ACADEMIC PRESS

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Information and Computation 186 (2003) 285–318

Information and Computation

www.elsevier.com/locate/ic

Infinite intersection types

Marcello M. Bonsangue *,[1] and Joost N. Kok

Leiden Institute of Advanced Computer Science, Leiden University, Leiden, Netherlands

Received 28 March 2002; revised 3 March 2003

Abstract

A type theory with *infinitary* intersection and union types for an extension of the λ-calculus is introduced. Types are viewed as upper closed subsets of a Scott domain and intersection and union type constructors are interpreted as the set-theoretic intersection and union, respectively, even when they are not finite. The assignment of types to λ-terms extends naturally the basic type assignment system. We prove soundness and completeness using a generalization of Abramsky's finitary logic of domains. Finally we apply the framework to applicative transition systems, obtaining a sound a complete infinitary intersection type assignment system for the lazy λ-calculus.

1. Introduction

Intersection types were introduced in [13] as an extension of Curry's simply typed λ-calculus. A remarkable property of intersection types is their ability to express computational properties of λ-terms, such as termination properties (e.g., strong normalization [19] and weak head normalization [2]) and reduction properties (e.g., reduction to a closed term [14]). In [8], an intersection type system was proved to be sound and complete with respect to Scott's set-theoretic semantics for simple types.

Since then, several variations of the original intersection type discipline have been explored, including enrichment with new type constructors like the union of types [5,6,18], conservative extensions like infinite intersection types [5,17] and modifications inducing more computationally adequate models, like the lazy λ-calculus [2].

* Corresponding author. Fax: +31-71-527-69-85.
E-mail addresses: marcello@liacs.nl (M.M. Bonsangue), joost@liacs.nl (J.N. Kok).
[1] The research of Dr. Bonsangue has been made possible by a fellowship of the Royal Netherlands Academy of Arts and Sciences.

0890-5401/$ - see front matter © 2003 Elsevier Science (USA). All rights reserved.
doi:10.1016/S0890-5401(03)00143-3

Leivant showed in [17] that considering infinite intersection types have several advantages over the ordinary intersection type discipline. First, infinite intersection types form a framework in which several typing mechanisms, such as parametric polymorphisms, stratified parametric polymorphisms and recursive types can be interpreted. Second, there is a relationship between the size of the types allowed and the computational nature of the functions representable by finite λ-terms. And finally, infinite intersection types allow for typing in a uniform way a number of interesting classes of λ-terms, like those representing the Berarducci numerals (a result independently obtained also in [5]).

Although infinite intersection types conceptually express better the idea of multiple typing, not much work has been done in generalizing the filter model of [8] toward a complete set-theoretical model for infinite intersection types. The reason is maybe due to the fact that a study of infinite intersection types cannot be carried out easily without considering infinite union types. As a consequence, a set theoretic-model has to be a completely distributive lattice, in contrast to the fact that canonical models for λ-calculi, like some Scott domains, induce topologies that do not satisfy the complete distributivity laws.

In this paper we solve this open problem by proving soundness and completeness of a type system for a collection of λ-calculi, in which arbitrary intersection and union types are allowed. Types are interpreted as upper closed subsets of a Scott domain, the so-called saturated sets. The meanings of the intersection and union type constructors are the set-theoretic intersection and the union, respectively, even when they are not finite. Types are assigned to terms in a such way that the interpretation of a term belongs to the interpretation of all types it is assigned to. The completeness result for our infinitary type system is based on the connection between Abramsky's domain logic and the Scott compact opens of a suitable domain [3].

This work fits into our studies toward an infinitary logic of domains. In [9,11] we focused on an infinitary logical form for Plotkin's powerdomain construction, while in this paper we concentrate on the function space construction. The key ingredients are a Stone-like duality for $\mathcal{T}_0$ topological spaces [10] and a characterization of sober spaces in terms of their completely distributive lattice of saturated sets (i.e., upper closed sets with respect to the specialization preorder induced by the opens) [11]. These results allow us to freely extend Abramsky's finitary logic of compact opens [3] to the infinitary logic of the saturated sets.

Our framework is very general, in the sense that our language of types and terms can be used to model several extensions and interpretation of the λ-calculus. It is also very expressive, because each element of a Scott domain can be assigned to a type representing its upper closure. Indeed, the intersection of all compact opens to which an element (not necessarily compact) belong coincides with its upper closure. This fact implies that, in our framework, each term has a minimal type (up to equivalence). The presence of infinite union types allows to denote any upper closed subset of a domain, as any upper closed set is just the union of the upper closure of all its inhabitants. As a consequence, some basic types like Nat denoting the infinite set of all natural numbers (without any order) are expressible in our framework (in contrast to Abramsky's domain logic that can express only compact opens and therefore one would need to add an extra bottom element to the set of all natural numbers).

In this paper we apply our framework to describe an infinitary intersection type system for the collection of all applicative transition systems, an operational model of lazy functional languages. The reason for our choice are twofold. The first reason is simplicity. In fact, to obtain a non-trivial model for the λ-calculus one needs to consider a fixed collection of basic types. To fall into the scope of the general theory developed by Abramsky and in this paper, a logical presentation of the associated Scott

domains would use a open domain expression with free variables assigned to Scott domains denoting the basic types. This in contrast to the simple (and closed) domain expression associated to the lazy λ-calculus. The second reason is re-usability of results already obtained by Abramsky in applying his logic of domain to the lazy lambda calculus [1].

The rest of the paper is organized as follows. In Section 2 we recall some properties of distributive lattices and topological spaces we will need in the rest of the paper. In Section 3 we present a framework for connecting Scott domains with infinitary type theories for λ-calculi. It consists of a language of domain expressions (called type expressions in [3]) interpreted both as Scott domains and as infinitary type languages. For each of those type languages, an axiomatization of the subtype judgment is given and proved (in the Sections 3.1 and 3.2) sound and complete with respect to subset inclusion of upper close subsets of the associated Scott domains. In Section 4 a language of terms is introduced for each domain expression. Basically the languages are extensions of the λ-calculus with terms for pairing, lifting, unfolding, etc. A sound and complete type assignment system is presented with respect to the interpretation of terms as elements of Scott domains. In Section 5 we apply our framework to applicative transition systems, showing the sound and completeness of an infinitary intersection type system and of its type assignment system. We conclude in Section 6 with some final remarks and possible applications of our results for studying properties of recursive types as well as of existential and universal types.

2. Mathematical preliminaries

We start with some basic definitions and facts about distributive lattices. We assume that the reader is familiar with basic concepts from domain theory.

A lattice L is a poset with join and meet for every finite subset. Below we write $\bigvee S$ and $x \vee y$ for the join of an arbitrary subset S of L and the binary join of two elements in L, respectively, if they exist. Dually, we denote by $\bigwedge S$ and $x \wedge y$ the meet of an arbitrary subset S of L and the binary meet of two elements in L, respectively. A lattice L is called *distributive* if

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$$

for all a, b and c in L. The above equation holds for a lattice if and only if it does so for its dual [22], where we substitute meets for joins and joins for meets. An element a of a lattice L is called *coprime* if $a \leqslant b_1 \vee b_2$ implies $a \leqslant b_1$ or $a \leqslant b_2$.

If the lattice L has join for arbitrary subsets, and not just finite ones then it is said to be *complete*. A complete lattice L that satisfies the infinite distributive law

$$a \wedge \bigvee S = \bigvee \{a \wedge s \mid s \in S\},$$

for all $a \in L$ and all subsets $S \subseteq L$, is called a *frame*.

A complete lattice L is *completely distributive* if, for all sets $\mathcal{A}$ of subsets of L,

$$\bigwedge \left\{ \bigvee S \mid S \in \mathcal{A} \right\} = \bigvee \left\{ \bigwedge f(\mathcal{A}) \mid f \in \Phi(\mathcal{A}) \right\},$$

where $f(\mathcal{A})$ denotes the set $\{f(S) \mid S \in \mathcal{A}\}$ and $\Phi(\mathcal{A})$ is the set of all total functions $f : \mathcal{A} \to \bigcup \mathcal{A}$ such that $f(S) \in S$ for all $S \in \mathcal{A}$. The above equation holds for a lattice if and only if it does so for its

dual [21], where we substitute meets for joins and joins for meets. Clearly every completely distributive lattice is a frame, and every frame is a distributive lattice.

A *topology* on a set X is a collection $\mathcal{O}(X)$ of subsets of X closed under finite intersections and arbitrary unions. A set $o \in \mathcal{O}(X)$ is called *open*. The collection $\mathcal{O}(X)$ of open subsets of a topological space X ordered by subset inclusion is a frame.

A subset k of a topological space X is said to be *compact* if for every directed subset S of $\mathcal{O}(X)$, $k \subseteq \bigcup S$ implies $k \leqslant o$ for some $o \in S$. The set of all compact open subset of X is denoted by $\mathcal{KO}(X)$. A topological space X is said to be *spectral* if the set $\mathcal{KO}(X)$ of compact open subsets is closed under finite intersections and finite unions, and for all opens o it holds $o = \bigcup\{k \in \mathcal{KO}(X) \mid k \subseteq o\}$. If X is a spectral space, then $\mathcal{KO}(X)$ ordered by subset inclusion is a distributive lattice.

A subset q of a topological space X is said to be *saturated* if $q = \bigcap\{o \in \mathcal{O}(X) \mid q \subseteq o\}$. The collection of the saturated subsets of X is denoted by $\mathcal{Q}(X)$, and it is closed under arbitrary intersections and arbitrary unions.

Spectral spaces are *coherent*, meaning that the collection of their compact saturated sets is closed under arbitrary intersections and finite unions, and open sets are the directed union of all compact saturated sets which are subset of them. For every coherent space X, it holds that saturated sets are unions of compact saturated sets, and compact saturated sets are intersections of compact opens.

For example, if $\mathcal{D}$ is a Scott domain [23] taken with the Scott topology, then $\mathcal{D}$ is spectral (and hence coherent). Coprime elements of $\mathcal{KO}(X)$ are the upper closure of a single compact element of $\mathcal{D}$, compact opens are the upper closure of finite subsets of compact elements of $\mathcal{D}$, compact saturated are the upper closure of finite subsets of any elements of $\mathcal{D}$, and finally saturated sets are the upper closure of any subsets of any elements of $\mathcal{D}$.

Theorem 2.1. *For a Scott domain $\mathcal{D}$ it holds that the following are equivalents*:
1. *the frame of open sets $\mathcal{O}(\mathcal{D})$ is free over the distributive lattice of compact opens $\mathcal{KO}(X)$*;
2. *the completely distributive lattice of saturated sets $\mathcal{Q}(\mathcal{D})$ is free over the frame of opens $\mathcal{O}(\mathcal{D})$*.

Proof. Scott domains taken with the Scott topology are $\mathcal{T}_0$ and spectral. The second part of this statement is due to Plotkin [20, Chapter 8, Theorem 6 and Exercise 77(1)] who proved that for domains, like Scott domains, satisfying two of the three conditions characterizing SFP domains, the intersection of Scott compact sets is again Scott compact. The latter is the property characterizing the spectral spaces. Thus Scott domains taken with the Scott topology are sober spaces [16, Corollary II.2.11], from which it follows that $\mathcal{O}(X)$ is the free distributive lattice over $\mathcal{KO}(X)$ [16, Corollary II.2.11] and that $\mathcal{Q}(X)$ is the free completely distributive lattice over $\mathcal{O}(X)$ [11, Theorem 3.6]. □

We conclude this section by stating a restricted version of the Scott-open filter theorem (also known as the Hofmann–Mislove theorem [15]). This theorem will play an important role in the proof of soundness of a type system with infinite intersection types. A proof of this proposition for the general case of X a sober space can be found in [4, Corollary 7.2.11].

Proposition 2.2. *For a spectral space X, open subset o of X, and filtered collection S of compact saturated subsets of X,*

$$\bigcap S \subseteq o \iff \exists k \in S.k \subseteq o.$$

3. Domain theory in infinitary logical form

In this section we present a framework for connecting λ-calculi with intersection type theories. It is an extension of the work initiated by Abramsky [1,3] since the theories used by Abramsky for the type theoretical interpretation of the language are finite, whereas we will consider also infinitary theories.

First we consider a language of domain expressions (called type expressions in [3]) with several constructors like product, sum, lift and function space. For simplicity, we do not consider here the coalesced sum, the strict function space, and the powerdomain constructors. The Plotkin Powerdomain has already been treated separately in [11] and can be introduced without much changes in the framework. The treatment of the strict function space and the coalesced sum constructors would require some more technicalities (but conceptually they could be treated here as well).

Definition 3.1. The syntax of a *domain expression* is defined by the following grammar:

$$\sigma ::= \mathbf{1} \mid e \mid \sigma \times \sigma \mid \sigma + \sigma \mid \sigma \to \sigma \mid (\sigma)_\perp \mid \mathsf{rec}e.\sigma$$

where e ranges over *expression variables*.

Domain expressions can be interpreted as Scott domains. The interpretation is as expected, in the sense that expression constructors are interpreted denotationally as the standard domain constructors in the category **SDom** with Scott domains as objects and Scott continuous function as morphisms. Define the set of *domain environments* $DEnv$ to be the set of all functions mapping expression variables to Scott domains. For each domain expression σ and domain environment $\varepsilon \in DEnv$, define a Scott domain $\mathcal{D}(\sigma)_\varepsilon$ inductively by interpreting **1** as the one element domain, an expression variable e as the Scott domain $\varepsilon(e)$, $\sigma_1 \times \sigma_2$ as the Cartesian product of $\mathcal{D}(\sigma_1)_\varepsilon$ and $\mathcal{D}(\sigma_2)_\varepsilon$ endowed with the pairwise order (that is, their product in **SDom**), $\sigma_1 + \sigma_2$ as the lifting of the disjoint union of $\mathcal{D}(\sigma_1)_\varepsilon$ and $\mathcal{D}(\sigma_2)_\varepsilon$ (that is, their coproduct in **SDom**), $\sigma_1 \to \sigma_2$ as the set of all Scott continuous functions from $\mathcal{D}(\sigma_1)_\varepsilon$ to $\mathcal{D}(\sigma_2)_\varepsilon$ endowed with the pointwise order, $(\sigma)_\perp$ as the lifting of the domain $\mathcal{D}(\sigma)_\varepsilon$, and $\mathsf{rec}e.\sigma$ as the canonical solution of the recursive domain equation $X = F(X)$, where F is the endofunctor in the category **SDom** mapping a Scott domain $\mathcal{D}$ to $\mathcal{D}(\sigma)_{\varepsilon[\mathcal{D}/e]}$ (here the expression $\varepsilon[\mathcal{D}/e]$ denotes the environment which maps e to $\mathcal{D}$ and at all other expression variables coincides with ε).

Using the above language of expressions, we can define models of several λ-calculi. For example, the domain expression $\sigma = \mathsf{rec}\ e.(e \to e) + (e \times e)$ induces the Scott model $\mathcal{D}(\sigma)$ of the ordinary λ-calculus extended with pairs. Λ-terms are then mapped to elements of $\mathcal{D}(\sigma)$.

In Section 5 we treat the lazy λ-calculus more extensively. The domain expression associated with it is $\mathsf{rec}e.(e \to e)_\perp$ [2].

For each domain expression σ we define a second interpretation $\mathbf{T}(\sigma)$ by means of a type language. Types are seen as syntactical entities built by means of an infinitary intersection type constructor '$\bigwedge$' and an infinitary union type constructor '$\bigvee$', and other type constructors associated with those used in the domain expression σ (e.g. the arrow type constructor '$\to$', or the lift type constructor '$(-)_\perp$'). Our approach is similar to the one followed by Abramsky for his logical interpretation of domains. However, it is important to stress here that the theories used by Abramsky are finite and describe the logics of

$$\frac{\{\phi_i \in \mathbf{T}(\sigma)_\varepsilon\}_{i\in I}}{\bigvee_{i\in I}\phi_i \in \mathbf{T}(\sigma)_\varepsilon} \qquad \frac{\{\phi_i \in \mathbf{T}(\sigma)_\varepsilon\}_{i\in I}}{\bigwedge_{i\in I}\phi_i \in \mathbf{T}(\sigma)_\varepsilon} \qquad \frac{u \in \mathcal{KO}(\varepsilon(e))}{u \in \mathbf{T}(e)_\varepsilon}$$

$$\frac{\phi_1 \in \mathbf{T}(\sigma_1)_\varepsilon \quad \phi_2 \in \mathbf{T}(\sigma_2)_\varepsilon}{\phi_1 \times \phi_2 \in \mathbf{T}(\sigma_1 \times \sigma_2)_\varepsilon} \qquad \frac{\phi \in \mathbf{T}(\sigma_1)_\varepsilon}{\phi + \mathsf{Bot} \in \mathbf{T}(\sigma_1 + \sigma_2)_\varepsilon} \qquad \frac{\phi \in \mathbf{T}(\sigma_2)_\varepsilon}{\mathsf{Bot} + \phi \in \mathbf{T}(\sigma_1 + \sigma_2)_\varepsilon}$$

$$\frac{\phi_1 \in \mathbf{T}(\sigma_1)_\varepsilon \quad \phi_2 \in \mathbf{T}(\sigma_2)_\varepsilon}{\phi_1 \to \phi_2 \in \mathbf{T}(\sigma_1 \to \sigma_2)_\varepsilon} \qquad \frac{\phi \in \mathbf{T}(\sigma)_\varepsilon}{(\phi)_\perp \in \mathbf{T}((\sigma)_\perp)_\varepsilon} \qquad \frac{\phi \in \mathbf{T}(\sigma[\mathsf{rec}e.\sigma/e])_\varepsilon}{\phi \in \mathbf{T}(\mathsf{rec}e.\sigma)_\varepsilon}$$

Fig. 1. Well-formed types.

compact open sets, whereas our type language is infinite and capable to describe the logics of the upper closed subsets of a Scott domain.

Definition 3.2. For every domain expression σ and domain environment ε, the class $(\phi \in)\mathbf{T}(\sigma)_\varepsilon$ is defined inductively by the inference rules given in Fig. 1, where $\sigma[\tau/e]$ is the domain expression obtained by substituting τ for e in σ, and I is an arbitrary index set. If $I = \emptyset$ then we write Top for $\bigwedge_{i\in I}\phi_i$ and we write Bot for $\bigvee_{i\in I}\phi_i$.

For example, $\mathsf{Top} \in \mathbf{T}(\sigma)$ for all domain expressions σ, whereas $\mathsf{Top} \to \mathsf{Bot} \in \mathbf{T}(\sigma \to \sigma)$. Free domain variables are associated to the compact opens of the Scott topology of the Scott domain $\varepsilon(e)$. Assume ε is a domain environment mapping the domain variable e to the Scott domain $(\mathsf{Nat})_\perp$ of flat natural numbers. Then any finite subset of the natural numbers is compact open in the Scott topology of $(\mathsf{Nat})_\perp$, and thus is a type in $T(e)_\varepsilon$. The type Nat of all natural numbers can be modelled as the infinite union of all finite subsets of natural numbers, that is, $\mathsf{Nat} = \bigvee \mathbf{T}(e)_\varepsilon$. Note that there are only set-many types in $T(e)_\varepsilon$, thus the above union is well defined. Similarly, one could define the type Pos of the positive natural numbers by setting $\mathsf{Pos} = \mathsf{Nat} \setminus \{0\}$. Because $\mathsf{Pos} \subseteq \mathsf{Nat}$ we will see in Fig. 5 that Pos will actually be a subtype of Nat. Note that the two types above cannot be described in the original framework of Abramsky, as they denote infinite unions of compact opens.

For any domain expression σ, we denote by $\mathbf{T}_0(\sigma)_\varepsilon$ the sub-language of $\mathbf{T}(\sigma)_\varepsilon$ obtained by restricting to finite intersection type constructors and finite union type constructors.

Semantics of types in $\mathbf{T}(\sigma)_\varepsilon$ is given in terms of the Scott domain $\mathcal{D}(\sigma)_\varepsilon$.

Definition 3.3. The denotational meaning of the types in $\mathbf{T}(\sigma)_\varepsilon$ is given in Fig. 2 by means of an interpretation relation $\models_{\sigma,\varepsilon} \subseteq \mathcal{D}(\sigma)_\varepsilon \times \mathbf{T}(\sigma)_\varepsilon$, for every domain expression σ and domain environment ε.

In Fig. 2, lpr and rpr are the canonical left and right projections from $\mathcal{D}(\sigma_1 \times \sigma_2)$ to $\mathcal{D}(\sigma_1)$ and $\mathcal{D}(\sigma_2)$, respectively. Similarly, inl and inr are the canonical left and right injections from $\mathcal{D}(\sigma_1)$ and $\mathcal{D}(\sigma_2)$ to $\mathcal{D}(\sigma_1 + \sigma_2)$, respectively. Finally, up is the canonical embedding of $\mathcal{D}(\sigma)$ into $\mathcal{D}((\sigma)_\perp)$, and $unfold$ is the canonical isomorphism between $\mathcal{D}(\mathsf{rec}e.\sigma)$ and $\mathcal{D}(\sigma[\mathsf{rec}e.\sigma/e])$, with $fold$ as inverse.

In the following we will omit the domain environment information when not necessary and write $[\![\phi]\!]_\sigma$ for the set $\{d \in \mathcal{D}(\sigma) \mid d \models_\sigma \phi\}$, where $\phi \in \mathbf{T}(\sigma)$.

Lemma 3.4. *For each domain expression σ and type $\phi \in \mathbf{T}(\sigma)$, $[\![\phi]\!]_\sigma$ is an upper closed subset of $\mathcal{D}(\sigma)$, so that $[\![-]\!]_\sigma$ is a function from $\mathbf{T}(\sigma)$ to $\mathcal{Q}(\mathcal{D}(\sigma))$.*

$$
\begin{array}{lcl}
d \models_{\sigma,\varepsilon} \bigvee_{i\in I} \phi_i & \iff & \exists i \in I.\ d \models_{\sigma,\varepsilon} \phi_i \\
d \models_{\sigma,\varepsilon} \bigwedge_{i\in I} \phi_i & \iff & \forall i \in I.\ d \models_{\sigma} \phi_i \\
d \models_{e,\varepsilon} u & \iff & d \in u \\
d \models_{\sigma_1\times\sigma_2,\varepsilon} \phi_1 \times \phi_2 & \iff & lpr(d) \models_{\sigma_1,\varepsilon} \phi_1 \text{ and } rpr(d) \models_{\sigma_2,\varepsilon} \phi_2. \\
inl(d) \models_{\sigma_1+\sigma_2,\varepsilon} \phi_1 + \mathsf{Bot} & \iff & d \models_{\sigma_1,\varepsilon} \phi_1 \\
inr(d) \models_{\sigma_1+\sigma_2,\varepsilon} \mathsf{Bot} + \phi_2 & \iff & d \models_{\sigma_2,\varepsilon} \phi_2 \\
f \models_{\sigma_1\to\sigma_2,\varepsilon} \phi_1 \to \phi_2 & \iff & \forall d \in \mathcal{D}(\sigma_1)_\varepsilon.\ d \models_{\sigma_1,\varepsilon} \phi_1 \text{ implies } f(d) \models_{\sigma_2,\varepsilon} \phi_2. \\
up(d) \models_{(\sigma)_\perp,\varepsilon} (\phi)_\perp & \iff & d \models_{\sigma,\varepsilon} \phi \\
d \models_{\mathsf{rec}e.\sigma,\varepsilon} \phi & \iff & unfold(d) \models_{\sigma[\mathsf{rec}e.\sigma/e],\varepsilon} \phi
\end{array}
$$

Fig. 2. Semantics of types.

Proof. The proof proceeds by induction on the structure of ϕ. We treat only one case. If $f \models_{\sigma_1\to\sigma_2} \phi_1 \to \phi_2$ and $f \leqslant g$ in $\mathcal{D}(\sigma_1 \to \sigma_2)$ then, for all $d \in \mathcal{D}(\sigma_1)$, $f(d) \leqslant g(d)$ in $\mathcal{D}(\sigma_2)$. By the induction hypothesis it follows that $g(d) \models_{\sigma_2} \phi_2$ for all $d \in [\![\phi_1]\!]_{\sigma_1} \subseteq \mathcal{D}(\sigma_1)$. Hence also $g \models_{\sigma_1\to\sigma_2} \phi_1 \to \phi_2$.

Note that recursive domain expressions do not introduce structure on the respective type. Thus, in order to prove $d' \models_{\mathsf{rec}e.\sigma} \phi$ when $d \models_{\mathsf{rec}e.\sigma} \phi$ and $d \leqslant d'$, it is enough to prove $unfold(d') \models_{\sigma[\mathsf{rec}e.\sigma/e]} \phi$ because $unfold(d) \models_{\sigma[\mathsf{rec}e.\sigma]} \phi$ and $unfold{:}\mathcal{D}(\mathsf{rec}e.\sigma) \to \mathcal{D}(\sigma[\mathsf{rec}e.\sigma/e])$ is the order isomorphism arising form the solution of the domain equation $e = \sigma(e)$. □

We conclude this section by presenting a formal proof system for deriving *assertions* over $\mathbf{T}(\sigma)$ of the form $\phi_1 \leqslant_\sigma \phi_2$ and $\phi_1 =_\sigma \phi_2$. Their semantics is defined by

$$
\begin{array}{l}
\models \phi_1 \leqslant_\sigma \phi_2 \iff \forall d \in \mathcal{D}(\sigma).\, d \models_\sigma \phi_1 \text{ implies } d \models_\sigma \phi_2, \\
\models \phi_1 =_\sigma \phi_2 \iff \models \phi_1 \leqslant_\sigma \phi_2 \text{ and } \models \phi_2 \leqslant_\sigma \phi_1.
\end{array}
$$

Our goal is to define a complete theory in the sense that $\phi_1 \leqslant_\sigma \phi_2$ is provable in the theory if and only if $\models \phi_1 \leqslant_\sigma \phi_2$. We will need an auxiliary *coprimeness predicate* $\mathsf{C}(\phi)$ on types ϕ of $\mathbf{T}(\sigma)$. Types for which the coprimeness predicate holds will be identified with coprime elements of the lattice $\mathcal{KO}(\mathcal{D}(\sigma))$. They will play a role in validating the axiom about the distribution of union types over the '$\to$' constructor when appearing on its right-hand side. Such an axiom is not valid in general [23]. Fig. 3 presents a formal system for the *coprimeness judgment*.

Rule [C$-\to$] bears great resemblance to the definition of the finite elements of the function space between two Scott domains, the so-called step functions [23]. Indeed, types for which the coprimeness predicate holds are precisely the upper closure of the compact (finite) elements of the respective Scott domain [3].

Note that $\mathsf{C}(\phi)$ implies that no disjunctions (up-to $=$) occur in ϕ, whereas the type theory Π given in [6] uses a predicate $\mathsf{P}(\phi)$ which essentially allows for disjunctions only if they occur on the left-hand side of the '$\to$' constructors.

$$[\mathsf{C}-\mathsf{Top}]\ \ \mathsf{C}(\mathsf{Top}) \qquad [\mathsf{C}-=]\ \ \frac{\mathsf{C}(\phi_1)\quad \phi_1 =_\sigma \phi_2}{\mathsf{C}(\phi_2)} \qquad [\mathsf{C}-\varepsilon]\ \ \frac{u \text{ coprime in } \mathcal{KO}(\varepsilon(e))}{\mathsf{C}(u)}$$

$$[\mathsf{C}-\times]\ \ \frac{\mathsf{C}(\phi_1)\quad \mathsf{C}(\phi_2)}{\mathsf{C}(\phi_1\times\phi_2)} \qquad [\mathsf{C}-+]\ \ \frac{\mathsf{C}(\phi)}{\mathsf{C}(\phi+\mathsf{Bot})} \qquad [+-\mathsf{C}]\ \ \frac{\mathsf{C}(\phi)}{\mathsf{C}(\mathsf{Bot}+\phi)}$$

$$[\mathsf{C}-(-)_\perp]\ \ \frac{\mathsf{C}(\phi)}{\mathsf{C}((\phi)_\perp)} \qquad [\mathsf{C}-\to]\ \ \frac{\{\mathsf{C}(\phi_i)\}_{i\in I}\quad \{\mathsf{C}(\phi'_i)\}_{i\in I}\quad (\forall J\subseteq I.\ \bigwedge_{j\in J}\phi'_j =_{\sigma'} \mathsf{Bot} \Longrightarrow \bigwedge_{j\in J}\phi_j =_\sigma \mathsf{Bot})}{\mathsf{C}(\bigwedge_{i\in I}\phi_i\to\phi'_i)}\ I \text{ finite}$$

Fig. 3. Coprimeness judgment.

The *logical axioms and rules* in Fig. 4 give to $\mathbf{T}(\sigma)$ the structure of a large completely distributive lattice. They hold for any domain expression σ, therefore we can omit the domain expression subscripts. The last axiom [*distr*] is about complete distributivity. Recall from Section 2 that we use $\Phi(\mathcal{S})$, for a set of sets $\mathcal{S}$, to denote the set of all total functions choosing an element from their input, that is, all functions $f:\mathcal{S}\to\bigcup\mathcal{S}$ such that $f(S)\in S$ for all $S\in\mathcal{S}$. Note that because of the presence of arbitrary choice functions, proofs involving the axiom of complete distributivity require the axiom of choice.

Finally, in Fig. 5 we give *structural axioms and rules* that relate the types to the logical structure. Again, we omit the domain expression subscripts from the assertions, when they can be deduced. All types in Fig. 5 are assumed well formed, thus implying that both u_1 and u_2 in rule $[e-\leqslant]$ are compact open subsets $\varepsilon(e)$, for the domain expression variable e used as subscript for the assertion.

Note that function types are contravariant in the argument and variant in the result, whereas the other type constructors '$\times$', '$+$' and '$(-)_\perp$' are variant. Furthermore, unions distribute over both sides of '$\times$' and '$+$', over '$(-)_\perp$', and over the left-hand side of the '$\to$' type construct, whereas intersections distribute over $\times$ and over the right-hand side of the '$\to$' type construct. Since $(\mathsf{Top})_\perp=\mathsf{Top}$ does not hold (the "extra" bottom element of a lifted domain will inhabits the right type but not the left one), only non-empty intersections distribute over both sides of '$+$' and '$(-)_\perp$'.

In the axioms $[\to-\bigvee]$ and $[\bigwedge-\to]$, we use *Fin*(S), for a set S, to denote the set of all finite subsets of S. Since an arbitrary union can always be decomposed as the directed union of finite unions and an

$$[\leq -ref]\ \ \phi\leq\phi \qquad [\leq -trans]\ \ \frac{\phi_1\leq\phi_2\quad \phi_2\leq\phi_3}{\phi_1\leq\phi_3}$$

$$[=-I]\ \ \frac{\phi_1\leq\phi_2\quad \phi_2\leq\phi_1}{\phi_1=\phi_2} \qquad [=-E]\ \ \frac{\phi_1=\phi_2}{\phi_1\leq\phi_2\quad \phi_2\leq\phi_1}$$

$$[\bigvee-I]\ \ \frac{\{\phi_i\leq\phi\}_{i\in I}}{\bigvee_{i\in I}\phi_i\leq\phi} \qquad [\bigvee-E]\ \ \frac{k\in I}{\phi_k\leq\bigvee_{i\in I}\phi_i}$$

$$[\bigwedge-I]\ \ \frac{\{\phi\leq\phi_i\}_{i\in I}}{\phi\leq\bigwedge_{i\in I}\phi_i} \qquad [\bigwedge-E]\ \ \frac{k\in I}{\bigwedge_{i\in I}\phi_i\leq\phi_k}$$

$$[\mathit{distr}]\ \ \bigwedge_{i\in I}\bigvee_{j\in J_i}\phi_{i,j}=\bigvee_{f\in\Phi(\{J_i\mid i\in I\})}\bigwedge_{i\in I}\phi_{i,f(i)}$$

Fig. 4. Subtype judgment: logical axioms and rules.

$$[e-\leq]\ \frac{u_1 \subseteq u_2}{u_1 \leq u_2} \qquad [\times-\leq]\ \frac{\phi_1 \leq \phi_2 \quad \phi_3 \leq \phi_4}{\phi_1 \times \phi_2 \leq \phi_2 \times \phi_4}$$

$$[+-\leq]\ \frac{\phi_1 \leq \phi_2}{\phi_1 + \mathsf{Bot} \leq \phi_2 + \mathsf{Bot}} \qquad [\leq -+]\ \frac{\phi_1 \leq \phi_2}{\mathsf{Bot} + \phi_1 \leq \mathsf{Bot} + \phi_2}$$

$$[\to -\leq]\ \frac{\phi_2 \leq \phi_1 \quad \phi_3 \leq \phi_4}{\phi_1 \to \phi_3 \leq \phi_2 \to \phi_4} \qquad [(-)_\perp -\leq]\ \frac{\phi_1 \leq \phi_2}{(\phi_1)_\perp \leq (\phi_2)_\perp}$$

$$[\bigvee -\times]\ (\bigvee_{i\in I} \phi_i \times \psi) = \bigvee_{i\in I}(\phi_i \times \psi) \qquad [\times - \bigvee]\ (\phi \times \bigvee_{i\in I} \psi_i) = \bigvee_{i\in I}(\phi \times \psi_i)$$

$$[\times - \bigwedge]\ \bigwedge_{i\in I}(\phi_i \times \psi_i) = (\bigwedge_{i\in I} \phi_i \times \bigwedge_{i\in I} \psi_i)$$

$$[\bigvee -+]\ (\bigvee_{i\in I} \phi_i + \mathsf{Bot}) = \bigvee_{i\in I}(\phi_i + \mathsf{Bot}) \qquad [+ - \bigvee]\ (\mathsf{Bot} + \bigvee_{i\in I} \phi_i) = \bigvee_{i\in I}(\mathsf{Bot} + \phi_i)$$

$$[\bigwedge -+]\ \frac{I \neq \emptyset}{(\bigwedge_{i\in I} \phi_i + \mathsf{Bot}) = \bigwedge_{i\in I}(\phi_i + \mathsf{Bot})} \qquad [+ - \bigwedge]\ \frac{I \neq \emptyset}{(\mathsf{Bot} + \bigwedge_{i\in I} \phi_i) = \bigwedge_{i\in I}(\mathsf{Bot} \times \phi_i)}$$

$$[+ - \wedge]\ (\mathsf{Bot} + \phi) \wedge (\psi + \mathsf{Bot}) = \mathsf{Bot}$$

$$[(-)_\perp - \bigvee]\ (\bigvee_{i\in I} \psi_i)_\perp = \bigvee_{i\in I}(\psi_i)_\perp \qquad [(-)_\perp - \bigwedge]\ \frac{I \neq \emptyset}{(\bigwedge_{i\in I} \phi_i)_\perp = \bigwedge_{i\in I}(\phi_i)_\perp}$$

$$[\to - \bigwedge]\ \bigwedge_{i\in I}(\phi \to \phi_i) \leq \phi \to \bigwedge_{i\in I} \phi_i \qquad [\bigwedge - \to]\ \bigwedge_{i\in I} \phi_i \to \phi \leq \bigvee_{J\in Fin(I)}(\bigwedge_{j\in J} \phi_j \to \phi)$$

$$[\bigvee - \to]\ \bigwedge_{i\in I}(\phi_i \to \phi) \leq \bigvee_{i\in I} \phi_i \to \phi \qquad [\to - \bigvee]\ \phi \to \bigvee_{i\in I} \phi_i \leq \bigvee_{J\in Fin(I)}(\phi \to \bigvee_{j\in J} \phi_j)$$

$$[\mathsf{C} - \to - \bigvee]\ \frac{\mathsf{C}(\phi)}{\phi \to \bigvee_{i\in I} \phi_i \leq \bigvee_{i\in I}(\phi \to \phi_i)}$$

Fig. 5. Subtype judgment: structural axioms and rules.

arbitrary intersection can always be decomposed as the filtered intersection of finite intersections, the axioms $[\to - \bigvee]$ and $[\bigwedge - \to]$ are about distributivity of directed unions and finite intersections over the left- and right-hand side of the $\to$ constructor, respectively. Arbitrary unions distribute over the right-hand side of the '$\to$' constructor (rule $[\mathsf{C} - \to - \bigvee]$) only if the right-hand side is a "coprime" type.

For any domain expression σ, if we restrict our proof system to the finitary language $\mathbf{T}_0(\sigma)$ then it coincides with Abramsky's domain logic for σ [3]. The axiom $[\to - \bigvee]$ and $[\bigwedge - \to]$ are new, and make sense only in a language with arbitrary disjunctions and conjunctions. Basically it is a statement of continuity of the '$\to$' type constructor, both on its left- and right-hand side. Note that because '$\to$' is contravariant on its left argument, filtered intersections are transformed into directed unions.

We write $\vdash A$ if the assertion A of $\mathbf{T}(\sigma)$ is derivable from the axioms and rules in Figs. 3–5. An assertion A of $\mathbf{T}(\sigma)$ is *valid* if $\models A$, and a rule is *sound* if it produces valid assertions from valid assertions.

3.1. Soundness

We have seen in the previous section that the logical and denotational interpretations of a domain expression σ are connected by a function

$$[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$$

which interprets types of $\mathbf{T}(\sigma)$ as saturated sets of the Scott topology of the Scott domain $\mathcal{D}(\sigma)$. In this and the next subsections we will prove that the above function is a pre-order isomorphism, from which it will follows that $\mathcal{D}(\sigma)$ is a sound and complete model for $\mathbf{T}(\sigma)$.

The following proposition is due to Abramsky [3, Proposition 4.2.2 and Theorem 4.3.2]. Basically, it is a statement of soundness and completeness of the finitary language $\mathbf{T}_0(\sigma)$.

Proposition 3.5. *For each domain expression σ, the function $[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ restricts and co-restricts to a total surjective function $[\![-]\!]_\sigma : \mathbf{T}_0(\sigma) \to \mathcal{KO}(\mathcal{D}(\sigma))$ such that, for all types ψ, ψ' in $\mathbf{T}_0(\sigma)$*
1. *$\mathsf{C}(\psi)$ is derivable if and only if $[\![\psi]\!]_\sigma$ is coprime in $\mathcal{KO}(\mathcal{D}(\sigma))$;*
2. *$\vdash \psi \leqslant \psi'$ if and only if $[\![\psi]\!]_\sigma \subseteq [\![\psi']\!]_\sigma$.*

The above implies that $[\![-]\!]_\sigma : \mathbf{T}_0(\sigma) \to \mathcal{KO}(\mathcal{D}(\sigma))$ is a pre-order isomorphism. In other words, we can identify the finitary types in $\mathbf{T}_0(\sigma)$ with the Scott compact open subsets of $\mathcal{D}(\sigma)$. This fact and the following will be used in the soundness theorem below.

Proposition 3.6. *For each domain expression σ, and infinitary type $\phi \in \mathbf{T}(\sigma)$, if $\mathsf{C}(\phi)$ holds then there exists a finitary type $\psi \in \mathbf{T}_0(\sigma)$ such that $\vdash \phi = \psi$.*

Proof. By induction on the length of the derivation of $\mathsf{C}(\phi)$. □

Thus the coprimeness judgment is essentially defined for the finitary sub-language $\mathbf{T}_0(\sigma)$, and it extends only to those infinitary types in $\mathbf{T}(\sigma)$ which are provably equivalent to finitary types in $\mathbf{T}_0(\sigma)$ for which the predicate C holds.

The coprimeness judgement can be used to derive the following normal form for finitary types.

Proposition 3.7. *For each domain expression σ, and finitary type $\phi \in \mathbf{T}_0(\sigma)$, there exists a finite index set I and finitary types $\psi_i \in \mathbf{T}_0(\sigma)$ for $i \in I$ such that $\vdash \phi = \bigvee_{i\in I} \phi_i$ and $\mathsf{C}(\phi_i)$ holds for all $i \in I$.*

Proof. See Abramsky [3, Lemma 4.2.1]. □

Now we come to our main result of this section.

Theorem 3.8 (Soundness)**.** *For any domain expression σ and infinitary types ϕ and ϕ' in $\mathbf{T}(\sigma)$, the following holds*:
1. *if $\mathsf{C}(\phi)$ is derivable then $[\![\phi]\!]_\sigma$ is coprime in $\mathcal{KO}(\mathcal{D}(\sigma))$;*
2. *if $\vdash \phi \leqslant \phi'$ then $\models_\sigma \phi \leqslant \phi'$.*

Proof. We prove both statements simultaneously by induction on the length of the derivation.

The validity and soundness of the axioms and rules for the coprime judgment in Fig. 3 follow from Proposition 3.5.1. For example, the soundness of the rule $[\mathsf{C}- \to]$ is proved in [3, Proposition 3.4.2], using the induction hypothesis that for all $i \in I$, $[\![\phi_i]\!]_\sigma$ and $[\![\phi'_i]\!]_\sigma$ are all coprime in $\mathcal{KO}(\mathcal{D}(\sigma))$.

The validity and soundness of the logical axioms and rules in Fig. 4 follow by the induction hypothesis and because the codomain of the interpretation function $[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ is a completely

distributive lattice. The latter follows because $\mathcal{Q}(\mathcal{D}(\sigma))$ is closed, by definition, under arbitrary unions and arbitrary ins-tersections, and thus it is a complete sub-lattice of $\mathcal{P}(\mathcal{P}(\mathcal{D}(\sigma)))$.

Finally, the validity and soundness of the structural axioms and rules in Fig. 5 is proved using the induction hypothesis. We only prove the soundness of the rule $[\mathsf{C}- \to - \bigvee]$ and the validity of the axioms $[\to - \bigvee]$ and $[\bigwedge - \to]$. The validity and soundness of rest of the axioms and rules can be carried out by simple set-theoretical arguments.

To prove the soundness of rule $[\mathsf{C}- \to - \bigvee]$, suppose $C(\phi)$ holds. Then, by the induction hypothesis, $[\![\phi]\!]_\sigma = \uparrow b$ with b a compact element of $\mathcal{D}(\sigma)$ (recall that the coprime of the Scott topology of a Scott domain are exactly the upper closure of the compact elements of that domain). We have

$$\begin{aligned}
\left[\!\!\left[\phi \to \bigvee_{i\in I} \phi_i\right]\!\!\right]_\sigma &= \left\{ f \mid f(\uparrow b) \subseteq \bigcup_{i\in I} [\![\phi_i]\!]_\sigma \right\} \\
&= \left\{ f \mid f(b) \in \bigcup_{i\in I} [\![\phi_i]\!]_\sigma \right\} \text{ by monotonicity} \\
&= \bigcup_{i\in I} \{ f \mid f(b) \in [\![\phi_i]\!]_\sigma \} \\
&= \bigcup_{i\in I} \{ f \mid f(\uparrow b) \subseteq [\![\phi_i]\!]_\sigma \} \\
&= [\![\bigvee_{i\in I} \phi \to \phi_i]\!]_\sigma .
\end{aligned}$$

Next we prove the validity of the axiom $[\bigwedge - \to]$. We will use the Scott-open filter theorem (Proposition 2.2) to prove that, for all Scott continuous functions $f{:}X \to Y$ between Scott domains X and Y, filtered collections of Scott compact saturated subsets $\{k_i\}_{i\in I}$ of X, and Scott open subsets o of Y, if $f(\bigcap_{i\in I} k_i) \subseteq o$ then there exists $i \in I$ such that $f(k_i) \subseteq o$. Let $[\![\phi]\!]_\sigma = q$ for $q \in \mathcal{Q}(\mathcal{D}(\sigma))$ and $[\![\phi_i]\!]_\sigma = q_i \in \mathcal{Q}(\mathcal{D}(\sigma))$ for some index set I such that $\{q_i | i \in I\}$ is a filtered collection of compact saturated subsets of $\mathcal{Q}(\mathcal{D}(\sigma))$. We use this restriction to simplify the proof, but below we hint how to remove it. By definition of saturated set, we have that there exists an index set L and opens $o_l \in \mathcal{O}(\mathcal{D}(\sigma))$, for every $l \in L$, such that $q = \bigcap_{l\in L} o_l$. Furthermore, because $\mathcal{D}(\sigma)$ taken with the Scott topology forms a coherent space, we have that every saturated set is the union of compact saturated sets. Thus, for every $i \in I$ there exists an index set J_i such that $q_i = \bigcup_{j\in J_i} k_{i,j}$, with all $k_{i,j}$ compact saturated. Let $r \to s$ denote the set $\{f{:}X \to Y \mid f(r) \subseteq s\}$ for r a subset of X and s a subset of Y. We have

$$\begin{aligned}
\left[\!\!\left[\bigwedge_{i\in I} \phi_i \to \phi\right]\!\!\right]_\sigma &= \bigcap_{i\in I} q_i \to q \\
&= \bigcap_{i\in I} \bigcup_{j\in J} k_{i,j} \to \bigcap_{l\in L} o_l \qquad [\text{by definition } q \text{ and } q_i\text{'s}] \\
&= \bigcap_{l\in L} \left(\bigcap_{i\in I} \bigcup_{j\in J} k_{i,j} \to o_l \right) \quad [\text{set-theoretical calculation}] \\
&= \bigcap_{l\in L} \left(\bigcup_{h\in \Phi(\{J_i | i\in I\})} \bigcap_{i\in I} k_{i,h(i)} \to o_l \right) \quad [\text{by complete distributivity}]
\end{aligned}$$

$$
\begin{aligned}
&= \bigcap_{l\in L}\bigcap_{h\in\Phi(\{J_i\mid i\in I\})}\left(\bigcap_{i\in I} k_{i,h(i)}\to o_l\right) \quad [\text{set-theoretical calculation}]\\
&= \bigcap_{l\in L}\bigcap_{h\in\Phi(\{J_i\mid i\in I\})}\left\{f\,\middle|\, f\left(\bigcap_{i\in I} k_{i,h(i)}\right)\subseteq o_l\right\} \quad [\text{definition } \bigcup_{i\in I} k_{i,h(i)}\to o_l]\\
&= \bigcap_{l\in L}\bigcap_{h\in\Phi(\{J_i\mid i\in I\})}\left\{f\,\middle|\, \bigcap_{i\in I} k_{i,h(i)}\subseteq f^{-1}(o_l)\right\}\\
&= \bigcap_{l\in L}\bigcap_{h\in\Phi(\{J_i\mid i\in I\})}\bigcup_{i\in I}\{f\mid k_{i,h(i)}\subseteq f^{-1}(o_l)\} \quad [\text{Scott open filter theorem 2.2}]\\
&= \bigcap_{l\in L}\bigcap_{h\in\Phi(\{J_i\mid i\in I\})}\bigcup_{i\in I}(k_{i,h(i)}\to o_l)\\
&= \bigcap_{l\in L}\bigcup_{i\in I}\bigcap_{j\in J_i}(k_{i,j}\to o_l) \quad [\text{by complete distributivity}]\\
&= \bigcap_{l\in L}\bigcup_{i\in I}\left(\bigcup_{j\in J_i} k_{i,j}\to o_l\right) \quad [\text{set-theoretical calculation}]\\
&= \bigcup_{i\in I}\bigcap_{l\in L}\left(\bigcup_{j\in J_i} k_{i,j}\to o_l\right) \quad [\text{by complete distributivity, } L \text{ does not depend on } I]\\
&= \bigcup_{i\in I}\left(\bigcup_{j\in J_i} k_{i,j}\to \bigcap_{l\in L} o_l\right) \quad [\text{set-theoretical calculation}]\\
&= \bigcup_{i\in I}(q_i\to q) \quad [\text{definition of } q \text{ and } q_i\text{'s}]\\
&= \left[\!\!\left[\bigvee_{i\in I}(\phi_i\to\phi)\right]\!\!\right]_\sigma .
\end{aligned}
$$

This part of the proof concludes by observing that, for any set I, $\bigcap_I q_i = \bigcap_{J\in Fin(I)}\bigcap_{j\in J} q_j$, and that the set $\{\bigcap_{j\in J} q_j \mid J\in Fin(I)\}$ is filtered.

Finally we prove the validity of the axiom $[\to -\bigvee]$. First of all, notice that each saturated subset $q\in\mathcal{Q}(\mathcal{D}(\sigma))$ is equals to the union (possibly infinite) of intersections (possibly infinite) of coprime elements of the Scott topology of $\mathcal{D}(\sigma)$. Indeed, $q=\bigcup\{\uparrow d \mid d\in q\}$, and each $d\in q$ is the least upper bound of a directed set of compact elements b_i's in $\mathcal{D}(\sigma)$. Hence, $\uparrow d=\bigcap\{\uparrow b \mid b\sqsubseteq d\}$.

Let $[\![\phi]\!]_\sigma = q$ for $q\in\mathcal{Q}(\mathcal{D}(\sigma))$ and $[\![\phi_i]\!]_\sigma = q_i\in\mathcal{Q}(\mathcal{D}(\sigma))$ for some index set I such that $\{q_i\mid i\in I\}$ is directed in $\mathcal{Q}(\mathcal{D}(\sigma))$. Similarly as before, this restriction simplifies the proof and can be easily removed. By above, we have that q is the union of intersections of coprime elements, say $q=\bigcup_{j\in J}\bigcap_{l\in L_j}\uparrow b_{j,l}$. We have

$$[\![\phi\to\bigvee_{i\in I}\phi_i]\!]_\sigma = q\to\bigcup_{i\in I} q_i$$

$$= \bigcup_{j\in J} \bigcap_{l\in L_j} \uparrow b_{j,l} \to \bigcup_{i\in I} q_i$$

$$= \bigcap_{j\in J} \left(\bigcap_{l\in L_j} \uparrow b_{j,l} \to \bigcup_{i\in I} q_i \right) \quad \text{[set-theoretical calculation]}$$

$$= \bigcap_{j\in J} \bigcup_{M\in Fin(L_j)} \left(\bigcap_{m\in M} \uparrow b_{j,m} \to \bigcup_{i\in I} q_i \right) \text{by } [\bigwedge - \to]$$

By $[\mathsf{C} - \to - \bigvee]$ we have that

$$\uparrow b_{j,m} \to \bigcup_{i\in I} q_i = \bigcup_{i\in I} (\uparrow b_{j,m} \to q_i)$$

for each $j \in J$, $M \in Fin(L_j)$ and $m \in M$. Since M is a finite set and the set $\{q_i \mid i \in I\}$ is directed, the following holds for each $j \in J$ and $M \in Fin(L_j)$:

$$\bigcap_{m\in M} \uparrow b_{j,m} \to \bigcup_{i\in I} q_i = \bigcup_{i\in I} \left(\bigcap_{m\in M} \uparrow b_{j,m} \to q_i \right)$$

Therefore we can continue the previous chain of equalities as follows:

$$\bigcap_{j\in J} \bigcup_{M\in Fin(L_j)} \left(\bigcap_{m\in M} \uparrow b_{j,m} \to \bigcup_{i\in I} q_i \right) = \bigcap_{j\in J} \bigcup_{M\in Fin(L_j)} \bigcup_{i\in I} \left(\bigcap_{m\in M} \uparrow b_{j,m} \to q_i \right)$$

$$= \bigcap_{j\in J} \bigcup_{i\in I} \bigcup_{M\in Fin(L_j)} \left(\bigcap_{m\in M} \uparrow b_{j,m} \to q_i \right) \quad \text{[unions distribute over unions]}$$

$$= \bigcap_{j\in J} \bigcup_{i\in I} \left(\bigcap_{l\in L_j} \uparrow b_{j,l} \to q_i \right) \text{[by } [\bigwedge - \to]\text{]}$$

$$= \bigcup_{i\in I} \bigcap_{j\in J} \left(\bigcap_{l\in L_j} \uparrow b_{j,l} \to q_i \right) \text{[complete distributivity, } I \text{ does not depends on } J\text{]}$$

$$= \bigcup_{i\in I} \left(\bigcup_{j\in J} \bigcap_{l\in L_j} \uparrow b_{j,l} \to q_i \right) \quad \text{[set-theoretical calculation]}$$

$$= \bigcup_{i\in I} (q \to q_i) \text{ [definition of saturated set } q\text{]}$$

$$= \left[\!\!\left[\bigvee_{i\in I} (\phi \to \phi_i) \right]\!\!\right]_\sigma$$

As before we conclude by observing that, for any set I, $\bigcup_I q_i = \bigcup_{J\in Fin(I)} \bigcup_{j\in J} q_j$, and that the set $\{\bigcup_{j\in J} q_j \mid J \in Fin(I)\}$ is directed. □

It follows that the function $[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ is monotone. The proof also shows that the axiom schemes $[\to - \bigvee]$ and $[\bigwedge - \to]$ can be understood as an expression of compactness and finite

approximability, respectively. Logically, they allow for rewriting an arbitrary infinitary type in $\mathbf{T}(\sigma)$ into intersections of unions of finitary types in $\mathbf{T}_0(\sigma)$. This fact will be essential in the proof of the completeness result.

Lemma 3.9 (Normal form)**.** *For each domain expression σ and infinitary type $\phi \in \mathbf{T}(\sigma)$ there exists a set of finitary types $\phi_{i,j} \in \mathbf{T}_0(\sigma)$, $i \in I$ and $j \in J$, such that $\vdash \phi = \bigwedge_{i\in I} \bigvee_{j\in J} \phi_{i,j}$.*

Proof. The proof proceeds by induction on the structure of ϕ. We only treat the case $\phi \equiv \phi^l \to \phi^r$. By the induction hypothesis we have that, for some types $\phi^l_{i,j}$'s and $\phi^r_{h,k}$'s in $\mathbf{T}_0(\sigma)$, the assertions $\phi^l = \bigwedge_{i\in I} \bigvee_{j\in J} \phi^l_{i,j}$ and $\phi^r = \bigwedge_{h\in H} \bigvee_{k\in K} \phi^r_{h,k}$ are both derivable in $\mathbf{T}$. Thus we have

$$
\begin{aligned}
\phi^l \to \phi^r &= \bigwedge_{i\in I} \bigvee_{j\in J} \phi^l_{i,j} \to \bigwedge_{h\in H} \bigvee_{k\in K} \phi^r_{h,k} \quad [\text{induction hypothesis}] \\
&= \bigvee_{m\in M} \bigwedge_{i\in I} \phi^l_{i,m} \to \bigwedge_{h\in H} \bigvee_{k\in K} \phi^r_{h,k} \ [\text{complete distributivity, for some } M] \\
&= \bigwedge_{h\in H} \left(\bigvee_{m\in M} \bigwedge_{i\in I} \phi^l_{i,m} \to \bigvee_{k\in K} \phi^r_{h,k} \right) \ [\text{by } [\to - \textstyle\bigwedge]] \\
&= \bigwedge_{h\in H} \bigwedge_{m\in M} \left(\bigwedge_{i\in I} \phi^l_{i,m} \to \bigvee_{k\in K} \phi^r_{h,k} \right) \ [\text{by } [\textstyle\bigvee - \to]] \\
&= \bigwedge_{h\in H} \bigwedge_{m\in M} \bigvee_{I'\in Fin(I)} \left(\bigwedge_{i\in I'} \phi^l_{i,m} \to \bigvee_{k\in K} \phi^r_{h,k} \right) \ [\text{by } [\textstyle\bigwedge - \to]] \\
&= \bigwedge_{h\in H} \bigwedge_{m\in M} \bigvee_{I'\in Fin(I)} \bigvee_{K'\in Fin(K)} \left(\bigwedge_{i\in I'} \phi^l_{i,m} \to \bigvee_{k\in K'} \phi^r_{h,k} \right) \ [\text{by } [\to - \textstyle\bigvee]]
\end{aligned}
$$

Since I' and K' are finite index sets, $\bigwedge_{i\in I'} \phi^l_{m,i} \to \bigvee_{k\in K'} \phi^r_{h,k}$ is a type in $\mathbf{T}_0(\sigma)$. Therefore $\phi^l \to \phi^r$ is provably equivalent in $\mathbf{T}(\sigma)$ to the intersection of unions of types in $\mathbf{T}_0(\sigma)$. □

Using the axiom [*distr*] from Fig. 4 we can distribute union types over intersection types to obtain the dual of the above normal form.

Corollary 3.10. *For each domain expression σ and infinitary type $\phi \in \mathbf{T}(\sigma)$ there exists a set of finitary types $\phi_{i,j} \in \mathbf{T}_0(\sigma)$, $i \in I$ and $j \in J_i$, such that*
1. *for every $i \in I$, the set $\{\phi_{i,j} \mid j \in J_i\}$ is filtered, and*
2. $\vdash \phi = \bigvee_{i\in I} \bigwedge_{j\in J_i} \phi_{i,j}$.

Proof. Immediate, using Lemma 3.9, the axiom [*distr*] of complete distributivity and the fact that an arbitrary intersection type $\bigwedge_{j\in J_i} \phi_{i,j}$ is equivalent to $\bigwedge_{K\in Fin(J_i)} \bigwedge_{j\in K} \phi_{i,j}$. Note that, if all $\phi_{i,j}$'s are in $\mathbf{T}_0(\sigma)$ then also $\bigwedge_{j\in K} \phi_{i,j} \in \mathbf{T}_0(\sigma)$ for all finite subsets K of J_i, and that the set $\{\bigwedge_{j\in K} \phi_{i,j} \mid K \in Fin(J_i)\}$ is filtered. □

3.2. Completeness

In order to prove completeness of the type system $\mathbf{T}(\sigma)$, we proceed as follows. We first introduce an intermediate language $\mathbf{T}_1(\sigma)$ adding a single top-level infinitary union type to $\mathbf{T}_0(\sigma)$. We then extend Abramsky's pre-isomorphism

$$[\![-]\!]_\sigma : \mathbf{T}_0(\sigma) \to \mathcal{KO}(\mathcal{D}(\sigma))$$

to a pre-isomorphism $[\![-]\!]_\sigma : \mathbf{T}_1(\sigma) \to \mathcal{O}(\mathcal{D}(\sigma))$. In turn, we extend the latter to a pre-isomorphism $[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$, from which the completeness result will follows.

Definition 3.11. For each domain expression σ, define the class $(\phi \in)\mathbf{T}_1(\sigma)$ inductively by the following inference rule:

$$\frac{\{\phi_i \in \mathbf{T}_0(\sigma)\}_{i\in I}}{\bigvee_{i\in I} \phi_i \in \mathbf{T}_1(\sigma)},$$

where I is an arbitrary (possibly empty) index set.

For example, if all ϕ_i's are in $\mathbf{T}_0(\sigma)$ then $\bigvee_i \phi_i$ is in $\mathbf{T}_1(\sigma)$, $\bigvee_i (\phi_i)_\perp$ is in $\mathbf{T}_1((\sigma)_\perp)$ but $(\bigvee_i \phi_i)_\perp$ is not in $\mathbf{T}_1((\sigma)_\perp)$. Clearly, $\mathbf{T}_0(\sigma)$ is a sub-language of $\mathbf{T}_1(\sigma)$, which is in turn a sub-language of $\mathbf{T}(\sigma)$. Furthermore, the function $[\![-]\!]_\sigma : \mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ restricts and co-restricts to a surjection $[\![-]\!]_\sigma : \mathbf{T}_1(\sigma) \to \mathcal{O}(\mathcal{D}(\sigma))$, simply because

$$\left[\!\!\left[\bigvee_{i\in I} \psi_i \right]\!\!\right]_\sigma = \bigcup_{i\in I} [\![\psi_i]\!]_\sigma$$

and $[\![\psi_i]\!]_\sigma \in \mathcal{KO}(\mathcal{D}(\sigma))$. It is surjective because each Scott open set in $\mathcal{O}(\mathcal{D}(\sigma))$ is the union of the compact sets below it. The latter follows because $\mathcal{D}(\sigma)$ is a Scott domain, and thus, when taken with the Scott topology forms a spectral space.

In the next theorem we prove completeness for assertions with at the left-hand side an infinitary type from $\mathbf{T}(\sigma)$, and at the right-hand side an infinitary type from $\mathbf{T}_1(\sigma)$.

Theorem 3.12 (Completeness for $\mathbf{T}_1(\sigma)$)**.** *For every domain expression σ infinitary type $\phi_1 \in \mathbf{T}(\sigma)$ and $\phi_2 \in \mathbf{T}_1(\sigma)$, $\models_\sigma \phi_1 \leqslant \phi_2$ if and only if $\vdash \phi_1 \leqslant \phi_2$.*

Proof. The direction from right to left is the soundness and follows from Theorem 3.8. The other direction can be proved as follows. By Corollary 3.10 we can prove $\phi_1 = \bigvee_{i\in I} \bigwedge_{j\in J_i} \psi_{i,j}$ for some $\psi_{i,j}$'s in $\mathbf{T}_0(\sigma)$ and such that, for a fixed i, they form a filtered set. Also, by definition of $\mathbf{T}_1(\sigma)$ we have $\phi_2 \equiv \bigvee_{l\in L} \psi_l$, for some set of ψ_l's in $\mathbf{T}_0(\sigma)$. Thus $[\![\psi_{i,j}]\!]_\sigma$ and $[\![\psi_l]\!]_\sigma$ are compact opens of $\mathcal{D}(\sigma)$, for all $i \in I$, $j \in J_i$ and $l \in L$. We have

$$\begin{aligned} \models_\sigma \phi_1 \leqslant \phi_2 &\iff [\![\phi_1]\!]_\sigma \subseteq \left[\!\!\left[\bigvee_{l\in L} \psi_l \right]\!\!\right]_\sigma \quad [\phi_2 \equiv \bigvee_{l\in L} \psi_l] \\ &\iff \left[\!\!\left[\bigvee_{i\in I} \bigwedge_{j\in J_i} \psi_{i,j} \right]\!\!\right]_\sigma \subseteq \left[\!\!\left[\bigvee_{l\in L} \psi_l \right]\!\!\right]_\sigma \quad [\phi_1 = \bigvee_{i\in I} \bigwedge_{j\in J_i} \psi_{i,j},\ \text{soundness}] \end{aligned}$$

$$\iff \bigcup_{i\in I}\bigcap_{j\in J_i}[\![\psi_{i,j}]\!]_\sigma \subseteq \bigcup_{l\in L}[\![\psi_l]\!]_\sigma \text{ [definition of } \models_\sigma\text{]}$$

$$\iff \forall i \in I.\ \bigcap_{j\in J_i}[\![\psi_{i,j}]\!]_\sigma \subseteq \bigcup_{l\in L}[\![\psi_l]\!]_\sigma$$

$$\iff \forall i \in I.\exists L_i \in \mathit{Fin}(L).\ \bigcap_{j\in J_i}[\![\psi_{i,j}]\!]_\sigma \subseteq \bigcup_{L_i}[\![\psi_l]\!]_\sigma \text{ [} \textstyle\bigcap_{j\in J_i}[\![\psi_{i,j}]\!]_\sigma\text{'s are compact]}$$

$$\iff \forall i \in I.\exists j \in J_i \exists L_i \in \mathit{Fin}(L).[\![\psi_{i,j}]\!]_\sigma \subseteq \bigcup_{L_i}[\![\psi_l]\!]_\sigma \quad \text{[Scott open filter theorem 2.2]}$$

$$\iff \forall i \in I.\exists j \in J_i \exists L_i \in \mathit{Fin}(L).\psi_{i,j} \leqslant \bigvee_{L_i}\psi_l \quad \text{[Proposition 3.5.2]}$$

$$\Longrightarrow \forall i \in I \exists j \in J_i.\psi_{i,j} \leqslant \bigvee_{l\in L}\psi_l \text{ [by [}\textstyle\bigvee -I\text{], } L_i \subseteq L\text{]}$$

$$\Longrightarrow \forall i \in I\ \bigwedge_{j\in J_i}\psi_{i,j} \leqslant \bigvee_{l\in L}\psi_l \text{ [by [}\textstyle\bigwedge -E\text{], } j \subseteq J_i\text{, and [}\leqslant -\text{trans]]}$$

$$\iff \bigvee_{i\in I}\bigwedge_{j\in J_i}\psi_{i,j} \leqslant \bigvee_{l\in L}\psi_l \text{ [by [}\textstyle\bigvee -I\text{]]}$$

$$\iff \phi_1 \leqslant \phi_2.$$

In the above proof we have used the fact that arbitrary intersections of compact opens are again compact. This follows from the fact that $\mathcal{D}(\sigma)$ taken with the Scott topology is a coherent space. □

Since $\mathbf{T}_1(\sigma)$ is contained in $\mathbf{T}(\sigma)$, completeness of assertions on types in $\mathbf{T}_1(\sigma)$ follows. Therefore the interpretation function $[\![-]\!]_\sigma{:}\mathbf{T}_1(\sigma) \to \mathcal{O}(\mathcal{D}(\sigma))$ is a pre-order isomorphism. In the next lemma we will extend this results to $[\![-]\!]_\sigma{:}\mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$.

Lemma 3.13. *For each domain expression σ, the function $[\![-]\!]_\sigma{:}\mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ is a pre-order isomorphism.*

Proof. Because $\mathcal{D}(\sigma)$ is a Scott domain, by Theorem 2.1 we have that $\mathcal{Q}(\mathcal{D}(\sigma))$ is the free completely distributive lattice over $\mathcal{O}(\mathcal{D}(\sigma))$, which in turn we have seen to be pre-order isomorphic to $\mathbf{T}_1(\sigma)$ via $[\![-]\!]$. Since $\mathbf{T}_1(\sigma)$ is contained in $\mathbf{T}(\sigma)$ and the latter is a completely distributive lattice (up to =) it follows, by freeness, that there must be a function $h{:}\mathcal{Q}((\mathcal{D}(\sigma)) \to \mathbf{T}(\sigma)$ such that
1. $\vdash h([\![\psi]\!]_\sigma) = \psi$ for each $\psi \in \mathbf{T}_1(\sigma)$,
2. $\vdash h(\bigcap_{i\in I} q_i) = \bigwedge_{i\in I} h(q_i)$, and
3. $\vdash h(\bigcup_{i\in I} q_i) = \bigvee_{i\in I} h(q_i)$,

where I is an arbitrary index set and all q_i's are in $\mathcal{Q}((\mathcal{D}(\sigma))$. Note that, since $q = \bigcap\{o \in \mathcal{O}((\mathcal{D}(\sigma)) \mid q \subseteq o\}$ for every saturated set $q \in \mathcal{Q}(\mathcal{D}(\sigma))$, and $[\![-]\!]_\sigma$ is a pre-order isomorphism between $\mathbf{T}_1(\sigma)$ and $\mathcal{O}(\mathcal{D}(\sigma))$, we have that

$$h(q) = \bigwedge\{h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } q \subseteq [\![\psi]\!]_\sigma\}.$$

The above intersection is well defined because there are only set many types in the image of the function h. From the above, it follows that $h{:}\mathcal{Q}((\mathcal{D}(\sigma)) \to \mathbf{T}(\sigma)$ is monotone. Recall that also the function $[\![-]\!]_\sigma{:}\mathbf{T}(\sigma) \to \mathcal{Q}((\mathcal{D}(\sigma))$ is monotone by the soundness Theorem 3.8.

Next we prove that the two functions are the inverse of each other. For every $q \in \mathcal{Q}((\mathcal{D}(\sigma))$ we have

$$\begin{aligned}
[\![h(q)]\!]_\sigma &= [\![h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } q \subseteq [\![\psi]\!]_\sigma\}_\sigma]\!] \\
&= \bigcap\{[\![\psi]\!]_\sigma \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } q \subseteq [\![\psi]\!]_\sigma\} \; [[\![-]\!]_\sigma \text{ preserves intersections, } \vdash h([\![\psi]\!]_\sigma) = \psi] \\
&= \bigcap\{o \in \mathcal{O}(\mathcal{D}(\sigma)) \mid q \subseteq o\} \; [[[\![-]\!]_\sigma{:}\mathbf{T}_1(\sigma) \to \mathcal{O}(\mathcal{D}(\sigma)) \text{is a pre-order isomorphism}] \\
&= q
\end{aligned}$$

and for every $\phi \in \mathbf{T}(\sigma)$ we have

$$\begin{aligned}
h([\![\phi]\!]_\sigma) &= \bigwedge\{h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } [\![\phi]\!]_\sigma \subseteq [\![\psi]\!]_\sigma\} \\
&= \bigwedge\{h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } \vdash \phi \leqslant \psi\}. \quad [\text{Theorem 3.12}]
\end{aligned}$$

Since $\vdash \phi \leqslant \psi$ and $\vdash \psi = h([\![\psi]\!]_\sigma)$, we have by $[\bigwedge -I]$,

$$\vdash \phi \leqslant \bigwedge\{h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } \vdash \phi \leqslant \psi\}.$$

To prove the converse we use Lemma 3.9 in order to obtain a set of type $\psi_i \in \mathbf{T}_1(\sigma)$ such that $\vdash \phi = \bigwedge_{i\in I} \psi_i$. Thus $\vdash \phi \leqslant \psi_i$ for all $i \in I$ (by $[\bigwedge -E]$ and $[\leqslant -trans]$), from which it follows by $[\bigwedge -I]$ that

$$\vdash \bigwedge\{h([\![\psi]\!]_\sigma) \mid \psi \in \mathbf{T}_1(\sigma) \text{ and } \vdash \phi \leqslant \psi\} \leqslant \bigwedge_{i\in I} \psi_i = \phi.$$

Therefore $\vdash h([\![\phi]\!]_\sigma) = \phi$, and $[\![-]\!]_\sigma{:}\mathbf{T}(\sigma) \to \mathcal{Q}(\mathcal{D}(\sigma))$ is a pre-order isomorphism. □

We can finally prove the completeness of $\mathbf{T}$.

Theorem 3.14 (Completeness)**.** *For every domain expression σ, and $\phi_1, \phi_2 \in \mathbf{T}(\sigma)$, $\models_\sigma \phi_1 \leqslant \phi_2$ if and only if $\vdash \phi_1 \leqslant \phi_2$.*

Proof. Since $[\![-]\!]_\sigma$ is a pre-order isomorphism we have

$$\models_\sigma \phi_1 \leqslant \phi_2 \iff [\![\phi_1]\!]_\sigma \subseteq [\![\phi_2]\!]_\sigma \iff \vdash \phi_1 \leqslant \phi_2. \qquad \square$$

4. A type assignment system

In this section we consider a language to talk about the elements of a domain. Basically it is an extension of the λ-calculus with terms for pairing, lifting, and least upper bound operators.

Definition 4.1. For every domain expression σ and domain environment ε, let $Var(\sigma)$ be an infinite set of variables $x^\sigma, y^\sigma, \ldots$ We define the set of well-formed terms $(M \in)\Lambda(\sigma)_\varepsilon$ inductively by the inference rules and axioms given in Fig. 6 (we omit the domain environment information).

$$x^\sigma \in \Lambda(\sigma) \qquad \frac{x \in \Lambda(\sigma) \quad M \in \Lambda(\sigma)}{\mathsf{lub}(x).M \in \Lambda(\sigma)}$$

$$\frac{M \in \Lambda(\sigma) \quad N \in \Lambda(\tau)}{\langle M, N\rangle \in \Lambda(\sigma \times \tau)} \qquad \frac{M \in \Lambda(\sigma_1 \times \sigma_2) \quad x \in \Lambda(\sigma_1) \quad y \in \Lambda(\sigma_2) \quad N \in \Lambda(\tau)}{\mathsf{let}\ M\ \mathsf{be}(\langle x, y\rangle).N \in \Lambda(\tau)}$$

$$\frac{M \in \Lambda(\sigma)}{\mathsf{inl}(M) \in \Lambda(\sigma + \tau)} \qquad \frac{M \in \Lambda(\sigma_1 + \sigma_2) \quad x \in \Lambda(\sigma_1) \quad y \in \Lambda(\sigma_2) \quad N_1, N_2 \in \Lambda(\tau)}{\mathsf{cases}\ M\ \mathsf{of}\ \mathsf{inl}(x).N_1\ \mathsf{else}\ \mathsf{inr}(y).N_2 \in \Lambda(\tau)}$$

$$\frac{M \in \Lambda(\tau)}{\mathsf{inr}(M) \in \Lambda(\sigma + \tau)}$$

$$\frac{x \in \Lambda(\sigma) \quad M \in \Lambda(\tau)}{\lambda x^\sigma.M \in \Lambda(\sigma \to \tau)} \qquad \frac{M \in \Lambda(\sigma \to \tau) \quad N \in \Lambda(\sigma)}{MN \in \Lambda(\tau)}$$

$$\frac{M \in \Lambda(\sigma)}{\mathsf{up}(M) \in \Lambda((\sigma)_\perp)} \qquad \frac{M \in \Lambda((\sigma)_\perp) \quad x \in \Lambda(\sigma) \quad N \in \Lambda(\tau)}{\mathsf{lift}\ M\ \mathsf{to}\ \mathsf{up}(x).N \in \Lambda(\tau)}$$

$$\frac{M \in \Lambda(\sigma[\mathsf{rec}e.\sigma/e])}{\mathsf{fold}(M) \in \Lambda(\mathsf{rec}e.\sigma)} \qquad \frac{M \in \Lambda(\mathsf{rec}e.\sigma)}{\mathsf{unfold}(M) \in \Lambda(\sigma[\mathsf{rec}e.\sigma/e])}$$

Fig. 6. Well-formed terms.

When no confusion arises, we will write a variable x^σ as x.

The operator lub, λ, let, cases, and lift are variable binding in the standard way. Note that each term construct is presented by an introduction and elimination rule for each domain expressions σ. Since the denotation of a recursive domain expression $\mathsf{rec}e.\sigma$ is the solution of a domain equation up-to isomorphism, we need the unfolding of term in $\mathsf{rec}e.\sigma$ to denote the corresponding term in $\sigma[\mathsf{rec}e.\sigma/e]$, and conversely we need the folding of it.

The meaning of well-formed terms in $\Lambda(\sigma)$ is given by elements of the Scott domain $\mathcal{D}(\sigma)$ associated with the domain expression σ. Intuitively, lub is the least upper bound operator, $\langle -, -\rangle$ the pairing operator, let, cases, and lift are various form of assignment to local variables, inl, and inr are the left and right injections into the sum of two types, and up is the injection into the lifting of a type.

More formally, let *TEnv* be the set of term environments ρ mapping term variables in $\mathrm{V}ar(\sigma)$ to elements of a Scott domain $\mathcal{D}(\sigma)$, for each domain expression σ. The semantics $[\![M]\!]^\sigma_\rho$ of a term $M \in \Lambda(\sigma)$ at an environment ρ is defined by induction on its structure in Fig. 7.

Note that the meaning of $\mathsf{lub}(x).M$ is well defined in any domain $\mathcal{D}(\sigma)$, where σ is such that $M \in \Lambda(\sigma)$. In fact one can prove by induction on the structure of M, that $d_k \leqslant d_k + 1$ for each $k \in \omega$, where all d_k's are defined as in Fig. 7. Thus $(d_k)_{k\in\omega}$ forms an ω-chain that has least upper bound $\bigsqcup_{k\in\omega} d_k$ in $\mathcal{D}(\sigma)$.

Next we define, for each domain expression σ, a type assignment system for terms in $\Lambda(\sigma)$ relative to the infinitary type theory $\mathbf{T}(\sigma)$. Judgments have the form $\Gamma \vdash_\varepsilon M{:}\phi$, where ε is a domain environment, the *subject* M is a term in $\Lambda(\sigma)_\varepsilon$, the *predicate* ϕ is a type in $\mathbf{T}(\sigma)_\varepsilon$, and the *assumption* Γ on the free variables of M is a set consisting of pairs $x^\tau{:}\phi_x$ for x^τ a term variable in $\mathrm{V}ar(\tau)$ and ϕ_x a type in $\mathbf{T}(\tau)_\varepsilon$ for some domain expression τ. We assume that each term variable in Γ appears at most once, and that

$$
\begin{array}{lcl}
[\![x]\!]^{\sigma}_{\rho} & = & \rho(x) \\
[\![\mathsf{lub}(x).M]\!]^{\sigma}_{\rho} & = & \bigsqcup_{k\in\omega} d_k \quad \text{where} \begin{cases} d_0 = \bot \text{ and} \\ d_{k+1} = [\![M]\!]^{\sigma}_{\rho[d_k/x]} \end{cases} \\
[\![\lambda x.M]\!]^{\sigma\to\tau}_{\rho} & = & \lambda d \in \mathcal{D}(\sigma).[\![M]\!]^{\tau}_{\rho[d/x]} \\
[\![MN]\!]^{\tau}_{\rho} & = & [\![M]\!]^{\sigma\to\tau}_{\rho}([\![N]\!]^{\sigma}_{\rho}) \\
[\![\langle M,N\rangle]\!]^{\sigma\times\tau}_{\rho} & = & \langle [\![M]\!]^{\sigma}_{\rho}, [\![N]\!]^{\tau}_{\rho}\rangle \\
[\![\mathsf{let}\ M\ \mathsf{be}(\langle x,y\rangle).N]\!]^{\sigma}_{\rho} & = & [\![N]\!]^{\tau}_{\rho[x/lpr([\![M]\!]^{\sigma_1\times\sigma_2}_{\rho}),y/rpr([\![M]\!]^{\sigma_1\times\sigma_2}_{\rho})]} \\
[\![\mathsf{inl}(M)]\!]^{\sigma+\tau}_{\rho} & = & inl([\![M]\!]^{\sigma}_{\rho}) \\
[\![\mathsf{inr}(M)]\!]^{\sigma+\tau}_{\rho} & = & inr([\![M]\!]^{\tau}_{\rho}) \\
[\![\mathsf{cases}\ M\ \mathsf{of}\ \mathsf{inl}(x).N_1\ \mathsf{else}\ \mathsf{inr}(y).N_2]\!]^{\sigma}_{\rho} & = & \begin{cases} [\![N_1]\!]^{\sigma}_{\rho[x/d]} & \text{if } [\![M]\!]^{\sigma_1+\sigma_2}_{\rho} = inl(d) \\ [\![N_2]\!]^{\sigma}_{\rho[x/d_2]} & \text{if } [\![M]\!]^{\sigma_1+\sigma_2}_{\rho} = inr(d) \end{cases} \\
[\![\mathsf{up}(M)]\!]^{(\sigma)_\bot}_{\rho} & = & up([\![M]\!]^{\sigma}_{\rho}) \\
[\![\mathsf{lift}\ M\ \mathsf{to}\ \mathsf{up}(x).N]\!]^{\sigma}_{\rho} & = & \begin{cases} [\![N]\!]^{\sigma}_{\rho[x/d]} & \text{if } [\![M]\!]^{(\tau)_\bot}_{\rho} = up(d) \\ \bot & \text{if } [\![M]\!]^{(\tau)_\bot}_{\rho} = \bot \end{cases} \\
[\![\mathsf{fold}(M)]\!]^{\mathsf{rec}e.\sigma}_{\rho} & = & fold([\![M]\!]^{\sigma[\mathsf{rec}e.\sigma/e]}_{\rho}) \\
[\![\mathsf{unfold}(M)]\!]^{\sigma[\mathsf{rec}e.\sigma/e]} & = & unfold([\![M]\!]^{\mathsf{rec}e.\sigma})
\end{array}
$$

Fig. 7. Meaning of terms.

all free variables of M are declared in Γ. Furthermore we denote by $\Delta \leqslant \Gamma$ the fact that Δ and Γ are both assumptions about the same variables and $\vdash \phi_x \leqslant_\sigma \psi'_x$ for all $x^\sigma{:}\phi_x \in \Delta$ and $x^\sigma{:}\psi_x \in \Gamma$.

For example, if u_1 and u_2 are basic types in $\mathbf{T}(\varepsilon(e))_\varepsilon$, for some domain expression variable e and interpretation ε (that is, they are compact open subsets of the Scott domain $\varepsilon(e)$), then $x{:}u_1, y{:}u_2 \vdash \lambda x.y{:}u_1 \to u_2$ is a syntctically correct judgement only if the term variable $x \in Var(e)$ is different from y.

Validity of the typing judgments is defined in terms of the meaning of terms as elements of a Scott domain defined in Fig. 7. For a domain environment $\varepsilon \in DEnv$, and domain expression σ we define the relation $\models_\varepsilon$ as follows:

$$\Gamma \models_\varepsilon M{:}\phi \iff \forall \rho \in TEnv.\, \forall x^\tau{:}\phi_x \in \Gamma.\, \rho(x) \models_{\tau,\varepsilon} \phi_x \text{ implies } [\![M]\!]^{\sigma}_{\rho} \models_{\sigma,\varepsilon} \phi,$$

where $M \in \Lambda(\sigma)_\varepsilon$ and $\phi \in \mathbf{T}(\sigma)_\varepsilon$.

In Fig. 8 we present an axiomatization of the type judgments (we omit the domain environment information). The above type judgment is standard. Basically, it is Abramsky's endogenous logic $\Lambda(\sigma)$ for a domain expression σ [3] augmented to allow for infinite meets and joins. Its presentation is rather intuitive: there is exactly one type assignment to a term for each introduction and elimination rule of a type construct. The rule $[\vdash - \leqslant]$ expresses type subsumption both to the left and the right. In the presence of union types both subsumptions will be necessary to prove the completeness of our system.

Theorem 4.2 (Soundness)**.** *Let σ be a domain expression σ, and ε a domain environment. For all assumptions Γ, terms $M \in \Lambda(\sigma)_\varepsilon$, and types $\phi \in \mathbf{T}(\sigma)_\varepsilon$,*

$$[\vdash -\mathsf{Var}]\ \frac{x{:}\phi \in \Gamma}{\Gamma \vdash x{:}\phi} \qquad [\vdash - \leq]\ \frac{\Gamma \leq \Delta \quad \Delta \vdash M{:}\phi \quad \phi \leq \psi}{\Gamma \vdash M{:}\psi}$$

$$[\vdash - \bigvee]\ \frac{\{\Gamma, x{:}\phi_i \vdash M{:}\phi\}_{i\in I}}{\Gamma, x{:}\bigvee_{i\in I}\phi_i \vdash M{:}\phi} \qquad [\vdash - \bigwedge]\ \frac{\{\Gamma \vdash M{:}\phi_i\}_{i\in I}}{\Gamma \vdash M{:}\bigwedge_{i\in I}\phi_i}$$

$$[\vdash - \times -I]\ \frac{\Gamma \vdash M{:}\phi \quad \Gamma \vdash N{:}\psi}{\Gamma \vdash \langle M, N\rangle{:}\phi \times \psi} \qquad [\vdash - \times -E]\ \frac{\Gamma \vdash M{:}\phi_1 \times \phi_2 \quad \Gamma, x{:}\phi_1, y{:}\phi_2 \vdash N{:}\psi}{\Gamma \vdash \mathsf{let}\ M\ \mathsf{be}(\langle x, y\rangle).N{:}\psi}$$

$$[\vdash - + L - I]\ \frac{\Gamma \vdash M{:}\phi}{\Gamma \vdash \mathsf{inl}(M){:}\phi + \mathsf{Bot}} \qquad [\vdash - + L - E]\ \frac{\Gamma \vdash M{:}\phi + \mathsf{Bot} \quad \Gamma, x{:}\phi \vdash N_1{:}\psi}{\Gamma \vdash \mathsf{cases}\ M\ \mathsf{of}\ \mathsf{inl}(x).N_1\ \mathsf{else}\ \mathsf{inr}(y).N_2{:}\psi}$$

$$[\vdash - + R - I]\ \frac{\Gamma \vdash M{:}\phi}{\Gamma \vdash \mathsf{inr}(M){:}\mathsf{Bot} + \psi} \qquad [\vdash - + R - E]\ \frac{\Gamma \vdash M{:}\mathsf{Bot} + \phi \quad \Gamma, y{:}\phi \vdash N_2{:}\psi}{\Gamma \vdash \mathsf{cases}\ M\ \mathsf{of}\ \mathsf{inl}(x).N_1\ \mathsf{else}\ \mathsf{inr}(y).N_2{:}\psi}$$

$$[\vdash - \to -I]\ \frac{\Gamma, x{:}\phi_1 \vdash M{:}\phi_2}{\Gamma \vdash \lambda x.M{:}\phi_1 \to \phi_2} \qquad [\vdash - \to -E]\ \frac{\Gamma \vdash M{:}\phi_1 \to \phi_2 \quad \Gamma \vdash N{:}\phi_1}{\Gamma \vdash MN{:}\phi_2}$$

$$[\vdash -(-)_\perp - I]\ \frac{\Gamma \vdash M{:}\phi}{\Gamma \vdash \mathsf{up}(M){:}(\phi)_\perp} \qquad [\vdash -(-)_\perp - E]\ \frac{\Gamma \vdash M{:}(\phi)_\perp \quad \Gamma, x{:}\phi \vdash N{:}\psi}{\Gamma \vdash \mathsf{lift}\ M\ \mathsf{to}\ \mathsf{up}(x).N{:}\psi}$$

$$[\vdash -\mathsf{fold}]\ \frac{\Gamma \vdash M{:}\phi}{\Gamma \vdash \mathsf{fold}(M){:}\phi} \qquad [\vdash -\mathsf{unfold}]\ \frac{\Gamma \vdash M{:}\phi}{\Gamma \vdash \mathsf{unfold}(M){:}\phi}$$

$$[\vdash -\mathsf{lub} - I]\ \frac{\Gamma \vdash \mathsf{lub}(x).M{:}\phi \quad \Gamma, x{:}\phi \vdash M{:}\psi}{\Gamma \vdash \mathsf{lub}(x).M{:}\psi}$$

Fig. 8. The type assignment system.

$\Gamma \vdash M{:}\phi$ *implies* $\Gamma \models M{:}\phi$.

Proof. Soundness is proved by induction on the length of the proofs. We treat only two cases, all others are either routine or worked out in [3, Theorem 4.3.2]. Suppose the last step in the proof is an application of $[\vdash - \bigvee]$. By the induction hypothesis, $\Gamma, x{:}\phi_i \models M{:}\phi$ for all $i \in I$. For ρ a term environment in *TEnv*, let us denote by $\rho \models \Gamma$ that $\rho(x) \models \phi$ for all $x{:}\phi$ in Γ. We have,

$$\begin{aligned}
&(\forall i \in I.\Gamma, x{:}\phi_i \models M{:}\phi) \implies (\forall i \in I.\forall \rho \in TEnv.(\rho \models \Gamma, x{:}\phi_i \implies [\![M]\!]_\rho \models \phi))\\
&\quad\implies \forall \rho \in TEnv.\forall i \in I.(\rho \models \Gamma, x{:}\phi_i \implies [\![M]\!]_\rho \models \phi) \quad [\text{predicate logic}]\\
&\quad\implies \forall \rho \in TEnv.(\exists i \in I.\rho \models \Gamma, x{:}\phi_i \implies [\![M]\!]_\rho \models \phi) \quad [\text{predicate logic}]\\
&\quad\implies \forall \rho \in TEnv.\left(\rho \models \Gamma, x{:}\bigvee_{i\in I}\phi_i \implies [\![M]\!]_\rho \models \phi\right) \ [\text{definition of } \models]\\
&\quad\implies \Gamma, x{:}\bigvee_{i\in I}\phi_i \models M{:}\phi.
\end{aligned}$$

Next, we consider $[\vdash - \bigwedge]$. By the induction hypothesis, $\Gamma \models M{:}\phi_i$ for all $i \in I$. We have

$$
\begin{aligned}
&(\forall i \in I.\Gamma \models M{:}\phi_i) \implies (\forall i \in I.\forall \rho \in TEnv.(\rho \models \Gamma \Longrightarrow [\![M]\!]_\rho \models \phi_i)) \\
&\quad \Longrightarrow (\forall \rho \in TEnv.(\rho \models \Gamma \Longrightarrow \forall i \in I.[\![M]\!]_\rho \models \phi_i)) \quad \text{[predicate logic]} \\
&\quad \Longrightarrow \forall \rho \in TEnv.\left(\rho \models \Gamma \Longrightarrow [\![M]\!]_\rho \models \bigwedge_{i\in I} \phi_i\right) \text{ [definition of } \models] \\
&\quad \Longrightarrow \left(\Gamma \models M{:}\bigwedge_{i\in I} \phi_i\right) \qquad \square
\end{aligned}
$$

To get completeness we proceed as before. First we consider the finitary restriction of the above type judgment and apply Abramsky's completeness theorem for the endogenous logic $\Lambda(\sigma)$ [3, Theorem 4.3.4].

Proposition 4.3. *Let σ be a domain expression σ, and ε a domain environment. For all terms $M \in \Lambda(\sigma)_\varepsilon$, finitary types $\phi \in \mathbf{T}_0(\sigma)_\varepsilon$, and assumptions Γ such that $\phi_x \in \mathbf{T}_0(\sigma_x)_\varepsilon$ for every $x{:}\phi_x \in \Gamma$ and $x \in \mathrm{V}ar(\sigma_x)$,*

$$\Gamma \models M{:}\phi \quad \textit{implies } \Gamma \vdash M{:}\phi.$$

We extend Abramsky's result in order to accommodate infinite union and infinite intersection types. We use the above proposition, Lemma 3.9, the rules $[\vdash - \bigwedge]$ and $[\vdash - \bigvee]$ and the rule $[\vdash - \leqslant]$ together with the rules for the elimination of union and intersection types.

Theorem 4.4 (Completeness)**.** *Let σ be a domain expression σ, and ε a domain environment. For all terms $M \in \Lambda(\sigma)_\varepsilon$, assumptions Γ, and types $\phi \in \mathbf{T}(\sigma)_\varepsilon$,*

$$\Gamma \models M{:}\phi \quad \textit{if and only if } \Gamma \vdash M{:}\phi.$$

Proof. Theorem 4.2. The proof of the other implication is organized in two stages. First we consider the case when all types in the assumption Γ are general but the predicate ϕ assigned to M is finitary. We will use induction on the number of variables with non-finitary type in the assumption Γ to prove the result. The base case of the induction follows immediately from Proposition 4.3, since all types in Γ are finitary as well as the predicate ϕ.

In order to prove the inductive step, we will use the following fact (recall that $\phi \in \mathbf{T}_0(\sigma)$)

$$\Gamma, x{:}\bigwedge_{i\in I} \phi_i \models M{:}\phi \implies \exists i \in I.\Gamma, x{:}\phi_i \models M{:}\phi \tag{1}$$

for $\phi_i \in \mathbf{T}_0(\sigma_x)$ for all $i \in I$ and $x \in \mathrm{V}ar(\sigma_x)$, and such that the collection of types ϕ_i's is filtered. To prove this fact it suffices to note that $[\![M]\!]{:}TEnv \to \mathcal{D}(\sigma)$ is a Scott continuous function (this can be proved by induction on the structure of M), and hence its inverse preserves Scott open sets. Hence $[\![M]\!]^{-1}([\![\phi]\!])$ is a Scott open subset of $TEnv$. From $\Gamma, x{:}\bigwedge_{i\in I} \phi_i \models M{:}\phi$ it follows that

$$\left[\!\!\left[\bigwedge_{i\in I} \phi_i \right]\!\!\right] \subseteq \downarrow_x [\![M]\!]^{-1}([\![\phi]\!]),$$

where the right-hand side is the projection on x of $[\![M]\!]^{-1}([\![\phi]\!])$. But $[\![\bigwedge_{i\in I}\phi_i]\!] = \bigcap_{i\in I}[\![\phi_i]\!]$, and the latter is a filtered intersection of compact opens (by the assumption), and hence saturated. Finally, the implication in (1) follows from an application of the Scott-open filter theorem (Proposition 2.2).

We can now prove the inductive step. If ψ is a non-finitary type then it can be proven equivalent to the union of filtered intersections of finitary types by Lemma 3.9, complete distributivity and Theorem 3.14. We have

$$\begin{aligned}
\Gamma, x{:}\psi \models M{:}\phi \quad &\Longleftrightarrow \quad \Gamma, x{:}\bigvee_{i\in I}\bigwedge_{j\in J_i}\psi_{i,j} \models M{:}\phi \\
&\Longleftrightarrow \quad \forall i \in I.\Gamma, x{:}\bigwedge_{j\in J_i}\psi_{i,j} \models M{:}\phi \ [\text{definition of } \models, \text{ predicate logic}] \\
&\Longrightarrow \quad \forall i \in I.\exists j \in J_i \Gamma, x{:}\psi_{i,j} \models M{:}\phi \quad [\text{implication (1)}] \\
&\Longrightarrow \quad \forall i \in I.\exists j \in J_i \Gamma, x{:}\psi_{i,j} \vdash M{:}\phi \ [\psi_{i,j} \in \mathbf{T}_0(\sigma_x), \text{induction hypothesis}] \\
&\Longrightarrow \quad \forall i \in I.\Gamma, x{:}\bigwedge_{j\in J_i}\psi_{i,j} \vdash M{:}\phi \ [[\wedge -E], [\vdash - \leqslant]] \\
&\Longrightarrow \quad \Gamma, x{:}\bigvee_{i\in I}\bigwedge_{j\in J_i}\psi_{i,j} \vdash M{:}\phi \ [[\vdash - \vee]] \\
&\Longrightarrow \quad \Gamma, x{:}\psi \vdash M{:}\phi \ [\vdash \bigvee_{i\in I}\bigwedge_{j\in J_i}\psi_{i,j} = \psi, [\vdash - \leqslant]]
\end{aligned}$$

From the above we can conclude that the completeness results holds for all sequent such that the predicate ϕ assigned to the term M is finitary. We conclude the proof by allowing ϕ to be a general type in $\mathbf{T}(\sigma)$. We have

$$\begin{aligned}
\Gamma \models M{:}\phi \quad &\Longleftrightarrow \quad \Gamma \models M{:}\bigwedge_{i\in I}\bigvee_{j\in J}\phi_{i,j} \quad [\text{Lemma 3.9, Theorem 3.14}] \\
&\Longleftrightarrow \quad \forall i \in I.\exists j \in J.\Gamma \models M{:}\phi_{i,j} \ [\text{definition of } \models, \text{ predicate logic}] \\
&\Longleftrightarrow \quad \forall i \in I.\exists j \in J.\Gamma \vdash M{:}\phi_{i,j} \ [\text{previous part of the proof, } \phi_{i,j} \text{finitary}] \\
&\Longrightarrow \quad \forall i \in I.\Gamma \vdash M{:}\bigvee_{j\in J}\phi_{i,j} \ [[\vee -E], [\vdash - \leqslant]] \\
&\Longrightarrow \quad \Gamma \vdash M{:}\bigwedge_{i\in I}\bigvee_{j\in J}\phi_{i,j} \ [[\vdash - \wedge]] \\
&\Longrightarrow \quad \Gamma \vdash M{:}\phi \ [\vdash \bigwedge_{i\in I}\bigvee_{j\in J}\phi_{i,j} = \phi, [\vdash - \leqslant]] \qquad \square
\end{aligned}$$

5. An example: applicative transition systems

In this section we apply the framework developed in the previous sections to give an infinite intersection type theory for applicative transition systems, an operational model for the lazy lambda calculus. To begin with, we recall some basic definitions and facts.

Definition 5.1. The set Λ of λ-terms is defined by the following grammar:

$$M ::= x \mid \lambda x.M \mid MM,$$

where x is a variable from a fixed set $\mathrm{V}ar$.

The definitions of free variables, substitution and closed terms are standard [7]. As usual, the set of closed λ-terms is denoted by Λ^o.

There are several ways to give meaning to λ-terms. In this section we consider an operational model, based on a notion of transition relation [1].

Definition 5.2. A *quasi-applicative transition system* is a pair (A, ev) with $ev{:}A \rightharpoonup (A \to A)$. For $a \in A$, $a \Downarrow f$ denotes that $ev(a)$ is defined and $ev(a) = f$.

Quasi-applicative transition systems are an operational model for lazy functional languages. Informally, $a \Downarrow f$ means that given a term $a \in A$, the environment sees if it converges to some function f. If this is the case, then it can continue by providing another term $b \in A$ and applying f to it . As ordinary transition systems in concurrency theory, quasi-applicative transition systems can be used to identify processes with the same observable behavior. A natural observational equivalence for quasi-applicative transition system is the applicative bisimulation.

Definition 5.3. Given a quasi-applicative transition system (A, ev), a relation $R \subseteq A \times A$ is called a *applicative bisimulation* whenever, if $\langle a, b\rangle \in R$ then

$$a \Downarrow f \implies b \Downarrow g \text{ and } \forall c \in A.\ \langle f(c), g(c)\rangle \in R.$$

We write $a \lesssim^B b$ if there exists an applicative bisimulation R with $\langle a, b\rangle \in R$.

Applicative bisimulation can also be described in terms of its countable approximants [2], that is $\lesssim^B = \bigcap_\omega \lesssim^B_n$, where

- $\lesssim^B_0 = A \times A$, and
- $a \lesssim^B_{n+1} b$ if and only if $a \Downarrow f$ implies $b \Downarrow g$ and $f(c) \lesssim^B_n g(c)$ for all $c \in A$.

Definition 5.4. An *applicative transition system* is a quasi-applicative transition system (A, ev) such that for all $a, b, c \in A$

$$a \Downarrow f \text{ and } b \lesssim^B c \implies f(b) \lesssim^B f(c).$$

Next we consider two canonical examples of quasi-applicative transition systems. The first one (Λ^o, ev) is given by considering all closed λ-terms as elements of the system, and defining the evaluation function ev inductively as follows:

$$ev(\lambda x.M) = N \mapsto M[N/x], \quad ev(MN) = \begin{cases} P \mapsto g(P) & \text{if } M \Downarrow f \text{ and } f(N) \Downarrow g \\ undefined & \text{otherwise} \end{cases}$$

Informally, terms under abstraction do not evaluate, and the evaluation of an application MN proceed by first evaluating M, and if it converges to a function f then the evaluation proceeds by evaluating $f(N)$.

A second example of a quasi-applicative transition system is given by the Scott domain $\mathcal{D} = \mathcal{D}(\mathsf{rec}e.(e \to e)_\perp)_\varepsilon$, where ε is any domain environment in $DEnv$. Let $unfold$ be the isomorphism mapping $\mathcal{D}$ to $(\mathcal{D} \to \mathcal{D})_\perp$. The Scott domain $\mathcal{D}$ can be seen as a quasi-applicative transition system $(\mathcal{D}, ev)$, where $ev{:}\mathcal{D} \rightharpoonup (\mathcal{D} \to \mathcal{D})$ is defined as follows:

$$ev(d) = \begin{cases} f & \text{if } unfold(d) = up(f) \\ \text{undefined} & \text{if } unfold(d) = \perp \end{cases}$$

The Scott domain $\mathcal{D}$, is an applicative transition system because it is internally fully abstract, meaning that the order of $\mathcal{D}$ coincides with the applicative bisimulation preorder of $(\mathcal{D}, ev)$ [2, Theorem 4.1].

In the next subsection we will develop an infinitary intersection type theory for quasi-applicative transition systems, with the domain $\mathcal{D}$ as canonical model. Furthermore, we can interpret λ-terms in $\mathcal{D}$, so that we will be able to define an assignment system for λ-terms. Let $\mathrm{V}ar$ be a fixed set of term variables and $\gamma{:}\mathrm{V}ar \to \mathcal{D}$ an environment function. We define an interpretation $[\![-]\!]^{\mathcal{D}}_\gamma$ of λ-terms into $\mathcal{D}$, by

$$[\![-]\!]^{\mathcal{D}}_\gamma = [\![-]\!]^{\sigma}_\gamma \circ (-)^*,$$

where $[\![-]\!]^{\sigma}_\gamma{:}\Lambda(\sigma) \to \mathcal{D}(\sigma)$, for the domain expression $\sigma = \mathsf{rec}e.(e \to e)_\perp$, is the interpretation of terms given in the previous section, and $(-)^*{:}\Lambda \to \Lambda(\sigma)$ is a translation function defined inductively as follows:

$$\begin{aligned} (x)^* &= x^\sigma, \\ (MN)^* &= (\mathsf{lift}\,\mathsf{unfold}((M)^*)\,\mathsf{to}\,\mathsf{up}(x).x^{\sigma\to\sigma})(N)^*, \\ (\lambda x.M)^* &= \mathsf{fold}(\mathsf{up}(\lambda x^\sigma.(M)^*)). \end{aligned}$$

Note that the lift term above is a term in $\Lambda(\sigma \to \sigma)$. Since $(N)^*$ is a term in $\Lambda(\sigma)$ we can use the standard application function to obtain the meaning of $(MN)^*$. Similarly, the up term above is a term in $\Lambda((\sigma \to \sigma)_\perp)$, and thus its folding $(\lambda x.M)^*$ is a term in $\Lambda(\sigma)$.

Next we present an infinitary intersection type system for applicative transition systems. Types are syntactical entities built by means of the lazy function type constructor '$(- \to -)_\perp$', and infinitary intersection type constructor '$\bigwedge$'.

Definition 5.5. The class $(\phi \in)\mathbf{T}$ is defined inductively by the following grammar:

$$\phi \;::=\; \bigwedge_{i\in I} \phi_i \mid (\phi \to \phi)_\perp,$$

where I is an arbitrary index set. If $I = \emptyset$ then we write Top for $\bigwedge_{i\in I} \phi_i$.

For a quasi-applicative transition system $\mathcal{A} = (A, ev)$ we define the *satisfaction relation* $\models^{\mathcal{A}} \subseteq A \times \mathbf{T}$ by

$$a \models^{\mathcal{A}} \bigwedge_{i\in I} \phi_i \iff \forall i \in I.\, a \models^{\mathcal{A}} \phi_i,$$

$$a \models^{\mathcal{A}} (\phi_1 \to \phi_2)_\perp \iff a \Downarrow f \text{ and } \forall b \in A.\, b \models^{\mathcal{A}} \phi_1 \Longrightarrow f(b) \models^{\mathcal{A}} \phi_2.$$

As usual we define $\mathcal{A} \models \phi \leqslant \psi$ by $a \models^{\mathcal{A}} \phi$ implies $a \models^{\mathcal{A}} \psi$ for all $a \in A$. Furthermore, for any class C of quasi-applicative transition systems we say $\mathsf{C} \models \phi \leqslant \psi$ if and only if $\mathcal{A} \models \phi \leqslant \psi$ for all quasi-applicative transition systems $\mathcal{A}$ in C.

Note that types in $\mathbf{T}$ are also types in $\mathbf{T}(\mathsf{rec}e.(e \to e)_\perp)$ (where we omit the domain environment information because the expression is closed). This inclusion holds also for their interpretations.

Lemma 5.6. *Let σ be the domain expression $\mathsf{rec}e.(e \to e)_\perp$, and $\mathcal{D} = \mathcal{D}(\sigma)$. For any type $\phi \in T$ and $d \in \mathcal{D}$ we have $d \models^{\mathcal{D}} \phi$ if and only if $d \models_\sigma \phi$.*

Proof. The proof proceeds by induction on the structure of the type. We treat only one case for illustration.

$$\begin{aligned}
& d \models^{\mathcal{D}} (\phi_1 \to \phi_2)_\perp \\
& \quad \iff \mathit{unfold}(d) = \mathit{up}(f) \text{ and } \forall d' \in \mathcal{D}.\, d' \models^{\mathcal{D}} \phi_1 \Longrightarrow f(d') \models^{\mathcal{D}} \phi_2 \ [\text{definition of } (\mathcal{D}, ev)] \\
& \quad \iff \mathit{unfold}(d) = \mathit{up}(f) \text{ and } \forall d' \in \mathcal{D}.\, d' \models_\sigma \phi_1 \Longrightarrow f(d') \models_\sigma \phi_2 \quad [\text{induction hypothesis}] \\
& \quad \iff \mathit{unfold}(d) = \mathit{up}(f) \text{ and } f \models_{\sigma\to\sigma} \phi_1 \to \phi_2 \\
& \quad \iff \mathit{unfold}(d) \models_{(\sigma\to\sigma)_\perp} (\phi_1 \to \phi_2)_\perp \\
& \quad \iff d \models_\sigma (\phi_1 \to \phi_2)_\perp. \qquad \square
\end{aligned}$$

Next we characterize the notion of applicative bisimulation in terms of the formulas satisfied by the elements of a quasi-applicative transition system. Below we denote, for a quasi-applicative transition system (A, ev), and $a \in A$, by $\mathbf{T}(a)$ the class of types $\phi \in \mathbf{T}$ such that $a \models \phi$.

Lemma 5.7. *Let $\mathcal{A} = (A, ev)$ be a quasi-applicative transition system (A, ev), and $a, b \in A$. If $a \lesssim^B b$ then $\mathbf{T}(a) \subseteq \mathbf{T}(b)$.*

Proof. Assume $a \lesssim^B b$. We prove that $a \models^{\mathcal{A}} \phi$ implies $b \models^{\mathcal{A}} \phi$ for all $\phi \in \mathbf{T}$ by induction on the structure of ϕ. The only non trivial case is $\phi = (\phi_1 \to \phi_2)_\perp$. We have

$$\begin{aligned}
& a \models^{\mathcal{A}} (\phi_1 \to \phi_2)_\perp \iff a \Downarrow f \text{ and } \forall c \in A\colon c \models \phi_1 \Longrightarrow f(c) \models \phi_2 \ [\text{definition } \models^{\mathcal{A}}] \\
& \quad \Longrightarrow b \Downarrow g \text{ and } \forall c \in A\colon f(c) \lesssim^B g(c) \text{ and } \forall c \in A\colon c \models \phi_1 \Longrightarrow f(c) \models \phi_2 \ [a \lesssim^B b] \\
& \quad \Longrightarrow b \Downarrow g \text{ and } \forall c \in A\colon c \models \phi_1 \Longrightarrow g(c) \models \phi_2 \quad [\text{induction hypothesis}] \\
& \quad \Longrightarrow b \models (\phi_1 \to \phi_2)_\perp. \qquad \square
\end{aligned}$$

To prove the converse of this lemma we need to restrict the class of quasi-applicative transition systems. Let $\mathbf{T}_0$ be the sub-language of $\mathbf{T}$ obtained by the restriction to finite intersection types only.

Definition 5.8. A quasi-applicative transition system $\mathcal{A} = (A, ev)$ is said to be *finitary* if for all $a, b, c \in A$

$$a \Downarrow f \text{ and } \mathbf{T}_0(b) \subseteq \mathbf{T}_0(c) \Longrightarrow \mathbf{T}_0(f(b)) \subseteq \mathbf{T}_0(f(c)).$$

In other words, in a finitary quasi-applicative transition system, the functions resulting upon evaluation of elements of A preserves the order induced by the finite types of its argument. For example, the Scott domain $\mathcal{D}$ is finitary when seen as a quasi-applicative transition system. Assume $d \Downarrow f$ and let $d_1, d_2 \in \mathcal{D}$. We have

$$\begin{aligned}
&\mathbf{T}_0(d_1) \subseteq \mathbf{T}_0(d_2) \implies d_1 \leqslant d_2 \quad [\text{[2, Theorem 5.12]}]\\
&\implies f(d_1) \leqslant f(d_2)\ [f \text{ is monotone}]\\
&\implies f(d_1) \precsim^B f(d_2) \quad [\text{[2, Theorem 4.1]}]\\
&\implies \mathbf{T}_0(f(d_1)) \subseteq \mathbf{T}_0(f(d_2)) \quad [\text{Lemma 5.7}]
\end{aligned}$$

We have the following characterization theorem.

Theorem 5.9. *Let (A, ev) be a finitary quasi-applicative transition system. For all $a, b \in A$,*

$$a \precsim^B b \iff \mathbf{T}_0(a) \subseteq \mathbf{T}_0(b).$$

Proof. The implication from left to right follows from Lemma 5.7. To prove the converse we define the *functional depth* of a type in $\mathbf{T}_0$ by

$$\begin{aligned}
&\mathit{depth}\left(\bigwedge_I \phi_i\right) = \max\{\mathit{depth}(\phi_i) \mid i \in I\}\\
&\mathit{depth}((\phi_1 \to \phi_2)_\perp) = \mathit{depth}(\phi_2) + 1,
\end{aligned}$$

where $\max \emptyset = 0$. Further, given a quasi-applicative transition system (A, ev), $a \in A$, and a natural number n we define $\mathbf{T}_0(a, n)$ to be the set

$$\{\phi \in \mathbf{T}_0 \mid a \models^{\mathcal{A}} \phi \text{ and } \mathit{depth}(\phi) \leqslant n\}.$$

Next we show that for a finitary quasi-applicative transition system $\mathcal{A} = (A, ev)$,

$$\mathbf{T}_0(a, n) \subseteq \mathbf{T}_0(b, n) \implies a \precsim_n^B b$$

for all $a, b \in A$ with and natural number n. We proceed by induction on n. The case $n = 0$ is trivial. Assume now $a \not\precsim_{n+1}^B b$; we have to find $\phi \in \mathbf{T}_0(a, n)$ such that $b \not\models^{\mathcal{A}} \phi$. We distinguish two cases:

1. $a \Downarrow f$ and there is no $g{:}A \to A$ such that $b \Downarrow g$. Take $\phi = (\mathsf{Top} \to \mathsf{Top})_\perp$. Then $\mathit{depth}(\phi) = 1 \leqslant n + 1$, and $a \models^{\mathcal{A}} \phi$ but $b \not\models^{\mathcal{A}} \phi$.
2. $a \Downarrow f$, $b \Downarrow g$ but there exists $c \in A$ such that $f(c) \not\precsim_n^B g(c)$. By induction hypothesis there is $\phi_r \in \mathbf{T}_0(f(c), n)$ such that $g(c) \not\models^{\mathcal{A}} \phi_r$. Take now $\phi = (\phi_l \to \phi_r)_\perp$, where $\phi_l = \bigwedge \mathbf{T}_0(c, n)$. We have $\mathit{depth}(\phi) = \mathit{depth}(\phi_r) + 1 \leqslant n + 1$, and $b \not\models^{\mathcal{A}} \phi$ because $c \models^{\mathcal{A}} \phi_l$ but $g(c) \not\models^{\mathcal{A}} \phi_r$. To prove that $a \models^{\mathcal{A}} \phi$ it is sufficient to prove that $f(d) \models^{\mathcal{A}} \phi_r$ for all $d \in A$ such that $d \models^{\mathcal{A}} \phi_l$. Let $d \in A$ be such that $d \models^{\mathcal{A}} \phi_l$. Then $\mathbf{T}_0(c, n) \subseteq \mathbf{T}_0(d, n)$. Because (A, ev) is a finitary quasi-applicative transition system and $a \Downarrow f$, it follows that $\mathbf{T}_0(f(c), n) \subseteq \mathbf{T}_0(f(d), n)$. Since $\phi_r \in \mathbf{T}_0(f(c), n)$, we obtain that also $f(d) \models^{\mathcal{A}} \phi_r$. Therefore $a \models^{\mathcal{A}} \phi$. □

As a corollary of this theorem we obtain that $a \precsim^B b$ if and only if $\mathbf{T}(a) \subseteq \mathbf{T}(b)$ for all a, b elements of a finitary quasi-applicative transition system. Also, we have the following.

Corollary 5.10. *A finitary quasi-applicative transition system is an applicative transition system.*

Proof. Let $\mathcal{A} = (A, ev)$ be a finitary quasi-applicative transition system, and $a \in A$ such that $a \Downarrow f$. For all $b, c \in A$ we have $b, c \in A$

$$\begin{aligned} & b \lesssim^B c \implies \mathbf{T}_0(b) \subseteq \mathbf{T}_0(c) \quad [\text{Lemma 5.7}] \\ \implies & \mathbf{T}_0(f(b)) \subseteq \mathbf{T}_0(f(c)) \ [\mathcal{A} \text{ is finitary}] \\ \implies & f(b) \lesssim^B f(c) \ [\text{Theorem 5.9, } \mathcal{A} \text{ finitary.}] \end{aligned}$$ □

Note that Theorem 5.9 can be used to express behavioral information through an infinitary intersection type. Indeed, for a finitary applicative transition system $\mathcal{A} = (A, ev)$ and $a, b \in A$, we have that $a \models \bigwedge \mathbf{T}_0(b)$ if and only if $a \lesssim^B b$ ($\mathbf{T}_0(b)$ is a proper-set, and hence $\bigwedge \mathbf{T}_0(b)$ is a well-defined type in $\mathbf{T}$).

In Fig. 9 we introduce a system for subtyping judgments in $\mathbf{T}$.

Theorem 5.11 (Soundness)**.** *Let ϕ_1 and ϕ_2 be two types in $\mathbf{T}$. If $\vdash \phi_1 \leqslant \phi_2$ then $\mathcal{A} \models \phi_1 \leqslant \phi_2$ for any quasi-applicative transition system $\mathcal{A} = (A, ev)$.*

Proof. By induction on the length of the derivation. □

Before proving the completeness of type system $\mathbf{T}$ we need the following lemma.

Lemma 5.12. *Let σ be the domain expression* $\mathsf{rec}e.(e \to e)_\perp$. *For types $\phi, \psi \in \mathbf{T}$, if $\vdash \phi \leqslant \psi$ in $\mathbf{T}(\sigma)$ then $\vdash \phi \leqslant \psi$ also in $\mathbf{T}$.*

Proof. The proof is analogous to the one of Proposition 5.11 in [2]. Using the last two axioms in Fig. 9, each type $\phi \in \mathbf{T}$ can be proved equivalent to one in normal form with no intersection type at the right-hand side of the $(- \to -)_\perp$ constructor, with the exception of the empty intersection type Top.

Note that in $\mathbf{T}(\sigma)$, for $\sigma = \mathsf{rec}e.(e \to e)_\perp$, we have

- $\phi \leq \phi$
- $\dfrac{\phi_1 \leq \phi_2 \text{ and } \phi_2 \leq \phi_3}{\phi_1 \leq \phi_3}$
- $\dfrac{\{\phi \leq \phi_i\}_{i \in I}}{\phi \leq \bigwedge_{i \in I} \phi_i}$
- $\dfrac{k \in I}{\bigwedge_{i \in I} \phi_i \leq \phi_k}$
- $\dfrac{\phi_2 \leq \phi_1 \text{ and } \phi_3 \leq \phi_4}{(\phi_1 \to \phi_3)_\perp \leq (\phi_2 \to \phi_4)_\perp}$
- $\dfrac{I \neq \emptyset}{\bigwedge_{i \in I} (\phi \to \phi_i)_\perp \leq (\phi \to \bigwedge_{i \in I} \phi_i)_\perp}$
- $(\phi \to \mathsf{Top})_\perp \leq (\mathsf{Top} \to \mathsf{Top})_\perp$

Fig. 9. Subtyping judgements in $\mathbf{T}$.

$$\mathbf{T}(\sigma) \vdash \phi \leqslant \psi \iff \forall j \in J.\mathbf{T}(\sigma) \vdash \bigwedge\{\phi'_i \mid \mathbf{T}(\sigma) \vdash \psi_j \leqslant \phi_i\} \leqslant \psi'_j,$$

where ϕ is of the form $\bigwedge_{i\in I}(\phi_i \to \phi'_i)_\perp$, and ψ is of the form $\bigwedge_{j\in J}(\psi_j \to \psi'_j)_\perp$. Using this fact, and the normal form above, we can prove by induction on the complexity of ϕ and ψ in $\mathbf{T}$ that if $\mathbf{T}(\sigma) \vdash \phi \leqslant \psi$ then $\mathbf{T} \vdash \phi \leqslant \psi$. □

Finally we have a completeness theorem.

Theorem 5.13. *For any class* C *of quasi-applicative transition system including* $\mathcal{D}$*, and types* ϕ, ψ *in* $\mathbf{T}$*,* $\mathsf{C} \models \phi \leqslant \psi$ *if and only if* $\mathbf{T} \vdash \phi \leqslant \psi$*.*

Proof. The direction from right to left is proved in the soundness Theorem 5.11. For the other direction we have the following:

$$\begin{aligned}
&\mathsf{C} \models \phi \leqslant \psi \\
&\quad\Longrightarrow\ \mathcal{D} \models \phi \leqslant \psi \\
&\quad\Longleftrightarrow\ \forall d \in \mathcal{D}.\, d \models^{\mathcal{D}} \phi \implies d \models^{\mathcal{D}} \psi \\
&\quad\Longleftrightarrow\ \forall d \in \mathcal{D}.\, d \models_\sigma \phi \implies d \models_\sigma \psi \quad [\text{Lemma 5.6}] \\
&\quad\Longleftrightarrow\ \mathbf{T}(\sigma) \vdash \phi \leqslant \psi \quad [\text{Theorem 3.14}] \\
&\quad\Longrightarrow\ \mathbf{T} \vdash \phi \leqslant \psi \quad [\text{Lemma 5.12}] \qquad \square
\end{aligned}$$

We can define now a type assignment system relative to the infinitary type theory $\mathbf{T}$ for the interpretation $[\![-]\!]^{\mathcal{D}}$ of λ-terms Λ as elements of $\mathcal{D}$. As in the previous section, judgments have the form $\Gamma \vdash M{:}\psi$, where $M \in \Lambda$, $\psi \in \mathbf{T}$, and Γ is a set consisting of pairs $x{:}\phi$ for x a variable in $\mathrm{V}ar$ and ϕ a type in $\mathbf{T}$. Each variable in Γ appears at most once, and that all free variables of M are declared in Γ.

Validity of the typing judgments is defined in terms as follows:

$$\Gamma \models M{:}\psi \iff \forall \gamma{:}\mathrm{V}ar \to \mathcal{D}.\, \forall x{:}\phi \in \Gamma.\, \gamma(x) \models \phi \text{ implies } [\![M]\!]^{\mathcal{D}}_\gamma \models \psi.$$

The assignment system in Fig. 10 gives an axiomatization of the type judgments. The above type is derived from the assignment system in Fig. 8 for the domain expression $\mathsf{rec}e.(e \to e)_\perp$, with the restriction to infinite intersection type only. Its presentation is rather intuitive, and differs from the standard intersection type assignment system (e.g. [8]) because abstractions receive a type that is build by means of the type constructor '$(- \to -)_\perp$'. This means that $\mathsf{Top} \leqslant (\mathsf{Top} \to \mathsf{Top})_\perp$ is not a theorem, whereas $(\phi \to \mathsf{Top})_\perp \leqslant (\mathsf{Top} \to \mathsf{Top})_\perp$ is. Note that also $\Gamma \vdash M{:}\mathsf{Top}$ is a theorem in our system because Top is the empty intersection type.

We have the following soundness result.

Theorem 5.14. *For all assumptions* Γ*,* λ*-terms* M*, and types* $\phi \in \mathbf{T}$*,*

$$\Gamma \vdash M{:}\phi \ \textit{ implies } \ \Gamma \models M{:}\phi.$$

- $\dfrac{x{:}\phi \in \Gamma}{\Gamma \vdash x{:}\phi}$
- $\dfrac{\Gamma \leq \Delta \text{ and } \Delta \vdash M{:}\phi_1 \text{ and } \phi_1 \leq \phi_2}{\Gamma \vdash M{:}\phi_2}$
- $\dfrac{\{\Gamma \vdash M{:}\phi_i\}_{i\in I}}{\Gamma \vdash M{:}\bigwedge_{i\in I}\phi_i}$
- $\dfrac{\Gamma, x{:}\phi_1 \vdash M{:}\phi_2}{\Gamma \vdash \lambda x.M{:}(\phi_1 \to \phi_2)_\perp}$
- $\dfrac{\Gamma \vdash M{:}(\phi_1 \to \phi_2)_\perp \text{ and } \Gamma \vdash N{:}\phi_1}{\Gamma \vdash MN{:}\phi_2}$

Fig. 10. The type assignment system for the lazy λ-calculus.

Proof. The proof goes by induction on the length of the derivation. We prove as illustration the soundness of the rule introducing $\to$. For simplicity, let us denote by $\gamma \models \Gamma$ the fact that $\gamma(x) \models \phi$ for all $x{:}\phi$ in Γ. We have

$$
\begin{aligned}
&\Gamma, x{:}\phi_1 \models M{:}\phi_2\\
&\iff \forall\gamma.(\gamma \models \Gamma, x{:}\phi_1 \implies [\![(M)^*]\!]_\gamma \models \phi_2) \text{ [definitions of } \models \text{ and } [\![-]\!]^{\mathcal{D}}_\gamma]\\
&\implies \forall\gamma.(\gamma \models \Gamma \implies [\![\lambda x.(M)^*]\!]_\gamma \models \phi_1 \to \phi_2) \text{ [definitions of } \models \text{ and } [\![-]\!]_\gamma]\\
&\implies \forall\gamma.(\gamma \models \Gamma \implies [\![\mathsf{up}(\lambda x.(M)^*)]\!]_\gamma \models (\phi_1 \to \phi_2)_\perp \text{ [definitions of } \models \text{ and } [\![-]\!]_\gamma]\\
&\implies \forall\gamma.(\gamma \models \Gamma \implies [\![\mathsf{fold}(\mathsf{up}(\lambda x.(M)^*))]\!]_\gamma \models (\phi_1 \to \phi_2)_\perp \text{ [definitions of } \models \text{ and } [\![-]\!]_\gamma]\\
&\iff \Gamma \models \lambda x.M{:}(\phi_1 \to \phi_2)_\perp \text{ [definitions of } [\![-]\!]^{\mathcal{D}}_\gamma \text{ and } \models] \qquad \square
\end{aligned}
$$

We only prove completeness of the above type assignment system when all types in the assumptions (but not necessarily the predicate) are of the form $\bigwedge_{i\in I}\phi_i$ for some set of finitary types ϕ_i's in $\mathbf{T}_0$. Let $\mathbf{T}_\wedge$ denotes the class of types in $\mathbf{T}$ of the form as described above. We proceed as for the proof of completeness of the subtyping judgment in $\mathbf{T}$. The following lemma plays a role similar to that of Lemma 5.12.

Lemma 5.15. *For all type $\phi \in \mathbf{T}$, λ-terms $M \in \Lambda$ and assumption Γ consisting of pairs $x{:}\psi$ with $x \in \mathrm{V}ar$ and $\psi \in \mathbf{T}_\wedge$, it holds that*

$$\Gamma \vdash M^*{:}\phi \implies \Gamma \vdash M{:}\phi$$

where $(-)^*{:}\Lambda \to \Lambda(\sigma)$ is the translation function defined above for σ being the domain expression $\mathsf{rec}e.(e \to e)_\perp$.

Proof. Let σ be the domain expression $\mathsf{rec}e.(e \to e)_\perp$. The proof is similar to that of Theorem 4.4, and hence organized in two stages. First we consider the case when all types in the assumption Γ are in $\mathbf{T}_\wedge$ but the predicate ϕ assigned to M^* is finitary. We will use induction on the number of variables with non-finitary type in the assumption Γ to prove the result. The base case of the induction is proved in [2], before Theorem 5.14, since all types in Γ and the predicate are finitary.

Next we prove the inductive step. Let ψ be an infinitary type of the form $\bigwedge_{i\in I}\psi_i$ with all ψ_i's finitary. We have:

$$\Gamma, x{:}\psi \vdash M^*{:}\phi$$

$$
\begin{aligned}
&\iff \Gamma, x{:}\bigwedge_I \psi_i \vdash M^*{:}\phi \ [\text{normal form, rule } [\vdash - \leqslant] \text{ from Fig. 5}]\\
&\iff \Gamma, x{:}\bigwedge_I \psi_i \models M^*{:}\phi \quad [\text{soundness Theorem 4.2}]\\
&\implies \exists i \in I.\Gamma, x{:}\psi_i \models M^*{:}\phi \quad [\text{implication (1) of Theorem 4.4}]\\
&\iff \exists i \in I.\Gamma, x{:}\psi_i \vdash M^*{:}\phi \quad [\text{completeness Theorem 4.4}]\\
&\implies \exists i \in I.\Gamma, x{:}\psi_i \vdash M{:}\phi \ [\text{induction hypothesis, } \psi_i \text{ finitary}]\\
&\implies \Gamma, x{:}\psi \vdash M{:}\phi,
\end{aligned}
$$

where the last implication follows by an application of the subsumption rule in Fig. 10 because $\psi \leqslant \psi_i$ in $\mathbf{T}(\sigma)$ and hence also in $\mathbf{T}$ (by Lemma 5.12). From the above we can conclude that

$$\Gamma \vdash M^*{:}\phi \implies \Gamma \vdash M{:}\phi$$

holds for all sequent with ϕ a finitary type.

We conclude the proof by allowing now ϕ to be a general type in $\mathbf{T}$. Using rules $[\to - \bigwedge]$, $[\bigwedge - \to]$, $[\bigvee - \to$ and $[\to - \bigvee]$ and the completely distruibutive axiom, the type ϕ can be proved equivalent in $\mathbf{T}(\sigma)$ to a type of the form $\bigvee_{i\in I}\bigwedge_{j\in J}\phi_{i,j}$, where all $\phi_{i,j}$'s are in $\mathbf{T}_0$. We have

$$
\begin{aligned}
\Gamma \vdash M^*{:}\phi &\iff \Gamma \models M^*{:}\bigvee_{i\in I}\bigwedge_{j\in J}\phi_{i,j} \quad [\text{soundness Theorem 3.8}]\\
&\iff \exists i \in I.\forall j \in J.\Gamma \models M^*{:}\phi_{i,j} \ [\text{definition of } \models, \text{ predicate logic}]\\
&\iff \exists i \in I.\forall j \in J.\Gamma \vdash M^*{:}\phi_{i,j} \quad [\text{completeness Theorem 3.14}]\\
&\iff \exists i \in I.\forall j \in J.\Gamma \vdash M{:}\phi_{i,j} \ [\text{previous part of the proof, } \phi_{i,j} \text{ finitary}]\\
&\implies \exists i \in I.\Gamma \vdash M{:}\bigwedge_{j\in J}\phi_{i,j} \ [[\vdash - \bigwedge]]\\
&\implies \Gamma \vdash M{:}\phi
\end{aligned}
$$

where the last implication follows by an application of the subsumption rule in Fig. 10 because $\bigwedge_{j\in J}\phi_{i,j} \leqslant \phi$ in $\mathbf{T}(\sigma)$ and hence also in $\mathbf{T}$ (by Lemma 5.12). □

We conclude with the expected soundness and completeness result.

Theorem 5.16. *For all type $\phi \in \mathbf{T}$, λ-terms $M \in \Lambda$ and assumption Γ consisting of pairs $x{:}\psi$ with $x \in \mathcal{V}ar$ and $\psi \in \mathbf{T}_\bigwedge$,*

$$\Gamma \vdash M{:}\phi \quad \textit{if and only if } \Gamma \models M{:}\phi.$$

Proof. Soundness follows from Theorem 5.14. Completenss is obtained as follows:

$$
\begin{aligned}
&\Gamma \models M{:}\phi\\
&\quad\iff \forall\gamma.(\gamma \models \Gamma \implies [\![M]\!]^{\mathcal{D}}_\gamma \models \phi) \ [\text{definition of } \models]\\
&\quad\iff \forall\gamma.(\gamma \models \Gamma \implies [\![M^*]\!]^{\sigma}_\gamma \models \phi) \ [\text{definition of } [\![-]\!]^{\mathcal{D}}_\gamma, \sigma = \mathsf{rec}e.(e \to e)_\perp]\\
&\quad\iff \Gamma \models M^*{:}\phi \ [\text{definition of } \models, \text{ and } M^* \in \Lambda(\sigma), \mathbf{T} \subseteq \mathbf{T}(\sigma) \text{ for } \sigma = \mathsf{rec}e.(e \to e)_\perp)]
\end{aligned}
$$

$\iff \quad \Gamma \vdash M^*{:}\phi$ [completeness Theorem 4.4]
$\implies \quad \Gamma \vdash M{:}\phi$ [Lemma 5.15] $\square$

In order to obtain a full completenss result allowing types in **T** to appear also in the assumption, we believe that the assignment system of Fig. 10 should be extended with the following rule:

$$\frac{\{\Gamma, x{:}(\phi_1 \to \cdots (\bigwedge_{j\in J} \phi_{n,j} \to \phi)_\perp \cdots)_\perp \vdash M{:}\phi\}_{J\in Fin(I)}}{\Gamma, x{:}(\phi_1 \to \cdots (\bigwedge_{i\in I} \phi_{n,i} \to \phi)_\perp \cdots)_\perp \vdash M{:}\phi}.$$

This rule will allow a compositional reasoning about infinite filtered intersection at the left-hand side of the $(- \to -)_\perp$ contructor, without referring to union types.

6. Conclusion and future work

We have presented a complete infinitary intersection and union type system for a language of Scott domains. The language of type comprise product, sum, function and lift constructors as well as an infinite union and an infinite intersection constructors. The language of terms is an extension of the untyped λ-calculus with operators associated to the introduction and elimination of each type constructor. Types are interpreted as upper closed subsets of Scott domains (i.e., saturated sets of the Scott topology associated with the Scott domains), whereas terms denote elements of the Scott domains. A subtype relation has been characterized and proved sound and complete with respect to the ordinary subset inclusion relation. Also, an assignment system of types to terms has been axiomatized and proved sound and complete with respect to the set-theoretical 'element-of' relation.

The language of type includes also concrete base types. In this paper we have considered base types that are compact opens of some base Scott-domain. They are available for the type system, the subtype system and the type assignment system. The restriction to compactness of the base type could be relaxed to upper-closed by introducing two extra axiom schema's in the subtype judgment system

$$\Big[e - \bigwedge\Big] \bigcap_{i\in I} u_i = \bigwedge_{i\in I} u_i, \quad \Big[e - \bigvee\Big] \bigcup_{i\in I} u_i = \bigvee_{i\in I} u_i$$

where all u_i's are upper closed subsets of the Scott domain $\varepsilon(e)$. For the lazy lambda calculus we have decided for simplicity not to consider base types. However they could be added without problem, following the results in Sections 3 and 4.

Our work is in the same spirit of Abramsky's logic of domains. In fact all our results rely on Abramsky finitary logical characterization of Scott domains, and extend them to the infinitary propositional case. The expressive power gained by this extension is enormous (at cost of decidability, of course). For example, in our framework is possible to express types representing any subsets not containing a bottom element of a flat domain (a lifted set) whereas Abramsky framework can characterize only the finite ones. In fact, Abramsky logic characterizes the compact open subsets of a Scott domain, whereas our framework characterize all upper closed subsets.

Although we have not treated powerdomain in this paper, they can be added to this framework without any major modifications, according to their infinitary characterization given in [11]. For simplicity we have not considered the coalesced sum and the strict function space here. Their treatment is straightforward, but require the axiomatization of a termination predicate $\mathbf{T}(\phi)$ that holds if and only if $\perp \not\models \phi$.

For example the strict function space is axiomatized as the ordinary function space plus the extra rule "if $\mathbf{T}(\phi)$ then $\mathsf{Top} \to_\perp \phi \leqslant \mathsf{Bot}$".

Our type system includes union types, and inherits from [3] the need of a 'coprimeness predicate' for proving completeness. This implies that to interpret our type system, maintaining completeness, to a larger class of models rather than to a single Scott domain, one has to restrict the syntax of the types. In this paper we have shown how to apply our framework to the class of applicative transition systems in order to obtain a sound and complete infinitary intersection system for the lazy λ-calculus. We have shown how infinitary intersection types can be used for applicative transition system to model behavioral properties like applicative bisimulation.

Another possibility, that still need to be explored in full details, would be to considered only certain kind of union types, namely those generated by a least fixed point operator. Indeed we could consider for the class of applicative transition systems the following language of types:

$$\phi \ ::= \ \mathsf{Top} \mid \mathsf{Bot} \mid t \mid \phi \wedge \phi \mid (\phi \to \phi)_\perp \mid \mu t.\phi,$$

where t is a type variable and the type ϕ in $\mu t.\phi$ is such that t occurs positively in ϕ, meaning that the free type variable t occurs only at the right-hand side in the $\to$ construct. This restriction implies the monotonicity of the interpretation function. More interestingly, but not explored here, would be to consider type variables occurring either positively or negatively in ϕ, as the interpretation of $\mu t.\phi$ would need not to be an open set anymore but a saturated one, thus perfectly fitting in our framework.

A translation of the above language in our type system $\mathbf{T}(\mathsf{rec}e.(e \to e)_\perp)$ for the lazy λ-calculus is given by setting

$$\mu t.\phi \equiv \bigvee_{k\in\omega} \phi_k,$$

where $\phi_0 = \mathsf{Bot}$ and $\phi_{k+1} = \phi[\phi_k/t]$. Since t occurs positively in ϕ, $(\phi_k)_{k\in\omega}$ forms a chain. Furthermore, by $[\to - \bigvee]$, $[\bigvee - \to]$ and $[(-)_\perp - \bigvee]$ it follows that the interpretation function is continuous, and hence the interpretation of $\mu t.\phi$ is indeed the least fixed point of the function associated with ϕ. In this setting, one could, for example, easily deduce the validity and soundness of the following axiom and rules:

$$\phi[\mu t.\phi] \leqslant \mu t.\phi \ [\text{using } [\textstyle\bigvee - E]], \qquad \frac{\phi[\psi/t] \leqslant \psi}{\mu t.\phi \leqslant \psi} \ [\text{using } [\textstyle\bigvee - I]],$$

$$\frac{t \notin FV(\psi)}{\mu t.\phi \wedge \psi = \mu t.(\phi \wedge \psi)} \ [\text{using } [\mathit{distr}]]$$

The types we consider in this paper are infinite. There are however interesting fragments that have a finite and transparent representation [17]. For example, one could consider only recursive types and reworking out our framework either syntactically (as we have done above) or semantically, by considering an interpretation of $\mu t.\phi$ as an appropriate fixed point in the completely distributive lattice of saturated sets of a Scott domain. Since infinitary propositional logics are closely related to first order logics, it would be interesting to use our framework for a topological interpretation of a λ-calculus with bounded existential and universal quantification [12]. For example, if η is environment mapping type variables to saturated sets of a suitable Scott domain, then we can interpret the bounded universal and existential types as follows:

$$[\![\forall t \leqslant \phi.\psi]\!]_\eta = \bigcap \{[\![\psi]\!]_\eta[q/t] \mid q \subseteq [\![\phi]\!]_\eta\},$$

$$[\![\exists t \leqslant \phi.\psi]\!]_\eta = \bigcup\{[\![\psi]\!]_\eta[q/t] \mid q \subseteq [\![\phi]\!]_\eta\}.$$

Note that the above sets are filtered or directed if $[\![\psi]\!]_\eta$ is constant, monotone or antimononotone as function in t, respectively. We have seen that the latter properties follows if t occurs positively or negatively in ψ, respectively. By using the same kind of arguments as in the proof of soundness of rule $[\to - \bigvee]$ and $[\bigwedge - \to]$, we can prove the prenex normal form theorem. Recall that a sentence is prenex if all its quantifiers appear at the front. As a consequence we have a natural syntactic condition for characterizing an equivalence between impredicative and predicative polymorphism.

We conclude by noting that the size of the types can be limited without losing completeness. Since $\mathcal{D}$ is an ω-algebraic cpo, its cardinality is at most $2^{\aleph_0}$, while that of the collection of its saturated sets is at most $2^{|\mathcal{D}|} = 2^{2^{\aleph_0}}$. By the function $[\![-]\!]$ types in $\mathbf{T}$ are identified (up-to logical equivalence) with upper closed subsets of $\mathcal{D}$. Thus we can constrain the size of the index set of the intersection and union type constructors by any cardinal greater than $2^{2^{\aleph_0}}$, that is, under the Generalized Continuum Hypothesis, the infinite cardinal $\aleph_2$.

Acknowledgments

We are thankful to Martin Steffen, Samson Abramsky and Dana Scott for discussions on this topic. We are also grateful to the anonymous referees for suggesting improvements on the content of this paper. In particular we are indebted to one of the referee who found a mistake in the original proof of completeness of the type assignment system and suggested the actual proof presented in this paper.

References

[1] S. Abramsky, Domain theory and the logic of observable properties, Ph.D. Thesis, Queen Mary College, University of London, 1987.
[2] S. Abramsky, The lazy lambda calculus, Research Topics in Functional Programming, Addison-Wesley, Reading, MA, 1990, pp. 65–117.
[3] S. Abramsky, Domain theory in logical form, Annals of Pure and Applied Logic 51 (1) (1991) 1–77.
[4] S. Abramsky, A. Jung, Domain theory, in: S. Abramsky, D.M. Gabbay, T.S.E. Maibaum (Eds.), Handbook of Logic in Computer Science, Semantic Structures, vol. III, Clarendon Press, Oxford, 1994.
[5] F. Barbanera, M. Dezani-Ciancaglini, Intersection and union types, in: Proceedings of TACS'91, Lecture Notes in Computer Science, vol. 526, Springer, Berlin, 1991, pp. 651–674.
[6] F. Barbanera, M. Dezani-Ciancaglini, U. dé Liguoro, Intersection and union types: syntax and semantics, Information and Computation 119 (1995) 202–230.
[7] H. Barendregt, The Lambda Calculus: Its Syntax and Semantics, North-Holland, Amsterdam, 1984.
[8] H. Barendregt, M. Coppo, M. Dezani-Ciancaglini, A filter lambda model and the completeness of type assignment, Journal of Symbolic Logic 48 (4) (1983) 931–940.
[9] M.M. Bonsangue, in: Topological Dualities in Semantics, Electronic Notes in Theoretical Computer Science, vol. 8, Elsevier, Amsterdam, 1997.
[10] M.M. Bonsangue, B. Jacobs, J.N. Kok, Duality beyond sober spaces: topological spaces and observation frames, Theoretical Computer Science 151 (1) (1995) 79–124.
[11] M.M. Bonsangue, J.N. Kok, Towards an infinitary logic of domains: Abramsky logic for transition systems, Information and Computation 155 (1999) 170–201.

[12] L. Cardelli, P. Wegner, On understanding types, data abstraction and polymorphism, Computing Surveys 17 (4) (1985) 471–522.
[13] M. Coppo, M. Dezani-Ciancaglini, A new type-assignment for λ-term, Archiv. Math. Logik, 19, 1978, pp. 139–156.
[14] M. Dezani-Ciancaglini, F. Honsell, Y. Motohama, Compositional characterizations of λ-terms using intersection types, in: Proceedings of MFCS 2000, Lecture Notes in Computer Science, vol. 1893, Springer, Berlin, 2000, pp. 304–313.
[15] K.H. Hofmann, M.W. Mislove, Local compactness and continuous lattices, in: Continuous lattices – Proceedings Bremen 1979, Lecture Notes in Mathematics, vol. 871, Springer, Berlin 1981, pp. 209–248.
[16] P.T. Johnstone, Stone Spaces, Cambridge University Press, Cambridge, 1982.
[17] D. Leivant, Discrete polymorphism, in: Proceedings of 1990 ACM Conference of LISP and Functional Programming, pp. 288–297.
[18] B.C. Pierce, Programming with intersection types, union types, and polymorphism, Technical Report CMU-CS-91-106, Carnegie Mellon University, 1991.
[19] G. Pottinger, A type assignment for the strongly normalizable λ-term, in: H.B. Curry (Ed.), Essays on Combinatory Logic, Lambda Calculus, and Formalism, Academic Press, New York, 1980, pp. 561–577.
[20] G.D. Plotkin, Post-graduate Lecture Notes in Advanced Domain Theory (incorporating the 'Pisa notes'), Department of Computer Science, University of Edinburgh, 1981.
[21] G. Raney, Completely distributive complete lattices, in: Proceedings of the American Mathematical Society, Menasha, WI, and Providence, RI, vol. 3(4), 1952, pp. 677–680.
[22] E. Schröder, in: B.G. Teubner (Ed.), Vorlesunger über die Algebra der Logik, volume I, Leipzig, 1890 (republished in 1966 by Chelsea Publishing Co., New York).
[23] D.S. Scott, Domains for denotational semantics, in: M. Nielsen, E.M. Schmidt (Eds.), 9th International Colloquium on Automata, Languages and Programming; Aarhus, Denmark, Lecture Notes in Computer Science, vol.140, Springer, Berlin, 1982, pp. 577–613.