delta_i \leq^{\mathrm{R}}_D \delta'_i$ for all $i \leq n$. Now $\delta_{n+1} \times \cdots \times \delta_m \times \omega \leq^{\mathrm{R}}_C \omega$ by definition, hence

$$\delta_1 \times \cdots \times \delta_n \times \delta_{n+1} \times \cdots \times \delta_m \times \omega \;\leq^{\mathrm{R}}_C\; \delta'_1 \times \cdots \times \delta'_m \times \omega$$ □

Theorem 6.3 *The pre-orders $\leq^{\mathrm{R}}_D$ and $\leq^{\mathrm{R}}_C$ are the restriction to $\mathcal{L}^{\mathrm{R}}$ of $\leq_D$ and $\leq_C$, respectively.*

Proof Let $\sigma, \tau \in \mathcal{L}^{\mathrm{R}}_A$ for either $A = D, C$, then:

$$\begin{array}{rcll} \sigma \leq_A \tau & \Rightarrow & [\![\sigma]\!]^A \subseteq [\![\tau]\!]^A & \text{(Cor. 4.13)} \\ & \Rightarrow & \uparrow_A \Theta_A(\sigma) \subseteq \uparrow_A \Theta_A(\tau) & \text{(Lem. 4.12)} \\ & \Rightarrow & \Theta_A(\tau) \sqsubseteq \Theta_A(\sigma) & \text{(since } \Theta_A(\sigma) \in \uparrow_A \Theta_A(\tau)\text{)} \\ & \Rightarrow & \sigma \leq^{\mathrm{R}}_A \tau & \text{(Lem. 6.2)} \end{array}$$

Since trivially $\leq^{\mathrm{R}}_A \subseteq \leq_A$, this establishes the thesis. □

Definition 6.4 (Restricted Bases, Contexts, Judgments, and Type Assignment) *i*) A *restricted basis* is a basis Γ such that $\delta \in \mathcal{L}^{\mathrm{R}}_D$ for all $x{:}\delta \in \Gamma$. Similarly, a *restricted name context* is a context Δ with $\kappa \in \mathcal{L}^{\mathrm{R}}_C$ for all $\alpha{:}\kappa \in \Delta$. Finally, for $T \in \textsc{Trm} \cup \textsc{Cmd}$ we say that $\Gamma \vdash T : \sigma \mid \Delta$ is a *restricted judgement* if $\sigma \in \mathcal{L}^{\mathrm{R}}$ and Γ and Δ are a restricted basis and a restricted name context respectively.

ii) The restricted judgement $\Gamma \vdash T : \sigma \mid \Delta$ is *derivable in the restricted typing system*, written $\Gamma \vdash_{\mathrm{R}} T : \sigma \mid \Delta$, if it is derivable in the system of Def. 4.2 without using rule (ω), and all the judgements in the derivation are restricted.

Since the restricted system is just the intersection type system of Section 4, where types occurring in judgements are restricted, and rule (ω) is disallowed, we can use results from the previous section in proofs. However care is necessary, since the lack of rule (ω) invalidates expansion property, as we illustrate in Sect. 6.3.

We only observe that, while it is clear that in the restricted system no term can have type ω, this is still the case for commands, because $\omega \in \mathcal{L}^{\mathrm{R}}_C$ and we have subsumption rule in the system. However judgements of the shape $\Gamma \vdash_{\mathrm{R}} \mathtt{c} : \omega \mid \Delta$ cannot occur in any derivation deducing a type for a term. In fact, if the subject M of the conclusion is a term including the command $\mathtt{c}$ then, for some α, $\mu\alpha.\mathtt{c}$ must be a subterm of M; but rule (μ), which is the only applicable rule, doesn't admit $\Gamma \vdash_{\mathrm{R}} \mathtt{c} : \omega \mid \Delta$ as a premise.

6.2 Typability implies Strong Normalisation

In this subsection we will show that – as can be expected of a well-defined notion of type assignment that does not type recursion and has no general rule that types all terms – all typeable terms are strongly normalising.

For the full system of Def. 4.2 this is not the case. In fact, by means of types not allowed in the restricted system, it is possible to type the fixed-point constructor $\lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$ in a non-trivial way, as shown by the following derivation:

Example 6.5 The fixed-point combinator is typeable in the system of Def. 4.2:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{f{:}\omega\times\omega\to v, x{:}\omega \vdash f:\omega\times\omega\to v \mid}\,(Ax) \quad \dfrac{}{f{:}\omega\times\omega\to v, x{:}\omega \vdash xx:\omega \mid}\,(\omega)}{f{:}\omega\times\omega\to v, x{:}\omega \vdash f(xx):\omega\to v \mid}\,(App)}{f{:}\omega\times\omega\to v \vdash \lambda x.f(xx):\omega\times\omega\to v \mid}\,(Abs) \quad \dfrac{}{f{:}\omega\times\omega\to v \vdash \lambda x.f(xx):\omega \mid}\,(\omega)}{f{:}\omega\times\omega\to v \vdash (\lambda x.f(xx))(\lambda x.f(xx)):\omega\to v \mid}\,(App)}{\vdash \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)):(\omega\times\omega\to v)\times\omega\to v \mid}\,(Abs)$$

Notice that this term does not have a normal form, so is not strongly normalisable.

We start by showing that if a term is typeable in the restricted system then it is strongly normalising. We adapt Tait's computability argument and the idea of saturated sets to our system (see [38], Ch. 3).

Definition 6.6 (Term Stacks) The set Stk of (finite) *term stacks*, whose elements we shall denote by $\vec{L}$, is defined by the following grammar:

$$\mathsf{Stk}: \quad \vec{L} \;::=\; \epsilon \mid L :: \vec{L}$$

where ϵ denotes the empty stack and $L \in \textsc{Trm}$. Moreover, we define *stack application* as follows:

$$\begin{array}{lcl} M\epsilon & \triangleq & M \\ M(P::\vec{L}) & \triangleq & (MP)\,\vec{L} \end{array}$$

So, if $\vec{L} \equiv L_1 :: \cdots :: L_k :: \epsilon$ we have $M\vec{L} \equiv ML_1 \cdots L_k$. We extend the notion of structural substitution to stacks as follows:

$$\begin{array}{lcl} T[\alpha \Leftarrow \epsilon] & \triangleq & T \\ T[\alpha \Leftarrow P :: \vec{L}] & \triangleq & (T[\alpha \Leftarrow P])\,[\alpha \Leftarrow \vec{L}] \end{array}$$

for $T \in \textsc{Trm} \cup \textsc{Cmd}$, when each L_i does not contain α.

We normally omit the trailing ϵ of a stack. Notice that

$$[\alpha]M[\alpha \Leftarrow \vec{L}] \;\equiv\; [\alpha]M\,[\alpha \Leftarrow L_1]\,[\alpha \Leftarrow L_2] \cdots [\alpha \Leftarrow L_n] \;\equiv\; [\alpha](M\,[\alpha \Leftarrow \vec{L}])\vec{L}$$

The notion of string normalisation is formally defined as:

Definition 6.7 The set $\mathcal{SN}$ of terms that are *strongly normalisable* is the set of all terms $M \in \textsc{Trm}$ such that no infinite reduction sequence out of M exists; we write $\mathcal{SN}(M)$ for $M \in \mathcal{SN}$, and $\mathcal{SN}^*$ for the set of finite stacks of terms in $\mathcal{SN}$, and write $\mathcal{SN}(\vec{L})$ if $\vec{L} \in \mathcal{SN}^*$.

The following property of strong normalising terms is straightforward:

Property 6.8 *i) If $\mathcal{SN}(x\vec{M})$ and $\mathcal{SN}(\vec{N})$, then $\mathcal{SN}(x\vec{M}\vec{N})$.*
ii) If $\mathcal{SN}(M[N/x]\vec{P})$ and $\mathcal{SN}(N)$, then $\mathcal{SN}((\lambda x.M)N\vec{P})$.
iii) If $\mathcal{SN}(M)$, then $\mathcal{SN}(\mu\alpha.[\beta]M)$.
iv) If $\mathcal{SN}(\mu\alpha.[\beta]M[\alpha \Leftarrow \vec{N}]\vec{L})$ and $\mathcal{SN}(\vec{N})$, then $\mathcal{SN}((\mu\alpha.[\beta]M)\vec{N}\vec{L})$.
v) If $\mathcal{SN}(\mu\alpha.[\alpha]M[\alpha \Leftarrow \vec{N}]\vec{N}\vec{L})$, then $\mathcal{SN}((\mu\alpha.[\alpha]M)\vec{N}\vec{L})$.

Definition 6.9 (Type Interpretation) We define a pair of mappings

$$\|\cdot\| = \langle \|\cdot\|_1, \|\cdot\|_2 \rangle : (\mathcal{L}^{\mathrm{R}}_D \to \wp(\textsc{Trm})) \times (\mathcal{L}^{\mathrm{R}}_C \to \wp(\mathsf{Stk}))$$

interpreting types as sets of terms and stacks. Writing $\|\delta\| = \|\delta\|_1$ and $\|\kappa\| = \|\kappa\|_2$, the definition is as follows:

$$\begin{array}{rcll} \|\kappa{\to}v\| & = & \{\, T \mid \forall \vec{L} \in \|\kappa\|\, [\mathcal{SN}(T\vec{L})\,]\,\} & \\ \|\omega\| & = & \mathcal{SN}^* & \\ \|\delta{\times}\kappa\| & = & \{\, N{::}\vec{L} \mid N \in \|\delta\| \;\&\; \vec{L} \in \|\kappa\|\,\} & \\ \|\sigma{\wedge}\tau\| & = & \|\sigma\| \cap \|\tau\| & (\sigma,\tau \in \mathcal{L}^{\mathrm{R}}_A,\ A = D,C). \end{array}$$

We will now show that the interpretation of a type is a set of strongly normalisable terms and that neutral terms (those starting with a variable) are in the interpretation of any type.

Lemma 6.10 For any $\delta \in \mathcal{L}^{\mathrm{R}}_D$ and $\kappa \in \mathcal{L}^{\mathrm{R}}_C$:

i) $\|\delta\| \subseteq \mathcal{SN}$ *and* $\|\kappa\| \subseteq \mathcal{SN}^*$.

ii) $x\vec{N} \in \mathcal{SN} \Rightarrow x\vec{N} \in \|\delta\|$.

Proof By induction on the structure of types.

i) Let $\delta = \kappa{\to}v$ and $M \in \|\delta\|$: then, for any $\vec{L} \in \|\kappa\|$, by definition, $\mathcal{SN}(M\vec{L})$, so in particular $\mathcal{SN}(M)$. The case $\kappa = \omega$ follows by definition; the case $\kappa = \delta{\times}\kappa'$ follows by induction, since $\|\delta\| \subseteq \mathcal{SN}$ and $\|\kappa'\| \subseteq \mathcal{SN}^*$. The cases $\delta = \delta_1{\wedge}\delta_2$ and $\kappa = \kappa_1{\wedge}\kappa_2$ follow immediately by induction.

ii) Let $x\vec{N} \in \mathcal{SN}$ and $\delta = \kappa{\to}v$. If $\vec{L} \in \|\kappa\|$ then $\vec{L} \in \mathcal{SN}^*$ by point (*i*), so that $x\vec{N}\vec{L} \in \mathcal{SN}$ by observing that the only possible reductions are inside the components of $\vec{N}$ and $\vec{L}$, which are in $\mathcal{SN}$ by assumption. Then by definition $x\vec{N} \in \|\delta\|$. □

Observe that $v \notin \mathcal{L}^{\mathrm{R}}_D$, and therefore we do not have the clause $\|v\| = \mathcal{SN}$ in Def. 6.9, as was the case in [12]. This clause would be consistent with the previous definition, but having $v \in \mathcal{L}_D$ (the unrestricted language of term types) enforces the equation $v =_D \omega{\to}v$ which is false in the above interpretation. In fact, $\lambda x.xx \in \mathcal{SN} = \|v\|$, so that $(\lambda x.xx){::}\epsilon \in \mathcal{SN}^* = \|\omega\|$, but $(\lambda x.xx)((\lambda x.xx){::}\epsilon) \equiv (\lambda x.xx)(\lambda x.xx) \notin \mathcal{SN}$, and therefore $\|v\| \not\subseteq \|\omega{\to}v\|$. In [12] we managed to avoid this incoherence by ruling out ω from $\mathcal{L}^{\mathrm{R}}_C$ and by interpreting types $\kappa{\to}v$ differently according to the shape of κ; in fact, in that paper $\|\kappa{\to}v\|$ is the set of representable functions from $\mathcal{SN}^*$ to $\mathcal{SN}$ only when $\kappa \neq \omega$, while $\|\omega{\to}v\|$ is just $\mathcal{SN}$.

We will now show that our type interpretation respects type inclusion.

Lemma 6.11 For all $\sigma,\tau \in \mathcal{L}^{\mathrm{R}}$: $\sigma \leq^{\mathrm{R}} \tau \Rightarrow \|\sigma\| \subseteq \|\tau\|$.

Proof By easy induction over the rules in Def. 6.1. For $\kappa \leq^{\mathrm{R}}_C \omega$, notice that $\|\kappa\| \subseteq \mathcal{SN}^* = \|\omega\|$ by Lem. 6.10 (*i*). If $\kappa \leq^{\mathrm{R}}_C \kappa'$, then $\kappa = \delta_1{\times}\cdots{\times}\delta_n{\times}\omega$, $\kappa' = \delta'_1{\times}\cdots{\times}\delta'_m{\times}\omega$, $m \leq n$, and $\delta_i \leq^{\mathrm{R}}_D \delta'_i$. By induction, $\|\delta_i\| \subseteq \|\delta'_i\|$ for all $i \leq m$; notice that $\|\delta_{i+1}{\times}\cdots{\times}\delta_n{\times}\omega\| \subseteq \|\omega\| = \mathcal{SN}^*$ by Lem. 6.10 (*i*), so we can conclude $\|\kappa\| \subseteq \|\kappa'\|$. Assume $\kappa_1{\to}v \leq^{\mathrm{R}}_D \kappa_2{\to}v$ because $\kappa_2 \leq^{\mathrm{R}}_C \kappa_1$, then by induction $\|\kappa_2\| \subseteq \|\kappa_1\|$. Assume now $M \in \|\kappa_1{\to}v\|$, then by definition for all $\vec{L} \in \|\kappa_1\|$, we have $M\vec{L} \in \mathcal{SN}$. Since $\|\kappa_2\| \subseteq \|\kappa_1\|$, also for all $\vec{L} \in \|\kappa_2\|$ we have $M\vec{L} \in \mathcal{SN}$, and therefore also $M \in \|\kappa_2{\to}v\|$. □

The next lemma states that our type interpretation is closed under expansion for the logical and for the structural reduction, with the proviso that the term or stack to be substituted is an element of an interpreted type as well.

Lemma 6.12 For any $\delta,\delta' \in \mathcal{L}^{\mathrm{R}}_D$ and $\kappa \in \mathcal{L}^{\mathrm{R}}_C$:

i) If $M[N/x]\vec{P} \in \|\delta\|$ and $N \in \|\delta'\|$, then $(\lambda x.M)N\vec{P} \in \|\delta\|$.

ii) If $(\mu\alpha.[\beta]M[\alpha \Leftarrow \vec{N}])\vec{P} \in \|\delta\|$ and $\vec{N} \in \|\kappa'\|$, then $(\mu\alpha.[\beta]M)\vec{N}\vec{P} \in \|\delta\|$.

iii) If $(\mu\alpha.[\alpha]M[\alpha \Leftarrow \vec{N}]\vec{N})\vec{P} \in \|\delta\|$, then $(\mu\alpha.[\alpha]M)\vec{N}\vec{P} \in \|\delta\|$.

Proof By induction on the structure of types. If $\delta = \kappa \to \upsilon$, then from $M[N/x]\vec{P} \in \|\kappa \to \upsilon\|$ by definition we have $\mathcal{SN}(M[N/x]\vec{P}\vec{Q})$ for all $\vec{Q} \in \|\kappa\|$ and from $N \in \|\delta'\|$ that $\mathcal{SN}(N)$ by Prop. 6.10 (*i*). Then by Lem. 6.8 (*ii*) also $\mathcal{SN}((\lambda x.M)N\vec{P}\vec{Q})$, so by definition $(\lambda x.M)N\vec{P} \in \|\kappa \to \upsilon\|$. The case $\delta = \delta_1 \wedge \delta_2$ follows directly by induction.

The second and third case follow similarly, but rather using Prop. 6.8 (*iv*) and 6.8 (*v*), respectively. □

In Theorem 6.15 we will show that all typeable terms are strongly normalisable. In order to achieve that, in Lem. 6.14 we will first show that for a term M typeable with δ, any substitution instance M_ξ (*i.e.* replacing all free term variables by terms, and feeding stacks to all free names) is an element of the interpretation of δ, which by Lem. 6.10 implies that M_ξ is strongly normalisable. We need these substitutions to be applied all 'in one go', so define a notion of parallel substitution. The main result is then obtained by taking the substitution that replaces term variables by themselves and names by stacks of term variables. The reason we first prove the result for *any* substitution is that, in the proof of Lem. 6.14, in the case for $\lambda x.M$ and $\mu\alpha.\mathtt{c}$ the substitution is extended, by replacing the bound variable or name with a normal term (or stack).

Definition 6.13 *i*) A pair of partial mappings $\xi = \langle \xi_2, \xi_2 \rangle : (\textsc{Var} \to \textsc{Trm}) \times (\textsc{Name} \to \textsc{Trm}^*)$, where we simply write $\xi\, x = \xi_1(x)$ and $\xi\, \alpha = \xi_2(\alpha)$, is a *parallel substitution* if, for every $p, q \in dom(\xi)$, if $p \neq q$ then $p \notin fv(\xi q)$ and $p \notin fn(\xi q)$.

ii) Borrowing a notation for valuations, for a parallel substitution ξ we define the application of ξ to a term by:

$$\begin{array}{rcll}
([\alpha]M)_\xi & \triangleq & [\alpha]M_\xi\vec{L} & (\text{if } \xi\alpha = \vec{L}) \\
([\beta]M)_\xi & \triangleq & [\beta]M_\xi & (\text{if } \beta \notin dom(\xi)) \\
(\mu\beta.Q)_\xi & \triangleq & \mu\beta.Q_\xi & \\
x_\xi & \triangleq & N & (\text{if } \xi x = N) \\
y_\xi & \triangleq & y & (\text{if } y \notin dom(\xi)) \\
(\lambda x.M)_\xi & \triangleq & \lambda x.M_\xi & \\
(MN)_\xi & \triangleq & M_\xi N_\xi &
\end{array}$$

iii) We define $\xi[N/x]$ and $\xi[\alpha \Leftarrow \vec{L}]$ as, respectively,

$$\xi[N/x]\, y \triangleq \begin{cases} N & (\text{if } y = x) \\ \xi\, y & (\text{otherwise}) \end{cases}$$

$$\xi[\alpha \Leftarrow \vec{L}]\, \beta \triangleq \begin{cases} \vec{L} & (\text{if } \alpha = \beta) \\ \xi\, \beta & (\text{otherwise}) \end{cases}$$

iv) We say that ξ *extends* Γ *and* Δ, if, for all $x{:}\delta \in \Gamma$ and $\alpha{:}\kappa \in \Delta$, we have, respectively, $\xi\, x \in \|\delta\|$ and $\xi\, \alpha \in \|\kappa\|$.

Lemma 6.14 (Replacement Lemma) *Let ξ be a parallel substitution that extends Γ and Δ. Then: if $\Gamma \vdash_{\mathrm{R}} T : \sigma \mid \Delta$ then $T_\xi \in \|\sigma\|$.*

Proof By induction on the structure of derivations. We show some more illustrative cases.

(Ax): then $M = x$. Since $x{:}\delta \in \Gamma$ and ξ extends Γ, we have $x_\xi = \xi\, x \in \|\delta\|$.

(Abs): Then $M = \lambda x.M'$, $\delta = \delta' \times \kappa \to \nu$, and $\Gamma, x{:}\delta' \vdash M' : \kappa \to \nu \mid \Delta$. Take $N \in \|\delta'\|$; since x is bound, by Barendregt's convention we can assume that it does not occur free in the image of ξ, so $\xi[N/x]$ is a parallel substitution that extends $\Gamma, x{:}\delta'$ and Δ. Then by induction,

we have $M'_{\xi[N/x]} \in \|\kappa{\to}\nu\|$. Since x does not occur free in the image of ξ, $M'_{\xi[N/x]} = M'_{\xi}[N/x]$, so also $M'_{\xi}[N/x] \in \|\kappa{\to}\nu\|$. By Lem. 6.12 (*i*), also $(\lambda x.M'_{\xi})N \in \|\kappa{\to}\nu\|$. By definition of $\|\kappa{\to}\nu\|$, for any $\vec{L} \in \|\kappa\|$ we have $\mathcal{SN}((\lambda x.M'_{\xi})N\vec{L})$; notice that $N :: \vec{L} \in \|\delta{\times}\kappa\|$, so $(\lambda x.M')_{\xi} \in \|\delta'{\times}\kappa{\to}\nu\|$.

(*App*): Then $M = PQ$, and there exists δ such that $\Gamma \vdash P : \delta{\times}\kappa{\to}\nu \mid \Delta$ and $\Gamma \vdash Q : \delta \mid \Delta$. Then by induction, we have $P_{\xi} \in \|\delta{\times}\kappa{\to}\nu\|$ and $Q_{\xi} \in \|\delta\|$; notice that $P_{\xi}Q_{\xi} = (PQ)_{\xi}$. Take $\vec{L} \in \|\kappa\|$, we get $Q_{\xi} :: \vec{L} \in \|\delta{\times}\kappa\|$, so also $\mathcal{SN}(P_{\xi}Q_{\xi}\vec{L})$. But then also $P_{\xi}Q_{\xi} \in \|\kappa{\to}\nu\|$.

(μ): Then $M = \mu\alpha.[\beta]N$, and $\delta = \kappa{\to}\nu$. We distinguish two different sub-cases.

($\alpha = \beta$): Then $M = \mu\alpha.[\alpha]N$, $\delta = \kappa{\to}\nu$, and $\Gamma \vdash N : \kappa{\to}\nu \mid \alpha{:}\kappa, \Delta$. Take $\vec{L} \in \|\kappa\|$; since α is bound in M, $\xi[\alpha \Leftarrow \vec{L}]$ is a well-defined parallel substitution that extends Γ and $\alpha{:}\kappa, \Delta$, and by induction, $N_{\xi[\alpha \Leftarrow \vec{L}]} \in \|\kappa{\to}\nu\|$. Since α does not occur free in the image of ξ, $N_{\xi[\alpha \Leftarrow \vec{L}]} = N_{\xi}[\alpha \Leftarrow \vec{L}]$, so we have $N_{\xi}[\alpha \Leftarrow \vec{L}] \in \|\kappa{\to}\nu\|$, and therefore $\mathcal{SN}(N_{\xi}[\alpha \Leftarrow \vec{L}]\vec{L})$. But then also $\mathcal{SN}(\mu\alpha.[\alpha]N_{\xi}[\alpha \Leftarrow \vec{L}]\vec{L})$, by Lem. 6.8 (*iii*), and by Lem. 6.8 (*v*) $\mathcal{SN}((\mu\alpha.[\alpha]N_{\xi})\vec{L})$; so $(\mu\alpha.[\alpha]N)_{\xi} \in \|\kappa{\to}\nu\|$.

($\alpha \neq \beta$): Then $\Delta = \beta{:}\kappa', \Delta'$, and $\Gamma \vdash N : \kappa'{\to}\nu \mid \alpha{:}\kappa, \beta{:}\kappa', \Delta$, and ξ extends Γ and $\beta{:}\kappa', \Delta'$. Assume $\vec{L} \in \|\kappa\|$, then $\xi[\alpha \Leftarrow \vec{L}]$ extends Γ and $\alpha{:}\kappa, \beta{:}\kappa', \Delta'$ and, by induction, we have $N_{\xi[\alpha \Leftarrow \vec{L}]} \in \|\kappa'{\to}\nu\|$. Now let $\vec{Q} \in \|\kappa'\|$, then $\mathcal{SN}(N_{\xi[\alpha \Leftarrow \vec{L}]}\vec{Q})$ and then also $\mathcal{SN}((N\vec{Q})_{\xi[\alpha \Leftarrow \vec{L}]})$. Then $\mathcal{SN}(\mu\alpha.[\beta](N\vec{Q})_{\xi[\alpha \Leftarrow \vec{L}]})$ by Lem. 6.8 (*iii*), so by Def. 6.9, $\mathcal{SN}(\mu\alpha.[\beta](N\vec{Q})_{\xi[\alpha \Leftarrow \vec{L}]})$; then also $\mathcal{SN}(\mu\alpha.[\beta](N\vec{Q})_{\xi}[\alpha \Leftarrow \vec{L}])$.

Then $\mathcal{SN}((\mu\alpha.[\beta](N\vec{Q})_{\xi})\vec{L})$ by Lem. 6.8 (*iv*). Notice that $[\beta]N_{\xi}\vec{Q} = [\beta]N_{\xi}[\beta \Leftarrow \vec{Q}]$; since $\xi\beta = \vec{Q}$, we can infer that $[\beta]N_{\xi}\vec{Q} = [\beta]N_{\xi}$, so $\mathcal{SN}((\mu\alpha.[\beta]N)_{\xi}\vec{L})$. But then $(\mu\alpha.[\beta]N)_{\xi} \in \|\kappa{\to}\nu\|$.

($\wedge$): By induction.

($\leq$): By induction and Lem. 6.11. □

We now come to the main result of this section, that states that all terms typeable in the restricted system are strongly normalisable.

Theorem 6.15 (Typeable terms are $\mathcal{SN}$) *If $\Gamma \vdash M : \delta \mid \Delta$ for some Γ, Δ and δ, then $M \in \mathcal{SN}$.*

Proof Let ξ be a parallel substitution such that $\xi x = x$ for all $x \in dom(\Gamma)$ and $\xi\alpha = \vec{y}_{\alpha}$ for $\alpha \in dom(\Delta)$, where the length of the stack $\vec{y}_{\alpha}$ is $|\kappa|$ if $\alpha{:}\kappa \in \Delta$ (notice that ξ is well defined). By Lem. 6.10, ξ extends Γ and Δ. Hence, by Lem. 6.14, $M_{\xi} \in \|\delta\|$, and then $M_{\xi} \in \mathcal{SN}$ by Lem. 6.10 (*i*). Now

$$\begin{array}{rcl} M_{\xi} & \equiv & M\,[x_1/x_1, \ldots, x_n/x_n, \alpha_1 \Leftarrow \vec{y}_{\alpha_1}, \ldots, \alpha_m \Leftarrow \vec{y}_{\alpha_m}] \\ & \equiv & M\,[\alpha_1 \Leftarrow \vec{y}_{\alpha_1}, \ldots, \alpha_m \Leftarrow \vec{y}_{\alpha_m}] \end{array}$$

Then, by Prop. 6.8, for any $\vec{\beta}$ also $(\mu\alpha_1.[\beta_1] \cdots \mu\alpha_m.[\beta_m]M)\vec{y}_{\alpha_1} \cdots \vec{y}_{\alpha_m} \in \mathcal{SN}$, and therefore also $M \in \mathcal{SN}$. □

6.3 Strongly Normalising Terms are Typeable

In this section we will show the counterpart of the previous result, *i.e.* that all strongly normalisable terms are typeable in our restricted intersection system. This result has been claimed in many papers [49, 5], but has rarely been proven completely.

First we give the shape of terms and commands in normal forms.

Definition 6.16 (Normal Forms) The sets $\mathcal{N} \subseteq$ Trm (and, implicitly, $\mathcal{C} \subseteq$ Cmd) of *normal forms* are defined by the grammar:

$$N \ ::= \ xN_1 \cdots N_k \mid \lambda x.N \mid \mu\alpha.[\beta]N \ (\beta \neq \alpha \text{ or } \alpha \in N)$$

It is straightforward to verify that $\mathcal{N}$ and $\mathcal{C}$ coincide with the sets of irreducible terms and commands, respectively.

We can now show that all terms and commands in normal form are typeable in the restricted system.

Lemma 6.17 *i) If $N \in \mathcal{N}$ then there exist Γ, Δ and $\kappa \to v \in \mathcal{L}_D^{\mathrm{R}}$ such that $\Gamma \vdash_{\mathrm{R}} N : \kappa \to v \mid \Delta$.*
ii) If $\mathtt{C} \in \mathcal{C}$ then there exist Γ, Δ and $\kappa \in \mathcal{L}_C^{\mathrm{R}}$ such that $\Gamma \vdash_{\mathrm{R}} \mathtt{C} : (\kappa \to v) \times \kappa \mid \Delta$.

Proof By simultaneous induction over the definitions of $\mathcal{N}$ and $\mathcal{C}$.

- $(N \equiv xN_1 \ldots N_k)$: Since $N_1, \ldots, N_k \in \mathcal{N}$, by induction (*i*) we have that, for all $i \leq k$, there exist Γ_i, Δ_i and δ_i such that $\Gamma_i \vdash_{\mathrm{R}} N_i : \delta_i \mid \Delta_i$ (the structure of each δ_i plays no role in this part). Take $\Gamma = \Gamma_1 \wedge \cdots \wedge \Gamma_k \wedge \{x{:}\delta_1 \times \cdots \times \delta_k \times \omega \to v\}$ and $\Delta = \Delta_1 \wedge \cdots \wedge \Delta_k$. Then, by weakening, also $\Gamma \vdash_{\mathrm{R}} N_i : \delta_i \mid \Delta$ for all $i \leq k$, and $\Gamma \vdash_{\mathrm{R}} x : (\delta_1 \times \cdots \times \delta_k \times \omega) \to v \mid \Delta$. By repeated applications of (App) we get $\Gamma \vdash_{\mathrm{R}} xN_1 \cdots N_k : \omega \to v \mid \Delta$.
- $(N \equiv \lambda x.M)$: By induction (*i*) there exist Γ, δ', and Δ such that $\Gamma, x{:}\delta' \vdash M : \kappa \to v \mid \Delta$ (if $x \notin fv(M)$, we can add $x{:}\delta$ by weakening, for any $\delta' \in \mathcal{L}_D^{\mathrm{R}}$). Then by (Abs) we obtain $\Gamma \vdash \lambda x.M : \delta \times \kappa \to v \mid \Delta$.
- $(N \equiv \mu\alpha.\mathtt{C})$: By induction (*ii*) there exist Γ, κ, κ', and Δ such that $\Gamma \vdash_{\mathrm{R}} \mathtt{C} : (\kappa \to v) \times \kappa \mid \alpha{:}\kappa', \Delta$ (if $\alpha \notin fn(\mathtt{C})$, we can add $\alpha{:}\kappa'$ by weakening, for any $\kappa' \in \mathcal{L}_C^{\mathrm{R}}$). We get $\Gamma \vdash_{\mathrm{R}} \mu\alpha.\mathtt{C} : \kappa' \to v \mid \Delta$ by applying rule (μ).
- $(\mathtt{C} \equiv [\beta]N)$: By induction (*i*) there exist Γ, $\delta = \kappa \to v$, and Δ such that $\Gamma \vdash_{\mathrm{R}} N : \delta \mid \Delta$. If $\beta \notin dom(\Delta)$, by weakening also $\Gamma \vdash_{\mathrm{R}} N : \delta \mid \beta{:}\kappa, \Delta$, and by rule (Cmd) we obtain $\Gamma \vdash_{\mathrm{R}} [\beta]N : (\kappa \to v) \times \kappa \mid \beta{:}\kappa, \Delta$. If $\beta \in dom(\Delta)$ then $\Delta = \beta{:}\kappa', \Delta'$ for some κ', and we can construct:

$$\dfrac{\dfrac{\dfrac{\boxed{\phantom{\Gamma \vdash N : \kappa \to v \mid \beta{:}\kappa', \Delta'}}}{\Gamma \vdash N : \kappa \to v \mid \beta{:}\kappa', \Delta'}\,(St)}{\Gamma \vdash N : \kappa \to v \mid \beta{:}\kappa \wedge \kappa', \Delta'}\,(St)}{\Gamma \vdash [\beta]N : (\kappa \to v) \times (\kappa \wedge \kappa') \mid \beta{:}\kappa \wedge \kappa', \Delta'}\,(Cmd) \qquad \boxed{\phantom{(\kappa \to v) \times (\kappa \wedge \kappa') \leq^{\mathrm{R}} (\kappa \to v) \times \kappa}} \quad (\kappa \to v) \times (\kappa \wedge \kappa') \leq^{\mathrm{R}} (\kappa \to v) \times \kappa$$

$$\Gamma \vdash [\beta]N : (\kappa \to v) \times \kappa \mid \beta{:}\kappa \wedge \kappa', \Delta'$$

which shows the result. □

In the last case, we are forced to use $(\kappa' \to v) \times (\kappa \wedge \kappa')$ instead of $(\kappa' \to v) \times \kappa$ (otherwise we could not apply rule (μ) to type $\mu\alpha.[\beta]N$) which comes at the price of weakening the assumption $\beta{:}\kappa$ to $\beta{:}\kappa \wedge \kappa'$. However, this is not a disadvantage since we get, for example, $\Gamma \vdash_{\mathrm{R}} \mu\beta.[\beta]N : (\kappa \wedge \kappa') \to v \mid \Delta'$ which safely records in the antecedent type $\kappa \wedge \kappa'$ the functionality of N; notice that, in fact $\kappa' \to v \leq^{\mathrm{R}} (\kappa \wedge \kappa') \to v$.

We will now show that typing in the restricted system is closed under expansion, with the proviso that the term that gets substituted is typeable as well. We first establish that types are preserved from a contractum to the respective redex.

Lemma 6.18 (Contractum expansion) *i) If $\Gamma \vdash_{\mathrm{R}} M[N/x] : \delta \mid \Delta$ and $\Gamma \vdash_{\mathrm{R}} N : \delta' \mid \Delta$, then $\Gamma \vdash_{\mathrm{R}} (\lambda x.M)N : \delta \mid \Delta$.*
ii) If $\Gamma \vdash_{\mathrm{R}} \mu\alpha.\mathtt{C}[\alpha \Leftarrow N] : \delta \mid \Delta$ and $\Gamma \vdash_{\mathrm{R}} N : \delta' \mid \Delta$, then $\Gamma \vdash_{\mathrm{R}} (\mu\alpha.\mathtt{C})N : \delta \mid \Delta$.

Proof i) As in the corresponding case of Thm. 5.4, observing that term types have to be in $\mathcal{L}^{\mathrm{R}}_D$, we have that δ and δ' cannot be equivalent to ω. Remark that in the proof of Thm. 5.4 the fact that δ' might be equivalent to ω is of use omly in case $x \notin fv(M)$. Here we have to assume $\Gamma \vdash_{\mathrm{R}} N : \delta' \mid \Delta$, since otherwise $\Gamma \vdash_{\mathrm{R}} (\lambda x.M)N : \delta \mid \Delta$ is not derivable.

ii) As in the corresponding case of Thm. 5.4. We observe that by Lem. 4.9, $\Gamma \vdash_{\mathrm{R}} \mu\alpha.\mathtt{C}[\alpha \Leftarrow N] : \delta \mid \Delta$ and $\delta \not\sim^{\mathrm{R}}_D \omega$ imply that all the sub-derivations $\Gamma \vdash_{\mathrm{R}} \mathtt{C} : (\kappa_i \to \nu) \times \kappa_i \mid \alpha{:}\kappa'_i, \Delta$ are such that $\wedge_{i\in I}\kappa'_i \to \nu \leq^{\mathrm{R}}_D \delta$ which implies that $(\kappa_i \to \nu) \times \kappa_i \not\sim^{\mathrm{R}}_C \omega$. Indeed, $(\kappa_i \to \nu) \times \kappa_i \sim^{\mathrm{R}}_C \omega$ only if $\kappa_i \sim^{\mathrm{R}}_C \omega$ for all i (that is allowed) and $\nu \sim_D \omega$, which is not the case in either full and restricted type theories.

Note that the hypothesis $\Gamma \vdash_{\mathrm{R}} N : \delta' \mid \Delta$ is again necessary in case $\alpha \notin fn(\mathtt{C})$. □

It is tempting to conclude from Lem. 6.18 that if $M \to N$ by contracting a redex PQ such that $\Gamma \vdash_{\mathrm{R}} Q : \delta' \mid \Delta$ for some δ', then $\Gamma \vdash_{\mathrm{R}} N : \delta \mid \Delta$ implies $\Gamma \vdash_{\mathrm{R}} M : \delta \mid \Delta$. But, unfortunately, this is false even for the ordinary λ-calculus itself.

The problem is that in case of (β)-reduction the fact that

$$\text{if } \Gamma \vdash M[N/x] : \sigma \mid \text{ and } \Gamma \vdash N : \tau \mid \text{, then } \Gamma \vdash (\lambda x.M)N : \sigma \mid$$

does not extend to arbitrary contexts $C[M[N/x]]$: *i.e.*, it is false that

$$\text{if } \Gamma \vdash C[M[N/x]] : \sigma \mid \text{ and } \Gamma \vdash N : \tau \mid \text{, then } \Gamma \vdash C[(\lambda x.M)N] : \sigma \mid$$

In fact, the context Γ for typing $C[M[N/x]]$ changes in the proof, so induction does not apply. Reformulating this as

$$\text{if } \Gamma_1 \vdash C[M[N/x]] : \sigma \mid \text{ and } \Gamma_2 \vdash N : \tau \mid \text{, then } \Gamma_1 \wedge \Gamma_2 \vdash C[(\lambda x.M)N] : \sigma \mid$$

(which has been used by various authors) gives no improvement; the problem is that a free variable in N might be bound in the context, which implies that the derived type might change (get bigger in the sense of $\leq$), and that $\Gamma_1 \wedge \Gamma_2$ is not a correct basis for $C[(\lambda x.M)N]$, since it contains types for bound variables. This suggests the property

$$\begin{array}{c}\text{if } \Gamma_1 \vdash C[M[N/x]] : \sigma \mid \text{ and } \Gamma_2 \vdash N : \tau \mid, \\ \text{then there exists } \Gamma_3 \leq \Gamma_1, \rho \geq \sigma \text{ such that } \Gamma_3 \vdash C[(\lambda x.M)N] : \rho \mid\end{array}$$

but this is also not achievable, since the use of $\leq$ in the derivation for P in MP (where $MP \to MQ$) forces a $\geq$ step on M which is not always achievable; this throws the proof for the case of application irreparably out of kilter.

To illustrate this in the context of our system, take $\lambda x.(\lambda y.x)(xx)$ which reduces to $\lambda x.x$, which we can type as follows:

$$\dfrac{\dfrac{}{x{:}\omega\to v \vdash x : \omega\to v \mid}\,(Ax)}{\vdash \lambda x.x : (\omega\to v)\times\omega\to v \mid}\,(Abs)$$

We cannot infer this type for $\lambda x.(\lambda y.x)(xx)$ in the restricted system. To type the sub-term xx when rule (ω) is not available, the best we can do is (setting $\delta = \kappa\to v$):

$$\dfrac{\dfrac{\dfrac{}{x{:}(\delta\times\kappa\to v)\wedge\delta \vdash x : (\delta\times\kappa\to v)\wedge\delta \mid}\,(Ax)}{x{:}(\delta\times\kappa\to v)\wedge\delta \vdash x : \delta\times\kappa\to v \mid}\,(\leq^{\mathrm{R}}) \qquad \dfrac{\dfrac{}{x{:}(\delta\times\kappa\to v)\wedge\delta \vdash x : (\delta\times\kappa\to v)\wedge\delta \mid}\,(Ax)}{x{:}(\delta\times\kappa\to v)\wedge\delta \vdash x : \delta \mid}\,(\leq^{\mathrm{R}})}{x{:}(\delta\times\kappa\to v)\wedge\delta \vdash xx : \delta \mid}\,(App)$$

Using the same type for x, for $\lambda y.x$ we can construct:

$$\dfrac{\dfrac{\dfrac{}{y{:}\delta, x{:}(\delta\times\kappa\to\upsilon)\wedge\delta \vdash x : (\delta\times\kappa\to\upsilon)\wedge\delta \mid}\,(Ax)}{y{:}\delta, x{:}(\delta\times\kappa\to\upsilon)\wedge\delta \vdash x : \kappa'\to\upsilon \mid}\,(\leq^{\mathrm{R}})}{x{:}(\delta\times\kappa\to\upsilon)\wedge\delta \vdash \lambda y.x : \delta\times\kappa'\to\upsilon \mid}\,(Abs)$$

for any κ' such that $(\delta\times\kappa\to\upsilon)\wedge\delta \leq^{\mathrm{R}} \kappa'\to\upsilon$, and therefore

$$\dfrac{\dfrac{\boxed{} \qquad \boxed{}}{x{:}(\delta\times\kappa\to\upsilon)\wedge\delta \vdash (\lambda y.x)(xx) : \kappa'\to\upsilon \mid}\,(App)}{\vdash \lambda x.(\lambda y.x)(xx) : ((\delta\times\kappa\to\upsilon)\wedge\delta)\times\kappa'\to\upsilon \mid}\,(Abs)$$

So we cannot infer the type $(\omega\to\upsilon)\times\omega\to\upsilon$ for $\lambda x.(\lambda y.x)(xx)$ and a general subject-expansion result doesn't hold in the restricted system, not even by requiring typability of all substituted subterms.

We can however show the weaker but sufficient statement that although assignable types are not preserved, *typability* is: we cannot type $\lambda x.(\lambda y.x)(xx)$ with $(\omega\to\upsilon)\times\omega\to\upsilon$, but still we can type it in the restricted system.

We will established this through considering *leftmost-outermost* reduction (*lor*), following a suggestion by Betti Venneri; this technique is also the one used in [38], and in [7] in the context of the strict type assignment system of [5]; [10] presents the proof for the system with strict types and a co-variant type inclusion relation of [6].

Definition 6.19 An occurrence of a redex $\mathsf{R} = (\lambda x.P)Q$ or $(\mu\alpha.[\beta]P)Q$ in a term M is called the *left-most outer-most redex of* M ($lor(M)$), if and only if:

i) there is no redex R' in M such that $\mathsf{R}' = \mathsf{C}[\mathsf{R}]$ with $\mathsf{C}[-] \neq [-]$ (*outer-most*) ;

ii) there is no redex R' in M such that $M = \mathsf{C}_0[\mathsf{C}_1[\mathsf{R}']\,\mathsf{C}_2[\mathsf{R}]]$ (*left-most*).

We write $M \to_{lor} N$ when M reduces to N by contracting $lor(M)$.

The correct subject expansion result (with respect to *lor*, Lem. 6.20) is now the one used for the proof that all strongly normalisable terms are typeable, which uses induction on the length of the *lor*-path.

Lemma 6.20 Assume $M \to_{lor} N$ with $lor(M) = PQ$. If $\Gamma \vdash_{\mathrm{R}} N : \delta \mid \Delta$, and $\Gamma \vdash_{\mathrm{R}} Q : \delta'' \mid \Delta$, then there exist Γ', Δ' and δ' such that $\Gamma' \vdash_{\mathrm{R}} M : \delta' \mid \Delta'$.

Proof First we observe that, since $\delta \in \mathcal{L}^{\mathrm{R}}_D$, $\delta = \wedge_{i\in I}\kappa_i\to\upsilon$ for some I so that certain $\kappa_i \in \mathcal{L}^{\mathrm{R}}_C$, and $\delta \not\sim_D \omega$ (in the full type theory). Hence $\Gamma \vdash_{\mathrm{R}} N : \kappa_i\to\upsilon \mid \Delta$ for all $i \in I$. Hence it suffices to show the thesis when δ is an arrow type. We reason by induction over the structure of terms.

$(M \equiv VP_1\cdots P_n)$: We distinguish two sub-cases:

$(V \equiv lor(M))$: Then $M \to_{lor} V'P_1\cdots P_n \equiv N$ and V' is the contractum of $V \equiv PQ$. By Lem. 4.9 we know that $\Gamma \vdash_{\mathrm{R}} V'P_1\cdots P_n : \kappa\to\upsilon \mid \Delta$ implies that

$$\Gamma \vdash_{\mathrm{R}} V' : \delta_{j,1}\times\cdots\times\delta_{j,n}\times\kappa\to\upsilon \mid \Delta$$

for all $j \in J$, and $\Gamma \vdash_{\mathrm{R}} P_h : \delta_{j,h} \mid \Delta$ for all $h \in \underline{n}$. Since V' is the contractum of the redex PQ and $\Gamma \vdash_{\mathrm{R}} Q : \delta'' \mid \Delta$, we can apply Lem. 6.18 and get

$$\Gamma \vdash_{\mathrm{R}} V : d_{j,1}\times\cdots\times\delta_{j,n}\times\kappa\to\upsilon \mid \Delta,$$

from which, repeatedly applying rule (App), we get $\Gamma \vdash_{\mathrm{R}} VP_1\cdots P_n : \kappa\to\upsilon \mid \Delta$. Take $\Gamma' = \Gamma$, $\delta' = \kappa\to\upsilon$ and $\Delta' = \Delta$.

$(V \equiv z)$: Then $lor(M) = lor(P_h)$ for some $h \in \underline{n}$, and $N \equiv VP_1 \cdots P'_h \cdots P_n$ with $P_h \to_{lor} P'_h$. Reasoning as in the previous case we get $\Gamma \vdash_{\mathrm{R}} P'_h : \delta_{j,h} \mid \Delta$ and

$$\Gamma(z) \;\leq\; \delta_{j,1} \times \cdots \times \delta_{j,h} \times \cdots \times \delta_{j,n} \times \kappa \to \upsilon.$$

By induction there exist Γ_1, $\delta'_{j,h}$ and Δ_1 such that $\Gamma_{j,h} \vdash_{\mathrm{R}} P_h : \delta'_{j,h} \mid \Delta_{j,h}$. We then set

$$\begin{array}{rcl} \Gamma' & = & \Gamma \wedge \wedge_{j\in J, h\in H_j} \Gamma_{j,h} \wedge \{z{:}\delta_{j,1} \times \cdots \times \delta'_{j,h} \times \cdots \times \delta_{j,n} \times \kappa \to \upsilon\} \\ \Delta' & = & \Delta \wedge \wedge_{j\in J, h\in H_j} \Delta_{j,h} \end{array}$$

Then by applying rules (St) and (App) we conclude $\Gamma' \vdash_{\mathrm{R}} z\,P_1 \cdots P_n : \kappa \to \upsilon \mid \Delta'$ as desired.

$(M \equiv \lambda y.M')$: If $M \to_{lor} N$, then $N = \lambda y.N'$ and $M' \to_{lor} N'$. Then there exists δ_j and κ_j such that $\delta = \wedge_{j\in J} \delta_j \times \kappa_j \to \upsilon$, and $\Gamma, y{:}\delta_j \vdash_{\mathrm{R}} N' : \kappa_j \to \upsilon \mid \Delta$ for all $j \in J$. By induction, there exists Γ'', Δ'', δ'_j, and κ'_j such that $\Gamma'', y{:}\delta'_j \vdash_{\mathrm{R}} M' : \kappa'_j \to \upsilon \mid \Delta''$. Then, by rule (Abs), $\Gamma'' \vdash_{\mathrm{R}} \lambda y.M' : \delta'_j \times \kappa'_j \to \upsilon \mid \Delta''$ for all $j \in J$ so that $\Gamma'' \vdash_{\mathrm{R}} \lambda y.M' : \wedge_{j\in J} \delta'_j \times \kappa'_j \to \upsilon \mid \Delta''$; take $\Gamma' = \Gamma''$, $\Delta' = \Delta''$, and $\delta' = \wedge_{j\in J} \delta'_j \times \kappa'_j \to \upsilon$.

$(M \equiv \mu\alpha.[\beta]M')$: If $M \to_{lor} N$, then $N = \mu\alpha.[\beta]N'$ and $M' \to_{lor} N'$. By Lem. 4.9, using the fact that δ is non-trivial by assumption, and assuming $\alpha \neq \beta$ we have that

$$\Gamma \vdash N' : \wedge_{j\in J, h\in H_j} \kappa_{j,h} \to \upsilon \mid \alpha{:}\kappa'_j, \beta{:}\kappa'_{j,h}, \Delta,$$

where $\wedge_{j\in J, h\in H} \kappa_{j,h} \to \upsilon \leq_D \kappa_j \to \upsilon$ for all $j \in J$ and $\wedge_{j\in J} \kappa_j \to \upsilon \leq_D \delta$.

By induction there exist $\Gamma_{j,h}$, $\delta'_{j,h} = \wedge_{j\in J, h\in H_j, u\in U_{j,h}} \kappa'_{j,h,u} \to \upsilon$, κ''_j, $\kappa''_{j,h}$ and $\Delta_{j,h}$ such that

$$\Gamma_{j,h} \vdash_{\mathrm{R}} N' : \wedge_{j\in J, h\in H_j, u\in U_{j,h}} \kappa'_{j,h,u} \to \upsilon \mid \alpha{:}\kappa''_j, \beta{:}\kappa''_{j,h}, \Delta_{j,h}.$$

Taking $\begin{array}{rcl} \Gamma' & = & \Gamma \wedge \wedge_{j\in J, h\in H_j} \Gamma_{j,h} \\ \Delta' & = & \Delta \wedge \wedge_{j\in J, h\in H_j} \Delta_{j,h} \wedge \{\alpha{:}\wedge_{j\in J, h\in H_j} \kappa''_j, \beta{:}\wedge_{j\in J, h\in H_j, u\in U_{j,h}} (\kappa'_{j,h,u} \wedge \kappa''_{j,h})\} \end{array}$

we have that $\Gamma' \vdash_{\mathrm{R}} [\beta]N' : (\kappa'_{j,h,u} \to \upsilon) \times \kappa'_{j,h,u} \mid \Delta'$ for all $j \in J, h \in H_j, u \in U_{j,h}$, possibly using rule (St); we get the result by applying rules (μ) and $(\wedge)$.

$(M \equiv \mu\alpha.[\alpha]M')$: This case is similar to the previous one and easier. □

We observe that considering leftmost-outermost reduction in the proof above is crucial to rule out the case $M \equiv VP_1 \cdots P_j \cdots P_n \to VP_1 \cdots P'_j \cdots P_n$ because $P_j \to P'_j$ where $V \equiv PQ$ is a redex, hence different than a variable. In fact, the induction hypothesis now tells us that if $P'_j : \delta'_j$ then $P_j : \delta_j$ for some δ_j which is in general unrelated to δ'_j; now nothing ensures that δ_j is compatible with the type of V. Instead, we can solve the problem for $V \equiv z$ by taking a suitable weaker assumption for the typing of z, which is the same trick to circumvent the difficulty noted before the last lemma, that arises both with λ and μ-abstraction.

We can now show that all strongly normalisable $\lambda\mu$-terms are typeable in the restricted system.

Theorem 6.21 (Typeability of $\mathcal{SN}$-Terms) *For all $M \in \mathcal{SN}$ there exist Γ and Δ and a type δ such that $\Gamma \vdash_{\mathrm{R}} M : \delta \mid \Delta$.*

Proof First observe that *lor* is normalising. Then we reason by induction on the maximum of the lengths of reduction sequences for a strongly normalisable term M to its normal form (denoted by $\#(M)$).

$(\#(M) = 0)$: Then M is in normal form, and by Lem. 6.17, there exist Γ and δ such that $\Gamma \vdash_{\mathrm{R}} M : \delta \mid \Delta$.

$(\#(M) \geq 1)$: Let $M \to_{lor} N$ by contracting the redex $PQ = lor(M)$, then $\#(N) < \#(M)$. Then $\#(Q) < \#(M)$ (since $Q \in \mathcal{SN}$ is a proper subterm of a redex in M). Then by induction there exist Γ_1, Γ_2, Δ_1, Δ_2, δ_1, and δ_2 such that $\Gamma_1 \vdash_{\mathrm{R}} N : \delta_1 \mid \Delta_1$ and $\Gamma_2 \vdash_{\mathrm{R}} Q : \delta_2 \mid \Delta_2$. By Lem. 6.20 there exist Γ, Δ, and δ such that $\Gamma \vdash_{\mathrm{R}} M : \delta \mid \Delta$. □

7 Simply typed $\lambda\mu$-calculus and Intersection Types

In the previous sections we considered the $\lambda\mu$-calculus as a type-free calculus; in this section we will show that we can establish a connection between our intersection type assignment system and Parigot's logical assignment system. We will show that logical formulas translate into types of the appropriate sort, and moreover that this can be done in the restricted system of Sect. 6. Hence, the characterisation result carries over and can be used to establish the strong normalisation property of Parigot's calculus.

We use a version of Parigot's logical system as presented in [46], which is equivalent to the original one if just terms (so not also proper commands, *i.e.* elements of Cmd) are typed. This implies that the rule for $\bot$ does not need to be taken into account.

We briefly recall Parigot's first-order type assignment system, that we call the Simply-Typed $\lambda\mu$-calculus.

Definition 7.1 (Parigot's Simply Typed $\lambda\mu$-calculus) *i*) The set **LF** of *Logical Formulas* is defined by the following grammar:

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

where φ ranges over an infinite, denumerable set of *Proposition (Type) Variables*.

ii) *Judgements* are of the form $\Pi \vdash_{\mathrm{P}} M : A \mid \Sigma$, where $M \in \mathrm{Trm}$; Π and Σ are finite mappings from Var and Name, respectively, to formulas, and are normally written as finite sets of pairs of term variables and formulas and of names and formulas respectively, as in $\Pi = \{x_1{:}A_1, \ldots, x_n{:}A_n\}$ and $\Sigma = \{\alpha_1{:}B_1, \ldots, \alpha_m{:}B_m\}$.

iii) The *inference rules* of this system are:

$$(Ax): \ \frac{}{\Pi, x{:}A \vdash x : A \mid \Sigma} \qquad (\mu_1): \ \frac{\Pi \vdash M : A \mid \alpha{:}A, \Sigma}{\Pi \vdash \mu\alpha.[\alpha]M : A \mid \Sigma} \qquad (\mu_2): \ \frac{\Pi \vdash M : B \mid \alpha{:}A, \beta{:}B, \Sigma}{\Pi \vdash \mu\alpha.[\beta]M : A \mid \beta{:}B, \Sigma}$$

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

We write $\Gamma \vdash_{\mathrm{P}} M : A \mid \Sigma$ to denote that this judgement is derivable in this system.

Through the Curry-Howard correspondence (*formulas as types* and *proofs as terms*), the underlying logic of this system is the *minimal classical logic* ([4]).

Comparing Parigot's system with ours we observe that rules $(\to I)$ and $(\to E)$ bear some similarity with (Abs) and (App), and rules (μ_1) and (μ_2) are similar to a combination of (μ) and (Cmd):

Lemma 7.2 The following rules are derivable in the system of Def. 4.2 (and in the restricted system as well):

$$\frac{\Gamma \vdash M : \kappa \to \upsilon \mid \alpha{:}\kappa, \Delta}{\Gamma \vdash \mu\alpha.[\alpha]M : \kappa \to \upsilon \mid \Delta} \ (\mu_1) \qquad \frac{\Gamma \vdash M : \kappa' \to \upsilon \mid \alpha{:}\kappa, \beta{:}\kappa', \Delta}{\Gamma \vdash \mu\alpha.[\beta]M : \kappa \to \upsilon \mid \beta{:}\kappa', \Delta} \ (\mu_2)$$

Proof Consider the derivations:

$$\dfrac{\dfrac{\Gamma \vdash M : \kappa \to v \mid \alpha{:}\kappa, \Delta}{\Gamma \vdash [\alpha]M : (\kappa \to v) \times \kappa \mid \alpha{:}\kappa, \Delta}\ (Cmd)}{\Gamma \vdash \mu\alpha.[\alpha]M : \kappa \to v \mid \Delta}\ (\mu) \qquad \dfrac{\dfrac{\Gamma \vdash M : \kappa' \to v \mid \alpha{:}\kappa, \beta{:}\kappa', \Delta}{\Gamma \vdash [\beta]M : (\kappa' \to v) \times \kappa' \mid \alpha{:}\kappa, \beta{:}\kappa', \Delta}\ (Cmd)}{\Gamma \vdash \mu\alpha.[\beta]M : \kappa \to v \mid \beta{:}\kappa', \Delta}\ (\mu)$$

□

As an example illustrating the fact that the system for $\lambda\mu$ is more expressive than the simply typed λ-calculus, we consider the following proof of Peirce's Law, which we take from [45]:

$$\dfrac{\dfrac{}{x{:}(A\to B)\to A \vdash x : (A\to B)\to A \mid \alpha{:}A}\ (Ax) \quad \dfrac{\dfrac{\dfrac{}{x{:}(A\to B)\to A, y{:}A \vdash y : A \mid \alpha{:}A, \beta{:}B}\ (Ax)}{x{:}(A\to B)\to A, y{:}A \vdash \mu\beta.[\alpha]y : B \mid \alpha{:}A}\ (\mu_2)}{x{:}(A\to B)\to A \vdash \lambda y.\mu\beta.[\alpha]y : A\to B \mid \alpha{:}A}\ (\to I)}{\dfrac{\dfrac{x{:}(A\to B)\to A \vdash x(\lambda y.\mu\beta.[\alpha]y) : A \mid \alpha{:}A}{x{:}(A\to B)\to A \vdash \mu\alpha.[\alpha](x(\lambda y.\mu\beta.[\alpha]y)) : A \mid}\ (\mu_1)}{\vdash \lambda x.\mu\alpha.[\alpha](x(\lambda y.\mu\beta.[\alpha]y)) : ((A\to B)\to A)\to A \mid}\ (\to I)}\ (\to E)$$

We observe that the term $\lambda x.\mu\alpha.[\alpha](x(\lambda y.\mu\beta.[\alpha]y))$ is typable in the restricted type system (and hence in the full system as well) by a derivation with the very same structure. Indeed, let $\delta_{A\to B} \triangleq \kappa_{A\to B}\to v$, where $\kappa_{A\to B} \triangleq (\kappa_A\to v)\times\kappa_B$ and $\kappa_A, \kappa_B \in \mathcal{L}^{\mathrm{R}}_{\mathrm{C}}$ are arbitrary; set further $\kappa_{(A\to B)\to A} \triangleq \delta_{A\to B}\times\kappa_A$ and $\delta_{(A\to B)\to A} \triangleq \kappa_{(A\to B)\to A}\to v$ ($\triangleq \delta_{A\to B}\times\kappa_A\to v$), then

(with $\Gamma = x{:}((\kappa_A\to v)\times\kappa_B\to v)\times\kappa_A\to v$)

$$\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : ((\kappa_A\to v)\times\kappa_B\to v)\times\kappa_A\to v \mid \alpha{:}\kappa_A}\ (Ax) \quad \dfrac{\dfrac{\dfrac{}{\Gamma, y{:}\kappa_A\to v \vdash y : \kappa_A\to v \mid \alpha{:}\kappa_A, \beta{:}\kappa_B}\ (Ax)}{\Gamma, y{:}\kappa_A\to v \vdash \mu\beta.[\alpha]y : \kappa_B\to v \mid \alpha{:}\kappa_A}\ (\mu_2)}{\Gamma \vdash \lambda y.\mu\beta.[\alpha]y : (\kappa_A\to v)\times\kappa_B\to v \mid \alpha{:}\kappa_A}\ (Abs)}{\Gamma \vdash x(\lambda y.\mu\beta.[\alpha]y) : \kappa_A\to v \mid \alpha{:}\kappa_A}\ (App)}{\dfrac{\Gamma \vdash \mu\alpha.[\alpha](x(\lambda y.\mu\beta.[\alpha]y)) : \kappa_A\to v \mid}{\vdash \lambda x.\mu\alpha.[\alpha](x(\lambda y.\mu\beta.[\alpha]y)) : (((\kappa_A\to v)\times\kappa_B\to v)\times\kappa_A\to v)\times\kappa_A\to v \mid}\ (Abs)}\ (\mu_1)$$

As suggested in the example above, we can interpret formulas into intersection types of our system. Notably types from the restricted language $\mathcal{L}^{\mathrm{R}}$ do suffice.

Definition 7.3 The translation functions $\cdot^D : \mathbf{LF} \to \mathcal{L}^{\mathrm{R}}_D$ and $\cdot^C : \mathbf{LF} \to \mathcal{L}^{\mathrm{R}}_{\mathrm{C}}$ are defined by:

$$\begin{array}{rcl} \varphi^C & \triangleq & v\times\omega \\ (A\to B)^C & \triangleq & (A^C\to v)\times B^C \\ A^D & \triangleq & A^C\to v \end{array}$$

We extend these mappings to bases and name contexts by: $\Pi^D = \{x{:}A^D \mid x{:}A \in \Pi\}$ and $\Sigma^C = \{\alpha{:}A^C \mid \alpha{:}A \in \Sigma\}$.

It is straightforward to show that the above translations are well defined.

Theorem 7.4 (Derivability preservation) *If $\Pi \vdash_{\mathrm{P}} M : A \mid \Sigma$ then $\Pi^D \vdash M : A^D \mid \Sigma^C$.*

Proof Each rule of the simply-typed $\lambda\mu$-calculus has a corresponding one in the restricted intersection type system; hence it suffices to show that rules are preserved when translating formulas into types. The case of (Ax) is straightforward.

$$(\to I):\ \dfrac{\Pi^D, x{:}A^D \vdash M : B^D \mid \Sigma^C}{\Pi^D \vdash \lambda x.M : (A\to B)^D \mid \Sigma^C}\ (Abs) \ \triangleq\ \dfrac{\Pi^D, x{:}A^C\to v \vdash M : B^C\to v \mid \Sigma^C}{\Pi^D \vdash \lambda x.M : ((A^C\to v)\times B^C)\to v \mid \Sigma^C}\ (Abs)$$

$$(\to E):\ \frac{\Pi^D \vdash M : (A\to B)^D \mid \Sigma^C \qquad \Pi \vdash N : A^D \mid \Sigma^C}{\Pi \vdash MN : B^D \mid \Sigma}\ (App) \quad \triangleq$$

$$\frac{\Pi^D \vdash M : ((A^C\to v)\times B^C)\to v \mid \Sigma^C \qquad \Pi \vdash N : A^C\to v \mid \Sigma^C}{\Pi^D \vdash MN : B^C\to v \mid \Sigma^C}\ (App)$$

$$(\mu_1):\ \frac{\Pi^D \vdash M : A^D \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.[\alpha]M : A^D \mid \Sigma^C}\ (\mu_1) \ \triangleq\ \frac{\Pi^D \vdash M : A^C\to v \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.[\alpha]M : A^C\to v \mid \Sigma^C}\ (\mu_1)$$

$$(\mu_2):\ \frac{\Pi^D \vdash M : B^D \mid \alpha{:}A^C, \beta{:}B^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.[\beta]M : A^D \mid \beta{:}B^C, \Sigma^C}\ (\mu_2) \ \triangleq\ \frac{\Pi^D \vdash M : B^C\to v \mid \alpha{:}A^C, \beta{:}B^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.[\beta]M : A^C\to v \mid \beta{:}B^C, \Sigma^C}\ (\mu_2)$$

□

Strong normalisation of typeable terms in Parigot's simply-typed $\lambda\mu$-calculus (first proved in [46]) now follows as a corollary of our characterisation result.

Corollary 7.5 (Strong Normalisability of Parigot's Simply Typed $\lambda\mu$-calculus)
If $\Pi \vdash_{\mathrm{P}} M : A \mid \Sigma$, *then* $M \in \mathcal{SN}$.

Proof By Thm. 7.4 and Lem. 7.2, if $\Pi \vdash_{\mathrm{P}} M : A \mid \Sigma$ then $\Pi^D \vdash_{\mathrm{R}} M : A^D \mid \Sigma^C$. That M is $\mathcal{SN}$ now follows from Thm. 6.21. □

We end this section by observing that negation can be added to the syntax of logical formulas without disrupting the correspondence with the (restricted) intersection types. Let $\bot$ be added to the formula syntax as a special atom for falsehood. Then consider the logical system which is obtained from Def. 7.1 by replacing rules (μ_1) and (μ_2) by:

$$\frac{\Pi \vdash \mathsf{c} : \bot \mid \alpha{:}A, \Sigma}{\Pi \vdash \mu\alpha.\mathsf{c} : A \mid \Sigma}\ (Activate) \qquad \frac{\Pi \vdash M : A \mid \alpha{:}A, \Sigma}{\Pi \vdash [\alpha]M : \bot \mid \alpha{:}A, \Sigma}\ (Passivate)$$

The resulting system is the same as in [19], and in [48] Section 3.1 but with two contexts instead of a single one. The only difference is that in rule $(Passivate)$ we require the assumption $\alpha{:}A$ in the name contexts, which is just added to the conclusion in both [19] and [48] since there contraction and weakening are implicitly assumed.

In this new system, the rules (μ_1) and (μ_2) are admissible. By defining $\neg A \triangleq A\to\bot$ and considering all the formulas in the Σ context as negated, rule $(Activate)$ corresponds to the *reductio ad absurdum* rule of classical logic. On the other hand rule $(Passivate)$ can be read as the $\neg$-elimination rule, saying that from A and $\neg A$ we get falsity.

To interpret $\bot$ into our intersection type system we need to keep track of the contradiction from which it arises; therefore we write the slightly different rule:

$$\frac{\Pi \vdash M : A \mid \alpha{:}A, \Sigma}{\Pi \vdash [\alpha]M : \bot_A \mid \alpha{:}A, \Sigma}\ (Passivate')$$

where $\bot_A$ is a new constant for each formula A. Now, by adding

$$\bot_A^C \ \triangleq\ (A^C\to v)\times A^C$$

to the translation, we obtain:

$$\frac{\Pi^D \vdash \mathsf{c} : \bot_B^C \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.\mathsf{c} : A^D \mid \Sigma^C} \quad \triangleq \quad \frac{\Pi^D \vdash \mathsf{c} : (B^C\to v)\times B^C \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash \mu\alpha.\mathsf{c} : A^C\to v \mid \Sigma^C}\ (\mu)$$

$$\frac{\Pi^D \vdash M : A^D \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash [\alpha]M : \bot_A^C \mid \alpha{:}A^C, \Sigma^C} \ \triangleq\ \frac{\Pi^D \vdash M : A^C\to v \mid \alpha{:}A^C, \Sigma^C}{\Pi^D \vdash [\alpha]M : (A^C\to v)\times A^C \mid \alpha{:}A^C, \Sigma^C}\ (Cmd)$$

8 Related work

The starting point of the present work is [9], where a type assignment system for $\lambda\mu$-terms with intersection and union types was proved to be invariant under reduction and expansion. That system was without apparent semantical justification, which motivated the present new construction. With respect to the system of [9], our system does not use union types, and introduces product types for continuations, which is its main characteristic. The introduction of product types is inspired to the continuation model of [52], which is the main source of this paper, together with [17], which contains the first construction of a λ-model as a filter model. However, as explained in the introduction and in the main body of the paper, we have followed the inverse path, from the model to the type system, building over [22] and [1].

In [11] we conjectured that in an appropriate subsystem of the present one, it should be possible to type exactly all strongly normalising $\lambda\mu$-terms. We established that result in [12], though via a less elegant variant of our system than the restricted system in Sect. 6. The first to state the characterisation result for the λ-calculus was Pottinger [49], using a notion of type assignment similar to the intersection system of [20, 23], but extended in that it is also closed for η-reduction, which is equivalent to adding a co-variant type inclusion relation; in particular, this is a system that is defined without the type constant ω. However, to show that all typeable terms are strongly normalisable, [49] only *suggests* a proof using Tait's computability technique [53]. A detailed proof, using computability, in the context of the ω-free BCD-system [17] is given in [5]; to establish the same result, saturated sets are used by Krivine in [38] (Ch. 4), in Ghilezan's survey [33], and in [10].

The converse of that result, namely the property that all strongly normalisable terms are typeable has proven to be more elusive: it has been claimed in many papers but not shown in full (we mention [49, 5, 33]); in particular, the proof for the property that type assignment is closed for subject expansion (the converse of subject reduction) is dubious. Subject expansion can only reliably be shown for *left-most outermost* reduction, which is used for the proofs in [38, 13, 7, 10], and our result follows that approach.

The translation in Sect. 7 that maps simple types into our extension of intersection types is a form of negative translation; in [37] it is extended to the system in [9], thereby relating the original intersection and union type assignment system for $\lambda\mu$ to ours.

The model in [52] is not a model of de Groote and Saurin's $\Lambda\mu$-calculus, but a variant of it, dubbed a 'stream model' in [42]; it provides a sound interpretation of the extended calculus. Building over stream models, in [40] it has been proven that the same type theory, but with different rules of the type assignment, gives a finitary description of the model matching the reduction in the stronger sense that the approximation theorem holds.

In [43] an extension of $\Lambda\mu$ is considered, called $\Lambda\mu_{\mathsf{cons}}$. A type assignment for $\Lambda\mu_{\mathsf{cons}}$ based on [40]'s type system is proposed, and subject reduction and strong normalisation of the reduction on the typed $\Lambda\mu_{\mathsf{cons}}$ are proven. In [41] the same system is shown to enjoy Friedman's theorem.

Conclusions

We have presented a filter model for the $\lambda\mu$-calculus which is an instance of Streicher and Reus's continuation model, and a type assignment system such that the set of types that can be given to a term coincides with its denotation in the model. The type theory and the type assignment system can be viewed as the logic for reasoning about the computational meaning of $\lambda\mu$-terms, much as is the case for λ-calculus.

By restricting the assignment system to a subset of the intersection types we have obtained an assignment system where exactly the strongly normalising $\lambda\mu$-terms are typeable.

Finally, we have given a translation of intersection types into logical formulas and proved that if a term is typeable in Parigot's type assignment system for $\lambda\mu$, then it is typeable by its translation in the restricted intersection type system. As a by-product we have a new proof that proof terms in the $\lambda\mu$-calculus are strongly normalising.

References

[1] S. Abramsky. Domain Theory in Logical Form. *Annals of Pure and Applied Logic*, 51:1–77, 1991.

[2] F. Alessi and P. Severi. Recursive Domain Equations of Filter Models. In V. Geffert, J. Karhumäki, A. Bertoni, B. Preneel, P. Návrat, and M. Bieliková, editors, *SOFSEM 2008: Theory and Practice of Computer Science, 34th Conference on Current Trends in Theory and Practice of Computer Science*, volume 4910 of *Lecture Notes in Computer Science*, pages 124–135. Springer Verlag, 2008.

[3] R. Amadio and P.-L Curien. *Domains and lambda-calculi*. Cambridge University Press, 1998.

[4] Z.M. Ariola and H. Herbelin. Minimal Classical Logic and Control Operators. In J.C.M. Baeten, J.K. Lenstra, J. Parrow, and G.J. Woeginger, editors, *Proceedings of Automata, Languages and Programming, 30th International Colloquium, ICALP 2003*, volume 2719 of *Lecture Notes in Computer Science*, pages 871–885. Springer Verlag, 2003.

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

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

[7] 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.

[8] S. van Bakel. Completeness and Partial Soundness Results for Intersection & Union Typing for $\overline{\lambda}\mu\tilde{\mu}$. *Annals of Pure and Applied Logic*, 161:1400–1430, 2010.

[9] S. van Bakel. Sound and Complete Typing for $\lambda\mu$. In *Proceedings of 4th International Workshop* Intersection Types and Related Systems *(ITRS'10)*, volume 45 of *Electronic Proceedings in Theoretical Computer Science*, pages 31–44, 2010.

[10] S. van Bakel. Strict intersection types for the lambda calculus. *ACM Computing Surveys*, 43(3):20, 2011.

[11] S. van Bakel, F. Barbanera, and U. de'Liguoro. A Filter Model for $\lambda\mu$. In L. Ong, editor, *Proceedings of* 10th International Conference on Typed Lambda Calculi and Applications *(TLCA'11)*, volume 6690 of *Lecture Notes in Computer Science*, pages 213–228. Springer Verlag, 2011.

[12] S. van Bakel, F. Barbanera, and U. de'Liguoro. Characterisation of Strongly Normalising $\lambda\mu$-Terms. In S. Graham-Lengrand and L. Paolini, editors, *Proceedings of 5th International Workshop* Intersection Types and Related Systems *(ITRS'12)*, volume 121 of *Electronic Proceedings in Theoretical Computer Science*, pages 1–17, 2013.

[13] S. van Bakel and M. Dezani-Ciacaglini. Characterising Strong Normalisation for Explicit Substitutions. In *Proceedings of Latin American Theoretical Informatics (LATIN'02)*, volume 2286 of *Lecture Notes in Computer Science*, pages 356–370. Springer Verlag, 2002.

[14] S. van Bakel and M.G. Vigliotti. An Output-Based Semantics of $\lambda\mu$ with Explicit Substitution in the π-calculus - Extended Abstract. In J.C. M. Baeten, T. Ball, and F.S. de Boer, editors, *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference (TCS 2012)*, volume 7604 of *Lecture Notes in Computer Science*, pages 372–387. Springer Verlag, September 26-28 2012.

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

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

[17] 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.

[18] H. Barendregt, W. Dekkers, and R. Statman. *Lambda Calculus with Types*. Perspectives in logic. Cambridge University Press, 2013.

[19] 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.
[20] M. Coppo and M. Dezani-Ciancaglini. A New Type Assignment for Lambda-Terms. *Archive für Mathematischer Logic und Grundlagenforschung*, 19:139–156, 1978.
[21] M. Coppo and M Dezani-Ciancaglini. An Extension of the Basic Functionality Theory for the λ-Calculus. *Notre Dame journal of Formal Logic*, 21(4):685–693, 1980.
[22] M. Coppo, M. Dezani-Ciancaglini, F. Honsell, and G. Longo. Extended type structures and filter lambda models. In G. Lolli, G. Longo, and A. Marcja, editors, *Logic Colloquium 82*, pages 241–262. North-Holland, 1984.
[23] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal type schemes and λ-calculus semantics. In J.R. Hindley and J.P. Seldin, editors, *To H.B. Curry, Essays in combinatory logic, lambda-calculus and formalism*, pages 535–560. Academic press, New York, 1980.
[24] 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 233–223. ACM, 2000.
[25] R. David and W. Py. $\lambda\mu$-Calculus and Böhm's Theorem. *Journal of Symbolic Logic*, 66(1):407–413, 2001.
[26] Ugo de'Liguoro. The Approximation Theorem for the $\Lambda\mu$-Calculus. *MSCS*, to appear. Electronic version DOI: http://dx.doi.org/10.1017/S0960129515000286, 2015.
[27] M. Dezani-Ciancaglini, F. Honsell, and F. Alessi. A complete characterization of complete intersection-type preorders. *ACM Transactions on Computational Logic*, 4(1):120–147, 2003.
[28] 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)*, volume 136 of *Electronic Notes in Theoretical Computer Science*, pages 228–226, 2004.
[29] 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.
[30] M. Felleisen. *The Calculi of λ-v-CS Conversion: A Syntactic Theory of Control and State in Imperative Higher-Order Programming Languages*. PhD thesis, Department of Computer Science, Indiana University, Bloomington, Indiana, August 1987.
[31] M. Felleisen, D.P. Friedman, E. Kohlbecker, and B. Duba. Reasoning with Continuations. In *Proceedings of the First Symposium on Logic in Computer Science*, pages 131–141. IEEE, 1986.
[32] G. Gentzen. Investigations into logical deduction. In *The Collected Papers of Gerhard Gentzen*. Ed. M.E. Szabo, North Holland, 68ff (1969), 1935.
[33] S. Ghilezan. Strong normalization and typability with intersection types. *Notre Dame journal of Formal Logic*, 37(1):44–52, 1996.
[34] T. Griffin. A formulae-as-types notion of control. In *Proceedings of the 17th Annual ACM Symposium on Principles Of Programming Languages (POPL'17)*, pages 47–58, 1990.
[35] 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.
[36] M. Hofmann and Th. Streicher. Continuation models are universal for $\lambda\mu$-calculus. In *Proceedings of Logic in Computer Science (LICS)*, pages 387–397, 1997.
[37] K. Kikuchi and T. Sakurai. A Translation of Intersection and Union Types for the $\lambda\mu$ Calculus. Presented at *Classical Logic and Computation* 2014 (CL&C'14), Vienna, Austria, 2014.
[38] J.-L Krivine. *Lambda calculus, types and models*. Ellis Horwood, 1993.
[39] Y. Lafont, B. Reus, and Th. Streicher. Continuation Semantics or Expressing Implication by Negation. Report 9321, Ludwig-Maximilians-Universität, München, 1993.
[40] Ugo de'Liguoro. The Approximation Theorem for the $\Lambda\mu$-Calculus. *MSCS*, to appear. Electronic version DOI: http://dx.doi.org/10.1017/S0960129515000286, 2015.
[41] K. Nakazawa. Extensional models of typed lambda-mu calculus. Presented at International Workshop *Intersection Types and Related Systems* (ITRS'14), 2014.
[42] K. Nakazawa and S. Katsumata. Extensional Models of Untyped Lambda-mu Calculus. In H. Geuvers and U. de'Liguoro, editors, *Proceedings Fourth Workshop on Classical Logic and Computation, CL&C 2012*, volume 97 of *Electronic Proceedings in Theoretical Computer Science*, pages 35–47, 2012.
[43] K. Nakazawa and T. Nagai. Reduction System for Extensional Lambda-mu Calculus. In *Proceedings of Rewriting and Typed Lambda Calculi - Joint International Conference, RTA-TLCA 2014, Held as Part of the Vienna Summer of Logic, VSL 2014*, volume 8560 of *Lecture Notes in Computer Science*, pages

349–363, 2014.

[44] C.-H. Ong. A Semantic View of Classical Proofs: Type-Theoretic, Categorical, and Denotational Characterizations (Preliminary Extended Abstract). In *Proceedings of the 11th Annual Symposium on Logic in Computer Science (LICS '98)*, pages 230–221, 1996.

[45] 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*, pages 215–227, 1997.

[46] 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.

[47] M. Parigot. Proofs of Strong Normalisation for Second Order Classical Natural Deduction. *Journal of Symbolic Logic*, 62(4):1461–1479, December 1997.

[48] M. Parigot. On the Computational Interpretation of Negation. In *CSL*, volume 1862 of *Lecture Notes in Computer Science*, pages 472–484. Springer Verlag, 2000.

[49] 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.

[50] W. Py. *Confluence en $\lambda\mu$-calcul*. PhD thesis, Université de Savoie, 1998. PhD thesis.

[51] A. Saurin. On the Relations between the Syntactic Theories of $\lambda\mu$-Calculi. In M. Kaminski and S. Martini, editors, *Computer Science Logic, 22nd International Workshop, CSL 2008, 17th Annual Conference of the EACSL*, volume 5213 of *Lecture Notes in Computer Science*, pages 154–168. Springer Verlag, 2008.

[52] Th. Streicher and B. Reus. Classical logic: Continuation Semantics and Abstract Machines. *Journal of Functional Programming*, 11(6):543–572, 1998.

[53] W.W. Tait. Intensional interpretation of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):198–223, 1967.