INFORMATIQUE THÉORIQUE ET APPLICATIONS

FABIO ALESSI
FRANCO BARBANERA

Toward a semantics for the QUEST language

Informatique théorique et applications, tome 28, nº 6 (1994), p. 513-555.

<http://www.numdam.org/item?id=ITA_1994__28_6_513_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 28, n° 6, 1994, p. 513 à 555)

TOWARD A SEMANTICS FOR THE QUEST LANGUAGE (*)

by Fabio ALESSI ([1]) and Franco BARBANERA ([1])

Communicated by G. LONGO

Abstract. – *A model is given for the second order lambda calculus extended with inheritance, bounded quantification, recursive types, constructors and kinds. This language can be viewed as the core of the QUEST language defined by Cardelli [7]. Types are interpreted as intervals of partial equivalence relations. In such a model the meaning of the operator* μ, *the constructor of recursive types, turns out to be the natural one* i.e. *the least fixed point operator.*

Résumé. – *Nous proposons un modèle du second ordre du lambda calcul étendu à l'héritage, la quantification bornée, les types récursifs, les constructeurs et les genres. Ce langage peut être vu comme le noyau du language QUEST défini par Cardelli [7]. Les types sont interprétés comme des intervalles de relations d'équivalence partielle. Dans ce modèle, le sens de l'opérateur* μ, *le constructeur de types récursifs, se trouve être l'opérateur naturel de plus petit point fixe.*

INTRODUCTION

Because of its relevance in the field of object-oriented programming, the theoretical computer science community is currently making more and more efforts in investigating the syntax and sematics of languages extending second order lambda calculus with notions of *subtyping*. In [9] second order lambda calculus was extended in order to support both parametric and subtype (inheritance) polymorphism by means of bounded quantification.

According to the naïve view of inheritance often adopted in the informal justification of type constraint rules, types are regarded as standing for collections of values which interpret the run-time behaviour of programs, subtyping being interpreted by plain set-theoretic inclusion. A more refined view, in which types are modeled as *partial equivalence relations* (*i.e.* relations which are symmetric and transitive but not necessarily reflexive), was taken in [6]. The relational approach has several advantages,

(*) Received June 16, 1991, accepted August 30, 1993.
([1]) Dipartimento di Informatica corso Svizzera, 185 10149 Torino, Italy.

one of them being the possibility of overcoming the difficulties caused by extensional equality betwen terms, which is not achievable by an interpretation of types as sets of values (*see* [19]).

Partial equivalence relations are also the basis of other models for extensions of second order lambda calculus, where the interpretation of each type is built as a relation over a model D for the underlying language of untyped terms, as a limit (in a suitable sense) of a denumerable sequence of approximate relations built following the structure of D. This method, introduced in [12, 13] and exploited in [10] and [3] yields an interpretation not only of bounded quantification and inheritance, but of recursive types as well. The elegance of the construction finds a limitation in the fact that it does not support an extension to include the kind level, essentially because of the non-monotonicity of the type constructor $\rightarrow$ (w.r.t. set theoretical inclusion).

In order to overcome this drawback, in [11] a technical tool was devised: the *intervals* and their particular ordering. This device was originally developed to overcome the difficulty of the non-monotonicity of type constructors in models where types are viewed as ideals. The particular ordering among intervals allows us to define monotonic type operators over intervals out of non-monotonic ones over ideals. The idea of types as intervals of ideals is used in [14] to interpret a notion of "modal" types in a type assignment setting. Types as intervals are also used in [18] to model a language consisting of second order lambda calculus with subtyping, bounded quantification, existential types and a fixed point operator for terms, but not types. The model in [18] has the interesting property of being weakly extensional but not extensional.

What we do in the present paper is to use partial equivalence relations (in the style of [10] and [3]) and their properties, together with the technical device of intervals which is proved to be useful even when partial equivalence relations instead of ideals are considered. We model types as *intervals of partial equivalence relations,* managing to produce a model for the second order lambda calculus with subtyping, bounded quantification, recursive types, kinds and constructors (higher order operations on types). We call this language μ-FunK. It follows from the properties of intervals and their ordering that in our model of μ-FunK all the type constructors are continuous functions: an important consequence of this is that in the model a kind is given to each constructor constant present in the system ($\rightarrow$, $\forall_{\leq}$ and μ). Besides, the operator μ is interpreted in a very natural way, *i.e.* as the minimal fixed point operator over the space of continuous functions.

An apparently odd feature of this model is that the ordering among intervals is the one defined in [11], while in order to model subtyping we use a different ordering, similar to the one defined in [18]. Nonetheless the Amber rule [8], stating the good behaviour of μ with respect to the "subtyping relation ordering", turns out to be sound in the model in spite of the fact that μ is the minimal fixed point operator with respect to the "ordering for constructors".

As in [18], in order to be able to associate a domain of objects to each type, we take as "real" types only the maximal intervals, *i.e.* those intervals that correspond to partial equivalence relations. However, in order to obtain the good properties of the operator μ, we must have the possibility of interpreting a type even as an interval and not only as a maximal interval. We overcome this difficulty by extending the syntax with a predicate over types that is inperpreted as a predicate of maximality over intervals. This is not a real drawback indeed, since if we can derive a type for a term we can always derive the maximality of it simply by assuming the maximality of all the free type variables occurring in it.

We can look at μ-FunK as a relevant fragment of the powerful language QUEST of Cardelli [7]. A semantics for this language has also been proposed in [8]: although Cardelli and Longo succeed in modeling the whole F_ω, inheritance and bounded quantification, they lack an interpretation for μ.

Among the most recent works on modeling significative fragments of the QUEST language it is worth mentioning [1], which is an elegant categorical version of a model for a fragment of QUEST. Of course more constructors could be added to our formal system, and its semantics could be easily extended to encompass these as well. For instance, records and variant types can be dealt with in our framework using a type free domain including summands for records and variants.

Section 1 gives the syntax of μ-FunK and Section 2 the formal definition of model. The domain over which we define partial equivalence relations, the basis of our model, will be described in Section 3. The definition of interval of partial equivalence relations and the interpretation of types and constructors using intervals are done in Section 4. The model of all μ-FunK is in Section 5. Section 6, 7 and 8 will be devoted to the proofs of some non trivial theorems. In Section 9 we present a counter-example to a natural extension of our model to higher order types.

The present paper is an extended version of [2].

1. SYNTAX OF μ-FunK

In this section we shall describe the language μ-FunK as an extension of the second order λ-calculus where a (syntactic) relation of subtyping, denoted by $\leqq$, is defined between types, and where there are also recursive types (built using the μ operator), bounded quantification and kinds.

As usual, the expression $M[N/x]$ will denote the term obtained by replacing the term N for the variable x in M.

DÉFINITION 1.1 *(Kinds, Preconstructors and Preterms)*: The set of *Kinds* is defined by the following rules.

$$K ::= \mathrm{Type} \mid K \Rightarrow K.$$

The set of *preconstructors* is defined by the following rules.

$$C ::= t \mid \mathrm{Top} \mid \rightarrow \mid \mu \mid \quad \forall_{\leqq} \mid \lambda t : K.C \mid C\,(C)$$

where t ranges over the set of constructor variables.

The set of *preterms* is defined by the following rules.

$$M ::= x \mid \lambda x : C.M \mid M\,(M) \mid \Lambda t \leqq C.M \mid M\,(C)$$

where x ranges over the set of term variables.

In the following we shall abbreviate $\rightarrow CC$ with $C \rightarrow C$, $\forall_{\leqq} C\,(\lambda t : \mathrm{Type}.C)$ with $\forall t \leqq C.C$ and $\mu\,(\lambda t : \mathrm{Type}.C)$ with $\mu t.C$.

We shall call *constructors* the preconstructors which have a kind. Types will be the particular constructors which have kind Type. Terms will be the preterms which have a type.

We define three sorts of judgments:

a) typing judgment $\Gamma \vdash u : \zeta$

stating that the term (constructor) u has type (kind) ζ. If a judgment of this sort is about a term (a constructor) it will be called term- (constructor-) judgment.

b) subtyping judgment $\Gamma \vdash \tau \leqq \sigma$

stating that there exists an subtype relation between the types τ and σ.

c) totality judgment $\Gamma \vdash \mathrm{Max}\,(\sigma)$

stating that the type σ is maximal.

Remark: Let us give briefly the motivation for the introduction in the syntax of the predicate on types Max() and of the judgments (*c*). In our model, if we wish to associate a domain of objects to a type we have to model it not with an arbitrary interval, but with a maximal interval. In such a way it can be assimilated to a profinite partial equivalence relation (the particular partial equivalence relations we use in our model) and then be associated to a domain of objects in the standard way. However, if we forced types to be interpreded directly as maximal intervals we would lose, as stated in the introduction, the properties of the operator μ. What we do is then to distinguish at the syntactic level the types as non-maximal and maximal (the ones for which it is possible to prove that the predicate Max() holds). The maximal types will be indeed our "real" types since to them we are able to associate in the model a domain of objects. It is worth to stress however that using this syntax machinery is by no means restrictive, since, as it will be clear from the type formation rules, we can always obtain, given a derivation for a type, a derivation of the maximality of it only by assuming the maximality of the type variables we use in the derivation for that type. Moreover if we use the usual condition of contractiveness of functions on types, by using on them the μ operator we always get maximal types (*see* the remark at the end of this section and Lemma 8.1).

At this point one could wonder why one is forced to distinguish between maximal and non-maximal types at the syntactic level. Would it not be better to use the intervals to construct the recursive types and then simply throw out the non-maximal intervals from the final model?

The motivation for our choice lies in the fact that we wish to model as wide a fragment of the QUEST language as possible. It would be hopeless to try and interpret kinds (for instance Type $\Rightarrow$Type) and constructors if we considered Type to be the set of maximal intervals. If we did so the interpretation of a kind like Type $\Rightarrow$Type would have no structure and, due to this fact, one would have no natural way to interpret in it the constructors.

In the sequel we shall use:

A, A', $\cdots$ as metaviables for kinds,

σ, τ, $\cdots$ as metavariables for constructors,

M, N, $\cdots$ as metavariables for terms.

A *context of constraints* is a set $\mathbf{C}$ of type constraints of the form $t \leqq \sigma$ where t (called the subject of the constraint) is a type variabe and σ (called the type of the constraint and such that $t \notin FV(\sigma)$) is a type. In a context of constraints there are no two constraints with the same subject.

A *basis* is a finite set **B** of assumptions of the form $u_i : \zeta_i$ where u_i (called the subject of the assumption) is a term- (type-) variable and ζ_i is a constructor (kind) and such that no two distinct variables have the same constructor (kind).

A *context of maximality assumptions* is a set **S** of statements (maximality assumptions) of the form $\mathrm{Max}(t)$, where t (called the subject of the maximality assumption) is a type variable.

If H is a context of constraints (or a basis, a context of maximality assumptions) and v is a type constraint (or, respectively, an assumption, a maximality assumption), H, v will denote $H \cup \{v\}$.

A *context* for μ-FunK is a triple $\Gamma = \mathbf{C};\ \mathbf{B};\ \mathbf{S}$ where **C** is a context of constraints, **B** is a basis and **S** is a context of maximality assumptions.

$FV(\Gamma)$ is the set consisting of the subjects which occur in Γ and of the type variables in the types of the type constraints in Γ.

Let us define now what is a *valid* context.

This definition depends obviously on the formation rules we shall define below.

$\varnothing$ *valid* the empty context is a valid context.

$\mathbf{C};\ \mathbf{B},\ t : A;\ \mathbf{S}$ *valid* if $\mathbf{C};\ \mathbf{B};\ \mathbf{S}$ *valid,* $\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash A :$ Kind and $t \notin FV(\mathbf{C};\ \mathbf{B};\ \mathbf{S})$

$\mathbf{C};\ \mathbf{B},\ x : \sigma;\ \mathbf{S}$ *valid* if $\mathbf{C};\ \mathbf{B};\ \mathbf{S}$ *valid,* $\mathbf{C},\ \mathbf{B},\ \mathbf{S} \vdash \sigma$:Type; $\mathrm{Max}(\sigma)$ and $x \notin FV(\mathbf{C};\ \mathbf{B};\ \mathbf{S})$

$\mathbf{C},\ t \leqq \sigma;\ \mathbf{B},\ t :$ Type; $\mathbf{S}$ *valid* if $\mathbf{C};\ \mathbf{B};\ \mathbf{S}$ *valid,* $\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \sigma :$ Type and $t \notin FV(\mathbf{C};\ \mathbf{B};\ \mathbf{S})$

$\mathbf{C},\ \mathbf{B},\ \mathbf{S}\ \mathrm{Max}(t)$ *valid* if there exists A s.t. $\mathbf{C};\ \mathbf{B};\ \mathbf{S}$ *valid,* $\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash A :$ Kind and $\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash t : A$.

We shall group the rules of our system according to which sort of judgment they allow to form and according to the sort of terms involved.

In the following $\Gamma \vdash U;\ U'$ will be short for $\Gamma \vdash U$ and $\Gamma \vdash U'$.

Axioms and start rules

$(Ax\text{-Type})\ \Gamma \vdash \mathrm{Type} : \mathrm{Kind}$	if Γ is a *valid* context.
$(Ax\text{-Top})\ \Gamma \vdash \mathrm{Top} : \mathrm{Type}$	if Γ is a *valid* context.
$(Ax\text{-}\rightarrow)\ \Gamma \vdash \rightarrow : \mathrm{Type} \Rightarrow \mathrm{Type} \Rightarrow \mathrm{Type}$	if Γ is a *valid* context.
$(Ax\text{-}\forall_{\leqq})\ \Gamma \vdash \forall_{\leqq} : \mathrm{Type} \Rightarrow (\mathrm{Type} \Rightarrow \mathrm{Type}) \Rightarrow \mathrm{Type}$	if Γ is a *valid* context.
$(Ax\text{-}\mu)\ \Gamma \vdash \mu : (\mathrm{Type} \Rightarrow \mathrm{Type}) \Rightarrow \mathrm{Type}$	if Γ is a *valid* context.

(Start-C) $\mathbf{C}, t \leqq \sigma; \mathbf{B}; \mathbf{S} \vdash t \leqq \sigma$ if $\mathbf{C}, t \leqq \sigma; \mathbf{B}; \mathbf{S}$ is a *valid* context.
(Start-B) $\mathbf{C}; \mathbf{B}, u : \zeta; \mathbf{S} \vdash u : \zeta$ if $\mathbf{C}; \mathbf{B}, u : \zeta; \mathbf{S}$ is a *valid* context.
(Start-S) $\mathbf{C}; \mathbf{B}; \mathbf{S}, \mathrm{Max}\,(t) \vdash \mathrm{Max}\,(t)$ if $\mathbf{C}; \mathbf{B}; \mathbf{S}, \mathrm{Max}\,(t)$ is a *valid* context.

Rules for constructor and kind formation

$$\text{(Kind-form)} \quad \frac{\Gamma \vdash A : \mathrm{Kind} \quad \Gamma \vdash A' : \mathrm{Kind}}{\Gamma \vdash A \Rightarrow A' : \mathrm{Kind}}$$

$$(\Rightarrow\text{-Intro}) \quad \frac{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash A : \mathrm{Kind}; A' : \mathrm{Kind} \quad \mathbf{C}; \mathbf{B}, t : A; \mathbf{S} \vdash \sigma : A'}{\Gamma \vdash \lambda t : A.\sigma : A \Rightarrow A'}$$

$$(\Rightarrow -\text{Elim}) \quad \frac{\Gamma \vdash \sigma : A \Rightarrow A'; \tau : A}{\Gamma \vdash \sigma\tau : A'}$$

Rules for type constraints

$$(\leqq\text{-Top}) \quad \frac{\Gamma \vdash \sigma : \mathrm{Type}}{\Gamma \vdash \sigma \leqq \mathrm{Top}} \qquad (\leqq\text{-Refl}) \quad \frac{\Gamma \vdash \sigma : \mathrm{Type}}{\Gamma \vdash \sigma \leqq \sigma}$$

$$(\leqq\text{-Trans}) \quad \frac{\Gamma \vdash \sigma \leqq \tau; \tau \leqq \rho}{\Gamma \vdash \sigma \leqq \rho} \qquad (\leqq\text{-}\rightarrow) \quad \frac{\Gamma \vdash \sigma \leqq \sigma'; \tau' \leqq \tau}{\Gamma \vdash \sigma' \rightarrow \tau' \leqq \sigma \rightarrow \tau}$$

$$(\leqq\text{-}\forall_{\leqq}) \quad \frac{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash \sigma \leqq \sigma' \quad \mathbf{C}, t \leqq \sigma; \mathbf{B}, t : \mathrm{Type}; \mathbf{S} \vdash \tau' \leqq \tau}{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash \forall t \leqq \sigma'.\tau' \leqq \forall t \leqq \sigma.\tau}$$

$$(\leqq\text{-}\mu) \quad \frac{\mathbf{C}; \mathbf{B}, t : \mathrm{Type}; \mathbf{S} \vdash \sigma : \mathrm{Type}}{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash \mu t.\sigma = \sigma\,[\mu t.\sigma/t]}$$

($\mu t.\sigma = \sigma\,[\mu t.\sigma/t]$ stands for $\mu t.\sigma \leqq \sigma[\mu t.\sigma/t]$ and $\sigma\,[\mu t.\sigma/t] \leqq \mu t.\sigma$)

$$\text{(Amber)} \quad \frac{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash \mathrm{Max}\,(\sigma); \mathrm{Max}\,(\tau) \quad \mathbf{C}, s \leqq t; \mathbf{B}; \mathbf{S} \vdash \sigma \leqq \tau}{\mathbf{C}; \mathbf{B}; \mathbf{S} \vdash \mu s.\sigma \leqq \mu t.\tau}$$

if $s \notin FV\,(\tau)$ and $t \notin FV\,(\sigma)$.

Rules for maximality

$$(\text{Max-}\forall_{\leqq})\quad \Gamma \vdash \text{Max}\,(\forall_{\leqq})\quad \text{if } \Gamma \text{ is a } \textit{valid} \text{ context.}$$

$$(\text{Max-}\rightarrow)\quad \Gamma \vdash \text{Max}\,(\rightarrow)\quad \text{if } \Gamma \text{ is a } \textit{valid} \text{ context.}$$

$$(\text{Max-abstr})\quad \frac{\left\{\begin{array}{c}\mathbf{C};\ \mathbf{B},\ t : A;\ \mathbf{S},\ \text{Max}\,(t) \vdash \sigma : A';\ \text{Max}\,(\sigma)\\ \mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash A \Rightarrow A' : \text{Kind}\end{array}\right\}}{\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \text{Max}\,(\lambda\, t : A.\sigma)}$$

$$(\text{Max-appl})\quad \frac{\Gamma \vdash \sigma : A \Rightarrow A';\ \tau : A;\ \text{Max}\,(\sigma);\ \text{Max}\,(\tau)}{\Gamma \vdash \text{Max}\,(\sigma\tau)}$$

$$(\text{Max-}\mu)\quad \frac{\Gamma \vdash \sigma : \text{Type};\ \text{Max}\,(\lambda\, s : \text{Type}.\sigma)}{\Gamma \vdash \text{Max}\,(\mu\,(\lambda\, s : \text{Type}.\sigma))}\quad \text{if } \sigma \text{ is contractive in } s\ (\star)$$

$(\star)$ σ is *contractive* in s *iff*

1. $(t \in FV\,(\sigma) \Rightarrow \Gamma \vdash t : \text{Type})$[1].

2. σ is either a type variable different from s or a function type or a constructor application whose reduced form is contractive in s or a recursive type whose body is contractive in s [17].

Term formation rules

$$(\rightarrow\text{-Intro})\quad \frac{\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \sigma : \text{Type};\ \tau : \text{Type}\quad \mathbf{C};\ \mathbf{B},\ x : \sigma;\ \mathbf{S} \vdash M : \tau}{\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \lambda\, x : \sigma.M : \sigma \rightarrow \tau}$$

$$(\rightarrow\text{-Elim})\quad \frac{\Gamma \vdash M : \sigma \rightarrow \tau;\ N : \sigma}{\Gamma \vdash MN : \tau}$$

$$(\forall\text{-Intro})\quad \frac{\left\{\begin{array}{c}\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \text{Max}\,(\sigma);\ \sigma : \text{Type};\ \tau : \text{Type}\\ \mathbf{C},\ t \leqq \sigma;\ \mathbf{B},\ t : \textit{Type};\ \mathbf{S},\ \text{Max}\,(t) \vdash M : \tau\end{array}\right\}}{\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \Lambda\, t \leqq \sigma.M : \forall\, t \leqq \sigma.\tau}\quad \text{if } t \notin FV\,(B)$$

[1] This condition in the definition of contractiveness is not present in the system of [17], since in that system it is not possible to have kind variables of kind different from Type.

$$(\forall\text{-Elim}) \quad \frac{\Gamma \vdash M : \forall t \leqq \sigma.\tau;\ \rho \leqq \sigma;\ \mathrm{Max}(\rho)}{\Gamma \vdash M\,\rho : \tau\,[\rho/t]}$$

$$(\text{Subsump}) \quad \frac{\Gamma \vdash M : \sigma;\ \sigma \leqq \tau;\ \mathrm{Max}(\tau)}{\Gamma \vdash M : \tau}$$

Remark: Notice that the restriction of rule (Max-μ) to contractive types is necessary to obtain the soundness of μ-FunK with respect to the model we shall present. In this model it will be possible to apply the μ operator to all the constructors from Type to Type; for instance it is possible to model even types like $\mu t.t$ where t is a variable of kind Type $\Rightarrow$ Type. The contractiveness is however needed if we want to obtain, using the μ operator, maximal types, *i.e.* the ones we use as "real" types; in fact the condition $(t \in FV(\sigma) \Rightarrow \Gamma \vdash t : \mathrm{Type})$ for contractiveness fails trivially for $\mu t.t$.

The restriction to contractive types is of no relevance from a practical point of view. It can cause some problems only in certain higher order systems, where it might be difficult to decide whether a type is contractive or not.

2. SEMANTICS OF μ-FunK

Let us define now what is the formal semantics for μ-FunK. This will be given in the style of [6], from which we have taken the following definition.

DEFINITION 2.1 (*Kind frame*): A *Kind Frame* is a tuple

$$\mathrm{KIND} = \langle \mathtt{Kinds}, U : \mathtt{Kinds} \to \mathrm{Set},\ \{\Phi_{A,A'} \mid A,\ A' \in \mathtt{Kinds}\},$$
$$\mathtt{Max}(-),\ \mathtt{Type},\ \mathtt{Top},\ \Rrightarrow,\ \rightarrowtail,\ \leq,\ \forall_{\leq},\ \mathbb{\mu}\ \rangle$$

such that

1. $\mathtt{Kinds}$ is a set.
2. U is a total map from Kinds to Set.
3. $\Rrightarrow : \mathtt{Kinds} \times \mathtt{Kinds} \to \mathtt{Kinds}$.
4. $\Phi_{A,A'} : U(A \Rrightarrow A') \to (U(A) \to U(A'))$ is an injective map for all $A,\ A' \in \mathtt{Kinds}$.
5. $\mathtt{Type} \in \mathtt{Kinds}$ and $\mathtt{Top} \in U(\mathtt{Type})$.
6. $\mathtt{Max}(A) \subseteq U(A)$ for all $A \in \mathtt{Kinds}$.
7. $\rightarrowtail \in \mathtt{Max}(U(\mathtt{Type} \Rrightarrow \mathtt{Type} \Rrightarrow \mathtt{Type}))$.
8. $\mathbb{\mu} \in U((\mathtt{Type} \Rrightarrow \mathtt{Type}) \Rrightarrow \mathtt{Type})$.
9. $\forall_{\leq} \in \mathtt{Max}(U(\mathtt{Type} \Rrightarrow (\mathtt{Type} \Rrightarrow \mathtt{Type}) \Rrightarrow \mathtt{Type}))$.

10. $\leqq$ is a partial order over U (Type) with Top as maximum.

The following additional conditions must be satisfied.

Let β, γ, δ, $\varepsilon \in U$ (Type) and F, $G \in U(\mathtt{Type} \Rrightarrow \mathtt{Type})$.

(i) If $\beta \leqq \gamma$ and $\delta \leqq \varepsilon$ then $\gamma \to \delta \leqq \beta \to \varepsilon$ (soundness of rule ($\leqq$ - $\to$))

(ii) If $\beta \leqq \gamma$ and for all $\varphi \leqq \beta$ $F(\varphi) \leqq G(\varphi)$ then $\forall_{\leqq} \gamma F \leqq \forall_{\leqq} \beta G$ (soundness of rule ($\leqq$ $-\forall_{\leqq}$))

(iii) If *a)* $\beta \leqq \gamma$ implies $F(\beta) \leqq G(\gamma)$ and

b) F, $G \in \mathtt{Max}(\mathtt{Type} \Rrightarrow \mathtt{Type})$

then $\mu F \leqq \mu G$ (soundness of rule (Amber))

(iv) $\mathtt{Max}(A \Rrightarrow A') = \{F \in U(A \Rrightarrow A') \mid \forall \sigma \in \mathtt{Max}(A)\, F(\sigma) \in \mathtt{Max}(A')\}$

where

$\gamma \to \delta$ denotes $\Phi_{\mathtt{Type},\,\mathtt{Type}}(\Phi_{\mathtt{Type},\,\mathtt{Type} \Rrightarrow \mathtt{Type}}(\Rrightarrow)(\gamma))(\delta)$,

$\forall_{\leqq} \gamma F$ denotes $\Phi_{\mathtt{Type} \Rrightarrow \mathtt{Type},\,\mathtt{Type}}(H)(F)$

with

$H = \Phi_{\mathtt{Type},\,(\mathtt{Type} \Rrightarrow \mathtt{Type}) \Rrightarrow \mathtt{Type}}(\mathbb{V}_{\leqq})(\gamma)$

and

μF denotes $\Phi_{\mathtt{Type} \Rrightarrow \mathtt{Type},\,\mathtt{Type}}(\mathbb{\mu})(F)$.

When there will be no ambiguity we shall write Type for U(Type).

DEFINITION 2.2 *(Kind Interpretation)*: A *kind-interpretation* $[\![-]\!]$ is a function from kind expressions to Kinds, such that the following conditions are satisfied:

- $[\![\mathtt{Type}]\!] = \mathtt{Type}$
- $[\![A \Rrightarrow A']\!] = [\![A]\!] \Rrightarrow [\![A']\!]$.

A *constructor-environment* is a map η from constructor variables to $\bigcup_{A \in \mathtt{Kinds}} U(A)$.

As usual, given a constructor-environment η, $\eta(u \mid \to t)$ denotes the following constructor-environment

$$\eta(u \mid \to t)(s) = \begin{cases} u & \text{if } s = t \\ \eta(s) & \text{otherwise.} \end{cases}$$

DEFINITION 2.3 *(Constructor-Interpretations)*: A *constructor-interpretation* $[\![\ ]\!]$ in a given kind frame is a function from derivable constructor-judgments and constructor environments which satisfies the following conditions:

- $[\![\Gamma \vdash t : A]\!]_\eta = \eta(t)$
- $[\![\Gamma \vdash \mathrm{Top}]\!]_\eta = \mathtt{Top}$
- $[\![\Gamma \vdash \rightarrow : \mathrm{Type} \Rightarrow \mathrm{Type} \Rightarrow \mathrm{Type}]\!]_\eta = \Rrightarrow$
- $[\![\Gamma \vdash \mu : (\mathrm{Type} \Rightarrow \mathrm{Type}) \Rightarrow \mathrm{Type}]\!]_\eta = \mathfrak{M}$
- $[\![\Gamma \vdash \forall_{\leqq} : \mathrm{Type} \Rightarrow (\mathrm{Type} \Rightarrow \mathrm{Type}) \Rightarrow \mathrm{Type}]\!]_\eta = \mathbb{V}_{\leqslant}$
- $[\![\Gamma \vdash \sigma\tau : A']\!]_\eta = \Phi_{[\![A]\!]\,[\![A']\!]}\,([\![\Gamma \vdash \sigma : A \Rightarrow A']\!]_\eta)\,([\![\Gamma \vdash \tau : A]\!]_\eta)$
- $[\![\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \lambda t : A.\sigma : A \Rightarrow A']\!]_\eta = \Phi^{-1}_{[\![A]\!],\,[\![A']\!]}(f)$,

where $f \in (U([\![A]\!]) \rightarrow U([\![A']\!]))$ is defined by $f(u) = [\![\mathbf{C};\ \mathbf{B},\ t : A;\ \mathbf{S} \vdash \sigma : A]\!]_{\eta(u|\rightarrow t)}$.

We say that a constructor environment η satisfies a context Γ ($\eta \models \Gamma$) <u>iff</u>

a) for all $t : A \in \Gamma \quad \eta(t) \in U([\![A]\!])$

b) if $\mathrm{Max}(t) \in \Gamma$ and $t \leqq \sigma \in \Gamma$ then $\eta(t) \leqslant [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$.

Note that this definition is not circular because when we introduce t in a derivation t is fresh.

DEFINITION 2.4: (*Kind Models*): (i) A *quasi-kind model* is a kind frame for which a constructor interpretation is defined.

(ii) A *kind model* is a quasi-kind model in which rule (Max-μ) is sound.

DEFINITION 2.5 (*Frames*): A *frame* for μ-FunK is a pair

FRAME= $\langle \mathrm{KIND}, \mathrm{DOM} \rangle$ such that

1. KIND is a kind model
2. $\mathrm{DOM} = \langle \{\mathrm{Dom}^a \mid a \in \mathtt{Max}(\mathtt{Type})\},\ \{\chi_{a,b} \mid a, b \in \mathtt{Max}(\mathtt{Type})\},$
$\{\chi_f \mid f \in \mathtt{Max}(\mathtt{Type} \Rrightarrow \mathtt{Type})\},\ \{I_{a,b} \mid a, b \in \mathtt{Max}(\mathtt{Type}),\ a \leqslant b\} \rangle$

such that:

(i) for all $a \in \mathtt{Max}(\mathtt{Type})$ Dom^a is a set.

(ii) for all $a, b \in \mathtt{Max}(\mathtt{Type})$ there is a set of functions from Dom^a to Dom^b, denoted by $\star[\mathrm{Dom}^a \rightarrow \mathrm{Dom}^b]\star$, such that there exists a bijection

$$\chi_{a,b} : \quad \mathrm{Dom}^{a \rightarrow b} \rightarrow \star[\mathrm{Dom}^a \rightarrow \mathrm{Dom}^b]\star .$$

(Notice that $\Rrightarrow \in \mathtt{Max}(U(\mathtt{Type} \Rrightarrow \mathtt{Type} \Rrightarrow \mathtt{Type}))$ and then, since $a \Rrightarrow b$ is a maximal element, $\mathrm{Dom}^{a \rightarrow b}$ is defined).

(iii) for all $f \in \mathtt{Max}(\mathtt{Type} \Rightarrow \mathtt{Type})$ and for all $b \in \mathtt{Max}(\mathtt{Type})$, if $F = \Phi_{\mathtt{Type} \Rightarrow \mathtt{Type}}(f)$ then there exists a subset

$$\star[\prod_{a \in \mathtt{Max}(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F(a)}]\star \subseteqq \prod_{a \in \mathtt{Max}(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F(a)}$$

such that there exists a bijection

$$\chi_f : \quad \mathrm{Dom}^{\forall \leqq b\, F} \to \star[\prod_{a \in \mathtt{Max}(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F(a)}]\star$$

(iv) for all $a,\ b \in \mathtt{Max}(\mathtt{Type})$ s.t. $a \leqslant b$ we have $I_{a,b} : \mathrm{Dom}^a \to \mathrm{Dom}^b$.

A *term environment* satisfying a context Γ with respect to η is a map ξ which to each term variable x such that $x : \sigma \in \Gamma$ (and $\Gamma \vdash \sigma : \mathrm{Type}$) associates an element of $\mathrm{Dom}^{[\![\Gamma \vdash \sigma : \mathrm{Type}]\!]\eta}$.

An *environment* satisfying a context Γ is a pair $\langle \eta, \xi \rangle$ where η is a context environment and ξ is a term environment, both satisfying Γ.

DEFINITION 2.6 (*Term Interpretations*): A *term-interpretation* $[\![-]\!]$ in a given frame is a function from derivable term-judgments and environments which satisfies the following conditions:

- $[\![\mathbf{C};\ \mathbf{B},\ x : \sigma;\ \mathbf{S} \vdash x : \sigma]\!]_{\langle \eta, \xi \rangle} = \xi(x)$
- $[\![\Gamma \vdash MN : \tau]\!]_{\langle \eta, \xi \rangle} = \chi_{\alpha,\beta}([\![\Gamma \vdash M : \sigma \to \tau]\!]_{\langle \eta, \xi \rangle})([\![\Gamma \vdash N : \sigma]\!]_{\langle \eta, \xi \rangle})$

where $\alpha = [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$ and $\beta = [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$

- $\chi_{\alpha,\beta}([\![\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \lambda x : \sigma.M : \sigma \to \tau]\!]_{\langle \eta, \xi \rangle})(p) =$
  $[\![\mathbf{C};\ \mathbf{B},\ x : \sigma;\ \mathbf{S} \vdash M : \tau]\!]_{\langle \eta, \xi(x \mid \to p) \rangle}$

where $\alpha = [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$, $\beta = [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$ and $p \in \mathrm{Dom}^\alpha$

- $[\![\Gamma \vdash M\rho : \tau[\rho/t]]\!]_{\langle \eta, \xi \rangle} =$
  $\chi_f([\![\Gamma \vdash M : \forall_t \leqq \sigma.\tau]\!]_{\langle \eta, \xi \rangle})([\![\Gamma \vdash \rho : \mathrm{Type}]\!]_\eta)$

where $f = \boldsymbol{\lambda}\gamma \in H.[\![\Gamma \vdash \tau : \mathrm{Type}]\!]_{\eta(t \mid \to \gamma)}$ and $[\![\Gamma \vdash \rho : \mathrm{Type}]\!]_\eta \in H$, with $H = \{\beta \in \mathtt{Type} \mid \beta \leqslant [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta\}$

- $\chi_f([\![\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \Lambda t \leqq \sigma.M : \forall t \leqq \sigma.\tau]\!]_{\langle \eta, \xi \rangle})(\gamma) =$
  $[\![\mathbf{C},\ t \leqq;\ \sigma;\ \mathbf{B},\ t\text{: Type};\ \mathbf{S},\ \mathrm{Max}(t) \vdash M : \tau]\!]_{\langle \eta(t \mid \to \gamma), \xi \rangle}$ for $\gamma \in H$

and $f = \boldsymbol{\lambda} \in H.[\![\Gamma \vdash \tau : \mathrm{Type}]\!]_{\eta(t \mid \to \gamma)}$, with $H = \{\beta \in \mathtt{Type} \mid \beta \leqslant [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta\}$

- $[\![\Gamma \vdash M : \mu t.\sigma]\!]_{\langle \eta, \xi \rangle} = [\![\Gamma \vdash M : \sigma[\mu t.\sigma/t]]\!]_{\langle \eta, \xi \rangle}$.

Notice that the above definition of term interpretation makes it possible to prove that equational rules of the calculus like (α), (β) and (ξ) are sound w.r.t. to any model.

3. A TYPE-FREE STRUCTURE

Our model will be based on a suitable type free model D satisfying the equation

$$(\star) \qquad D \cong A + [D \to D]$$

where A is a countable flat domain of atoms (e.g. integers), $+$ represents coalesced sum and $[D \to D]$ the space of all continuous functions (w.r.t. Scott topology) from D to D.

For technical reasons we shall assume that the solution of equation ($\star$) is obtained as an inverse limit of a chain of embeddings of finite cpo's. This will allow the set of intervals of PPER's, to be defined below, to be a Scott domain. Thus we take D to be a *profinite domain* in the sense of [15]. *Embedding-projection pairs* of the form $\langle i \in [D_1 \to D_2], j \in [D_2 \to D_1] \rangle$ are pairs of continuous functions such that for all $d \in D_1$ and $e \in D_2$ $j(i(d)) = d$ and $i(j(e)) \sqsubseteq e$. The construction is described, for instance, in [16]. We may define in the usual way a denumerable family $\langle \varphi_n : D_n \to D, \psi_n : D \to D_n \rangle_{n \in \omega}$ of embedding-projection pairs, where each D_n has finite cardinality. These introduce a *notion of approximation* on the domain D as a denumerable family of continuous mappings $(-)_n : D \to D$, where each $(-)_n$ is the composition $\varphi_n \circ \psi_n$.

Via embedding-projection pairs we identify each D_n with its isomorphic image $\varphi_n(D_n) \subseteqq D$. So we get an ascending chain $\langle D_0 \subseteqq D_1 \subseteqq \cdots \rangle$ of domains. By means of such an identification we manage to simplify our arguments in several of the following proofs, being it possible to get rid of any explicit reference to the embedding-projection pairs $\langle \varphi_n \circ \psi_n \rangle : D_n \to D$.

As a direct consequence of starting from a finite cpo A we get the finiteness of each D_n.

The following properties of D and $(-)_n$ can easily be verified (*see,* e.g., [4]).

PROPOSITION 3.1: *For all $d \in D$ and $n, m \in \omega$*

1. $d_0 = \perp_D$
2. $(d_n)_m = (d_m)_n = d_{\min(n,m)}$
3. $d = \sqcup \{d_n | n \in \omega\}$

If $f \in [D \to D]$ *and* d, $e \in D$, *then*

4. $f_{n+1}(d_k) = f_{n+1}(d_n)$ for $n \leqq k$
5. $(f_{k+1}(d_n))_n = f_{n+1}(d_n)$ *for* $n \leqq k$
6. $f_{n+1}(d_n) = f_{n+1}(d) = (f(d_n))_n$
7. $f = f_{n+1}$ *iff* $\forall d \in D \quad f(d) = (f(d_n))_n$
8. $d \bullet e = \bigsqcup_{n \in \omega} (d_{n+1}(e_n))$.

These and some other useful properties of the notion of approximation will be used in the sequel without explicit mention.

4. A KIND MODEL

In this section we shall build a kind model for μ-FunK and a constructor interpretation for it, using the domain D built as in the previous section.

We shall interpret types as intervals of relations over the domain D and constructors which are not types as functions over such intervals. Some conditions are needed on the relations we use. As it will be shown in detail in what follows, the use of intervals is needed only as a technical tool in order to obtain the continuity (with respect to the Scott topology) of the type constructors $\to$, $\forall_{\leqq}$, and μ.

From the syntax of μ-FunK it is clear that we can consider only maximal types (which can be viewed simply as relations) as real types. This is not a restriction, as pointed out previously.

The notion of approximation defined on D will turn out to be a fundamental tool in the proof of a relevant property of intervals, namely that the application of the operator μ to a maximal contractive type yields a maximal type. This property can be proved by induction on the level of each approximation, following [12], [13], [10] and [3].

A nice feature of our interpretation is that the meaning of the operator μ is the fixed point operator on a space of continuous functions.

The following definition introduces the class of profinite partial equivalence relations (in analogy to profinite domains, *see* [15]), which will be the basis of our model.

DEFINITION 4.1 *(Profinite partial Equivalence Relations)* : A *Profinite Partial Equivalence Relation* (PPER) is a binary relation R over the domain D such that

1. $(\perp, \perp) \in R$
2. R is *symmetric* and *transitive*
3. R is *ω-complete*:

if $\{d_i\}_{i\in\omega}$ and $\{e_i\}_{i\in\omega}$ are increasing chains in D such that $\forall\, i \in \omega\, (d_i, e_i) \in R$ then $(\bigsqcup_i d_i, \bigsqcup_i e_i) \in R$

4. R is *closed under approximations:*

$$(x, y) \in R \Rightarrow \forall\, n \in \omega\, (x_n, y_n) \in R.$$

Notation: From now $x_n, y_n, x_m \ldots$ for $x, y \in D$ will always denote the approximations of x, y in D_n (or D_m). x_i, y_i, x_j will denote instead elements of sequences.

In the previous section we have defined D as the solution of the domain equation

$$D \cong A + [D \to D].$$

It is then possible to show that in such a case we case we can characterize PPER's by replacing conditions 3. and 4. in Definition 4.1 by the following condition:

3'. $(x, y) \in R \Leftrightarrow \forall\, n \in \omega\, (x_n, y_n) \in R$.

On PPER's we can define some standard operations.

DEFINITION 4.2: Let R and S be two PPER's.

(i) $R \to S =_{\text{Def}} \{(x, y) \mid (u, z) \in R \Rightarrow (x \bullet u, y \bullet z) \in S\}$

(ii) $\text{Dom}\,(R) =_{\text{Def}} \{[x]_R \mid (x, x) \in R\}$ where $[x]_R = \{y \mid (x, y) \in R\}$.

DEFINITION 4.3: Let P, P' be PPER's and let $n \in \omega$. We write $P \lhd_n P'$ whenever the following two conditions are satisfied:

1. $P \subseteqq P'$
2. If $(d, e) \in P'$ then $(d_n, e_n) \in P$.

In the following P_n will be the relation $\{(d_n, e_n) \mid (d, e) \in P\} = P \cap (D_n \times D_n)$.

Moreover the following abbreviation will be used $([r, R])_n = [r_n, R_n]$, for $[r_n, R_n]$ PPER's.

Observe that the intersection of an arbitrary family X of PPER's is profinite, so the set of PPER's is a complete lattice with respect to inclusion.

PROPOSITION 4.4 [10]: *For* $P \in PPER$ *and* $n \in \omega$:

(i) $P_n \subseteqq P$

(ii) $P_n \in PPER$

(iii) $P_n \lhd_n P_{n+1}$.

Let us consider now the operator which, given two PPER's, returns the function space between them, as defined in Definition 4.2. It is straightforward to check that this operator is not continuous with respect to the Scott topology (it is not monotone in the first argument). It is possible to gain the continuity of such an operator, beside other things, if we consider intervals of PPER's and not simply PPER's. This technical tool will allow us to have the continuity of the other operators as well.

Let us introduce, in the style of [11], the formal definitiion of what an interval of PPER's is.

DEFINITION 4.5 *(PPER-Intervals)*: (i) A *PPER-interval* $[a, A]$ over D, where $a, A \in$ PPER and $a \subseteqq A \subseteqq D$, is the set of all PPER's P such that $a \subseteqq P \subseteqq A$.

(ii) Cartwright's ordering $\subseteqq_C$ among PPER-intervals is defined in the following way:

$$[a, A] \subseteqq_C [b, B] \ \underline{\textit{iff}}\ [b, B] \subseteqq [a, A] \text{ as sets, } i.e.\ a \subseteqq b \text{ and } B \subseteqq A.$$

It is now easy to check that with respect to the ordering $\subseteqq_C$ the interval $[\{(\bot, \bot)\}, D \times D]$ is the minimum, while maximal elements are all the intervals of the form $[A, A]$. Then there exists a one-one correspondence between PPER's and maximal PPER-intervals.

PPER-int will denote the set of all PPER-intervals over D.

It is not difficult to check that a directed set in PPER-int is a set of intervals $\{[r_i, R_i] \mid i \in I\}$ such that $\{r_i \mid i \in I\}$ is a directed family of PPER's and $\{R_i \mid i \in I\}$ is a filtered family of PPER's, *i.e.* given $i, j \in I$ then there exists $k \in I$ such that $R_k \subseteqq R_i$ & $R_k \subseteqq R_j$. This fact will be used in the following without explicit mention.

PPER-int with the ordering $\subseteqq_C$ is a Scott domain.

THEOREM 4.6: *The partial order* **PPER-int,** $= (PPER - \text{int}, \subseteqq_C)$ *is a Scott domain.*

The proof of this theorem will be the topic of Section 6.

Any function F from PPER-int to PPER-int is naturally associated to a pair of functions from PPER-int to the set of PPER's, *i.e.* we can associate F with $\langle F^+, F^- \rangle$ where if $F([a, A]) = [b, B]$ then $F^+([a, A]) = b$ and $F^-([a, A]) = B$.

Let us define some relevant operations over PPER-int.

Let [PPER-int $\rightarrow$ PPER-int] be the space of continuous functions from PPER-int to PPER-int.

DEFINITION 4.7: (i) $[\rightarrow]$: (PPER-int) $\times$ (PPER-int) $\rightarrow$ PPER-int is defined in the following way:

$[a, A]\,[\rightarrow]\,[b, B] = [A \rightarrow b, a \rightarrow B]$, where $\rightarrow$ on PPER's is defined in 4.2.

(ii) $[\forall_{\subseteq}]$: PPER-int $\rightarrow$ [PPER-int $\rightarrow$ PPER-int] $\rightarrow$ PPER-int is defined in the following way:

$$[\forall_{\subseteq}]([b, B])(F) = [\bigcap_{a \subseteq B} F^+([a, a]), \bigcap_{a \subseteq b} F^-([a, a])]$$

where $[b, B] \in$ PPER-int and $F \in$ [PPER-int $\rightarrow$ PPER-int].

(iii) $[\mu]$: [PPER-int $\rightarrow$ PPER-int] $\rightarrow$ PPER-int is defined as the fixed point operator over

$$[\text{PPER-int} \rightarrow \text{PPER-int}].$$

THEOREM 4.8: *The functions* $[\rightarrow]$, $[\forall_{\subseteq}]$ *and* $[\mu]$ *are continuous.*

The proof of this theorem will be given in Section 7.

Then by means of the technical device of intervals and of their ordering $\subseteq_C$ we manage to obtain the continuity of operators. It is easy to check that the ordering $\subseteq_C$ cannot be used to model also the relation of subtyping. For this purpose we introduce now a new ordering between intervals similar to the one defined in [18].

DEFINITION 4.9 *(The Ordering* $\subseteq_M$*)*: Let $[a, A]$, $[b, B] \in$ PPER-int.

$$[a, A] \subseteq_M [b, B] \quad \underline{\textit{iff}}\ a \subseteq b \text{ and } A \subseteq B.$$

It possible now to define a kind frame

$$\mathrm{KIND} = \langle \mathtt{Kinds},\ \mathrm{U} : \mathtt{Kinds} \to \mathrm{Set},\ \{\Phi_{\mathrm{A},\mathrm{A}'} \mid \mathrm{A},\ \mathrm{A}' \in \mathtt{Kinds}\},$$
$$\mathtt{Max}\,(-),\ \mathtt{Type},\ \mathtt{Top},\ \Rrightarrow,\ \rightarrowtail,\ \leqslant,\ \forall_{\leqslant},\ \mu\!\!\!\mu\ \rangle$$

for μ-FunK based on PPER-int.

DEFINITION 4.10 (*The PPER-int kind frame*): The *PPER-int kind frame* is the kind frame defined in the following way:

`Kinds`	is the set of all Scott domains built by means of the function space operator out of the Scott domain PPER-int.
U	is the forgetful map between Scott domains and Sets.
$\Phi_{A,A'}$	is the identity over the space of functions from $U(A)$ to $U(A')$.
`Type`	is the Scott domain PPER-int.
`Top`	is the interval of PPER's $[\{(\bot,\ \bot)\},\ D \times D]$.
$\Rrightarrow$	is the function space constructor over Scott domains.
`Max (A)`	is the set of all the maximal[2] elements of $U(A)$, where the definition of *maximal* element of $U(A)$ is inductively given in the following way: • the maximal elements of `Type` are the maximal elements of PPER-int • the maximal elements of $A \Rrightarrow A'$ are the functions mapping maximal elements of $U(A)$ into maximal elements of $U(A')$.
$\rightarrowtail$	is $[\rightarrowtail]$ of Definition 4.7.
$\leqslant$	is the ordering $\subseteq_M$.
$\forall_{\leqslant}$	is $[\forall_{\leqslant}]$ of Definition 4.7.
$\mu\!\!\!\mu$	is $[\mu]$ of Definition 4.7.

We have now to prove that the definition above is indeed the definition of a kind frame, *i.e.* that it is correct and that the conditions (i)-(iv) of Definition 2.1 hold.

For the correctness what we first need is the continuity of $\forall_{\leqslant}$, $\rightarrowtail$ and $\mu\!\!\!\mu$, proved in Theorem 4.8. The correctness of the definition of $\mu\!\!\!\mu$ follows directly from this. For the correctness of $\forall_{\leqslant}$ and $\rightarrowtail$ we need also to have that

$$\rightarrowtail \in\ \mathtt{Max}\,(U\,(\mathtt{Type} \Rrightarrow \mathtt{Type} \Rrightarrow \mathtt{Type}))$$

[2] Notice that the name "maximal" is not related, except for Type (and this justifies the name), to the notion of maximality w.r.t. the order of kinds as Scott domains.

and

$$\forall_{\leqq} \in \mathtt{Max}\,(U\,(\mathtt{Type} \Rightarrow (\mathtt{Type} \Rightarrow \mathtt{Type}) \Rightarrow \mathtt{Type})).$$

These facts are stated in the following lemma, whose proof will not be given since it is quite easy.

LEMMA 4.11: *In the kind frame defined above:*

(i) *rule* (Max- $\rightarrow$) *is sound,* i.e. $\twoheadrightarrow \in \mathtt{Max}\,(U\,(\mathtt{Type} \Rightarrow \mathtt{Type} \Rightarrow \mathtt{Type}))$.

(ii) *rule* (Max-$\forall_{\leqq}$) *is sound,* i.e.

$$\forall_{\leqq} \in \mathtt{Max}\,(U\,(\mathtt{Type} \Rightarrow (\mathtt{Type} \Rightarrow \mathtt{Type}) \Rightarrow \mathtt{Type})).$$

THEOREM 4.12: *The* PPER-int kind frame *is a kind frame.*

Having Lemma 4.11 the only interesting part of the proof of the above theorem is the soundness of the rule (Amber) which will be given in Section 8.

We can define now a kind interpetation and a constructor interpretation for μ-FunK based on the PPER-int kind frame: it is sufficient to apply Definitions 2.2 and 2.3 to the PPER-int kind frame. We call these the *PPER-int kind interpretation* and the *PPER-int constructor interpretation.*

We can now prove that the PPER-int quasi-kind model defined above is indeed a *kind model, i.e.* that rule (Max-μ) is sound. The proof is done by induction on the level of approximation, using the fact that for contractive types the approximated interpretation at level $n+1$ of $\sigma \rightarrow \tau$ is completely determined by the approximated interpretation at level n of σ and τ, as follows easily from Proposition 3.1. (*see* [12]).

LEMMA 4.13: *Rule* (Max-μ) *is sound in the PPER-int kind frame.*

The proof of this lemma will be given in Section 8.

With Theorem 4.12 and Lemma 4.13 the proof of the two following theorems is quite straightforward.

THEOREM 4.14 *(Soundness of Max Rules): If we derive* $\Gamma \vdash \mathrm{Max}\,(\sigma)$ *then in the interpretation above* $[\![\Gamma \vdash \sigma : A]\!]_\eta \in \mathtt{Max}\,(A)$, i.e. *the rules concerning* Max *are sound.*

THEOREM 4.15 *(Soundness of* $\leqq$*Rules): If we derive* $\Gamma \vdash \sigma \leqq \tau$ *then in the interpretation above* $[\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta \leqq [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$ i.e. *the rules concerning* $\leqq$ *are sound.*

5. A MODEL FOR μ-FunK

As the last step of our model construction, we define a frame and a term interpretation for μ-FunK.

Let us first recall that a maximal PPER-interval corresponds to a PPER, then in what follows we shall not distinguish between them.

It possible to define a frame for μ-FunK *FRAME*= $\langle$KIND, DOM$\rangle$ based on the PPER-int kind model. Let us first recall what DOM has to be in *FRAME.*

$$\mathrm{DOM} = \langle \{\mathrm{Dom}^a \mid a \in \mathtt{Max}\,(\mathtt{Type})\},\ \{\chi_{a,b} \mid a,\ b \in \mathtt{Max}\,(\mathtt{Type})\},$$
$$\{\chi_f \mid f \in \mathtt{Max}\,(\mathtt{Type} \Rightarrow \mathtt{Type})\},\ \{I_{a,b} \mid a,\ b \in \mathtt{Max}\,(\mathtt{Type}) \text{ and } a \leqq b\}\rangle.$$

DEFINITION 5.1 *(The PPER-int frame)*: The *PPER-int frame* is the frame defined in the following way:

1. KIND is the PPER-int kind model
2. Dom^a is $\mathrm{Dom}\,(a)$
3. $\star[\mathrm{Dom}^a \to \mathrm{Dom}^b]\star$ are the functions from Dom^a to Dom^b represented by elements of $\mathrm{Dom}^{a \to b}$ i.e.

$f \in \star[\mathrm{Dom}^a \to \mathrm{Dom}^b]*$

<u>*iff*</u> there is $[d]_{a \to b}$ such that for all $[e]_a$ $f\,([e]) = [d \bullet e]_b$.

4. $\chi_{a,b}$ is the function which associates to each element of $\mathrm{Dom}^{a \to b}$ the represented function from Dom^a to Dom^b.
5. $\star[\prod_{a \in \mathtt{Max}\,(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F\,(a)}]\star$ is defined in the following way:

$$g \in \star[\prod_{a \in \mathtt{Max}\,(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F\,(a)}]\star$$

$$\underline{\textit{iff}}\ \exists\, c \in D \text{ s.t. } [c]_k \in \mathrm{Dom}^k \quad \text{and} \quad \forall\, p \in \mathtt{Max}\,(\mathtt{Type})$$

$$\text{with } p \leqq b \text{ we have } g\,(p) = [c]_p \text{ where } k = \bigcap_{a \leqq b} F^+\,(a).$$

6. χ_f is defined in the following way: $\chi_f\,([d]_{\forall \leqq b F})\,(a) = [d]_{f\,(a)}$ where $a \in \mathtt{Max}\,(\mathtt{Type}),\ a \leqq b$.
7. $I_{a,b}$ is defined in the following way: if $a \leqq b$ then $I_{a,b}\,([d]_a) = [d]_b$.

It is not difficult to check that $\chi_{a,b}$ and χ_f are bijections. To explain informally the definition of $\star[\prod_{a \in \mathtt{Max}\,(\mathtt{Type}),\, a \leqq b} \mathrm{Dom}^{F\,(a)}]\star$ in point 5. above,

we may say that the action performed by an element, d, of its type on another type (that is on a PPER) consists in forming the equivalence class, with respect to that type, of d.

To define the term interpretation for μ-FunK we shall first give an interpretation for *derivations* of judgment, i.e. an interpretation for judgments depending on the way these are derived, taking into account the possible use of rule (Subsum) as last rule in the derivation. This interpretation of derivation turns out to be also a constructor interpretation since it is possible to prove that different derivations for the same term yield the same interpretation (*see* [5]). Moreover this interpretation will satisfy the conditions of Definition 2.6.

Notice that in this interpretation the subderivations concerning Max are not relevant.

DEFINITION 5.2 *(Derivation Interpretation)*: Let $\langle \eta, \xi \rangle$ be an environment.

Given a derivation Π of a term-judgment $\Gamma \vdash M : \sigma$ we define its interpretation in the environment $\langle \eta, \xi \rangle$ satisfying Γ, $[\![\Pi/\Gamma \vdash MN : \tau]\!]_{\langle \eta, \xi \rangle}$ by induction on the depth of Π.

Base case

We have derived $\mathbf{C}$; $\mathbf{B}$, $x : \tau$; $\mathbf{S} \vdash x : \tau$ using rule (Start-**B**)

$$[\![\Pi/\mathbf{C};\ \mathbf{B},\ x : \tau;\ \mathbf{S} \vdash x : \tau]\!]_{\langle \eta, \xi \rangle} = \xi\,(x).$$

Induction cases

- Last rule is ($\rightarrow$-intro).

Then, using the induction hypothesis, we define

$$\begin{aligned}&[\![\Pi/\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \lambda x : \sigma.M : \sigma \rightarrow \tau]\!]_{\langle \eta, \xi \rangle}\\ &\quad = \chi^{-1}_{a,b}\,(\boldsymbol{\lambda}\,[d] \in a.[\![\Pi_1/\mathbf{C};\ \mathbf{B},\ x : \sigma;\ \mathbf{S} \vdash M : \tau]\!]_{\langle \eta, \xi\,(x \mid \rightarrow [d])\rangle}\end{aligned}$$

where $a = [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$ and $b = [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$.

- Last rule is ($\rightarrow$-elim).

Then, using the induction hypothesis, we define

$$\begin{aligned}&[\![\Pi/\Gamma \vdash MN : \tau]\!]_{\langle \eta, \xi \rangle}\\ &\quad = \chi_{a,b}\,([\![\Pi_1/\Gamma \vdash M : \sigma \rightarrow \tau]\!]_{\langle \eta, \xi \rangle})\,([\![\Pi_2/\Gamma \vdash N : \sigma]\!]_{\langle \eta, \xi \rangle})\end{aligned}$$

where $a = [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$ and $b = [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$.

• Last rule is ($\forall$-Intro).

Then, using the induction hypothesis, we define

$$[\![\Pi/\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \Lambda t \leqq \sigma.M : \forall t \leqq \sigma.\tau]\!]_{\langle \eta, \xi \rangle} = \chi_f^{-1}\,(\boldsymbol{\lambda}\,\alpha \in H.K)$$

where $f = \boldsymbol{\lambda}\,\beta \in \mathtt{Type}.[\![\mathbf{C};\ \mathbf{B},\ t : \mathrm{Type};\ \mathbf{S} \vdash \tau : \mathrm{Type}]\!]_{\eta\,(t|\to\beta)}$,

$$H = \{c \in \mathtt{Max}\,(\mathtt{Type}) \,|\, c \leqslant [\![\mathbf{C};\ \mathbf{B};\ \mathbf{S} \vdash \sigma : \mathrm{Type}]\!]_\eta\}$$

and

$$K = [\![\Pi_1/\mathbf{C},\ t \leqq \sigma;\ \mathbf{B},\ t : \mathrm{Type};\ \mathbf{S},\ \mathrm{Max}\,(t) \vdash M : \tau]\!]_{\langle \eta\, t|\to\alpha),\, \xi \rangle}.$$

• Last rule is ($\forall$-Elim).

$$\begin{aligned}&[\![\Pi/\mathbf{\Gamma} \vdash M\,\rho : \tau\,[\rho/t]]\!]_{\langle \eta, \xi \rangle}\\ &\quad = \chi_f\,([\![\Pi_1/\Gamma \vdash M : \forall t \leqq \sigma.\tau]\!]_{\langle \eta, \xi \rangle})\,[\![\Gamma \vdash \rho : \mathrm{Type}]\!]_\eta)\end{aligned}$$

where $f = \boldsymbol{\lambda}\beta \in \mathtt{Max}\,(\mathtt{Type}).[\![\mathbf{C};\ \mathbf{B},\ t : \mathrm{Type};\ \mathbf{S}, \vdash \tau : \mathrm{Type}]\!]_{\eta\,(t|\to\beta)}$.

• Last rule is (Subsump).

Then, using the induction hypothesis, we define

$$[\![\Pi/\Gamma \vdash M : \tau]\!]_{\langle \eta, \xi \rangle} = I_{a,b}\,([\![\Pi_1/\Gamma \vdash M : \sigma]\!]_{\langle \eta, \xi \rangle})$$

where $a = [\![\Gamma \vdash \sigma : \mathrm{Type}]\!]_\eta$ and $b = [\![\Gamma \vdash \tau : \mathrm{Type}]\!]_\eta$.

We can prove that this definition is correct taking into account the shapes of the term formation rules, Lemmas 4.14, 4.15 and the fact that D is a model of pure λ-calculus (this assures that the functions defined above using the metalinguistic abstraction $\boldsymbol{\lambda}$ are well defined).

It is easy to see that, in general, a term judgment can be obtained by several different derivations. This means that to obtain a term interpretation for μ-FunK from the derivation interpretation it is necessary to prove that the interpretation of a derivation of a term judgment depends only on the term judgment itself. This property, called coherence, can be proved by a cumbersome induction on derivations, following [6].

THEOREM 5.3 *(Soundness): The rules of μ-FunK are sound in the PPER-int model.*

The only difficult cases to be considered in the proof of the above theorem are the ones taken into account in Theorems 4.14 and 4.15.

6. PPER-int IS A SCOTT DOMAIN

This Section will be devoted to the proof of Theorem 4.6.

To prove that PPER-Int is a Scott domain we need some results about the complements in $D \times D$ of the PPER's, which we will refer to as *Co*-PPER's. In particular, we will prove that the set of *Co*-PPER's forms a Scott domain with respect to set-theoretic inclusion.

DEFINITION 6.1 (*Co*-PPER's): Let D be a Scott domain. *A* set $R^* \subseteq D \times D$ is a *Co*-PPER *iff* $R = D \times D \backslash R^*$ is a PPER.

LEMMA 6.2: *A set* $R^* \subseteq D \times D$ *is a* Co-*PPER iff the following conditions are satisfied:*

(i) $(x, y) \in R^* \Leftrightarrow (y, x) \in R^*$

(ii) $(x, y) \in R^* \ \& \ (y, z) \notin R^* \Rightarrow (x, z) \in R^*$

(iii) $(x, y) \in R^* \Leftrightarrow \exists n \in \omega.(x_n, y_n) \in R^*$.

Proof: $\Rightarrow$) Let R^* be the complement in $D \times D$ of a PPER R. Let us prove that conditions (i), (ii) and (iii) are satisfied.

(i) $(x, y) \in R^* \Leftrightarrow (x, y) \notin R \Leftrightarrow (y, x) \notin R \Leftrightarrow (y, x) \in R^*$.

(ii) Let $(x, y) \in R^*$ and $(x, z) \notin R^*$. If we assume $(y, z) \notin R^*$ it follows that $(x, y) \in R$ by the transitivity of R and the fact that $(x, z) \in R$, $(y, z) \in R$.

Thus we get $(y, z) \in R^*$.

(iii) ($\Rightarrow$) If $\forall n \in \omega \ (x_n, y_n) \notin R^*$, then $(x, y) \in R$ (because $\forall n \in \omega \, (x_n, y_n) \in R$). Therefore $(x, y) \in R^* \Rightarrow \exists n \in \omega. \ (x_n, y_n) \in R^*$. ($\Leftarrow$) Let $(x, y) \notin R^*$. Then $(x, y) \in R \Rightarrow \forall n \in \omega \, (x_n, y_n) \in R \Rightarrow \forall n \in \omega$ $(x_n, y_n) \notin R^*$.

$\Leftarrow$) Similarly to case $\Rightarrow$). $\square$

Remark: Note that, due to the (iii) ($\Leftarrow$) point of the above lemma, if $(a, b) \in R^*$ for $a, b \in D_n$, then R^* captures all pairs (x, y) such that $x_n = a$ and $y_n = b$.

LEMMA 6.3: *Let* $\{R_i^* \mid i \in I\} \subseteq$ Co-*PPER. Then* $\bigcup_{i \in I} R_i^*$ *is a* Co-*PPER.*

Proof: Trivial, since the intersection of PPER's is a PPER. $\square$

From Lemma 6.3 it follows that *Co*-PPER is a consistently complete cpo with respect to the set-theoretic inclusion. To prove that it is also algebraic

we need to isolate a basis of finite elements in *Co*-PPER's. For this purpose we introduce the notion of n-minimal *Co*-PPER.

DEFINITION 6.4: Let $Q^* \subseteqq D \times D$ be a *Co*-PPER and $n \in \omega$.

(i) $Q_n^* \subseteqq D \times D$ is the relation defined by

1. $\forall u, v \in D_n. (u, v) \in Q_n^* \Leftrightarrow (v, u) \in Q_n^*$
2. $\forall u, v, w \in D_n. (u, v) \in Q_n^* \;\&\; (v, w) \notin Q_n^* \Rightarrow (u, w) \in Q_n^*$
3. $\forall x, y \in D. (x, y) \in Q_n^* \Leftrightarrow (x_n, y_n) \in Q_n^*$

(ii) Q^* is *n-minimal* <u>*iff*</u> $Q^* = Q_n^*$ (that is $(x, y) \in Q^* \Leftrightarrow (x_n, y_n) \in Q^*$).

The above one is not a good definition unless one proves that Q_n^* is indeed a *Co*-PPER. That is what we shall do in the following Lemma 6.5.

We denote with $(Co\text{-PPER})_n$ the set of n-minimal *Co*-PPER's.

Remark: In order to give an intuition about what a minimal *Co*-PPER Q^* is, consider that the natural way to approximate Q^* amounts to taking its truncation at level n, namely $Q^* \cap (D_n \times D_n)$. But this is not a *Co*-PPER (*see* the previous remark). However we manage to complete $Q^* \cap (D_n \times D_n)$ to a *Co*-PPER Q_n^* by adding all the pairs which are approximated by some of its elements. Q_n^* is minimal in the sense that it is contained in every *Co*-PPER R^* containing $Q^* \cap (D_n \times D_n)$.

LEMMA 6.5: (i) *Let Q^* be a* Co-*PPER. Then Q_n^* is a* Co-*PPER as well.*

(ii) $Q_n^* \subseteqq Q^*$.

(iii) $Q^* = \bigcup_{n \in \omega} Q_n^*$.

Proof:

(i) $(x, y) \in Q_n^* \Leftrightarrow (x_n, y_n) \in Q_n^* \Leftrightarrow (y_n, x_n) \in Q_n^* \Leftrightarrow (y, x) \in Q_n^*$.

$$(x, y) \in Q_n^* \;\&\; (y, z) \notin Q_n^* \Rightarrow (x_n, y_n) \in Q_n^* \;\&\; (y_n, z_n) \notin Q_n^*$$
$$\Rightarrow (x_n, z_n) \in Q_n^* \Rightarrow (x, z) \in Q_n^*.$$

We have then proved that conditions (i) and (ii) of Lemma 6.2 hold. Condition (iii) of the same lemma holds trivially. Thus Q_n^* is a *Co*-PPER.

(ii) Let $(x, y) \in Q_n^*$. Then $(x_n, y_n) \in Q^*$, so by the remark after Lemma 6.2 $(x, y) \in Q^*$.

(iii) The left to right inclusion follows by (ii). For the converse, let $(x, y) \in Q^*$. By Lemma 6.2 there exists $m \in \omega$ such that $(x_m, y_m) \in Q^*$, hence $(x_m, y_m) \in Q_m^*$. By definition of Q_n^* it follows that $(x, y) \in Q_m^*$. □

LEMMA 6.6: $\forall n \in \omega\,(\text{Co-}PPER)_n$ *has finite cardinality.*

Proof: The map $Q^* \mapsto Q^* \cap D_n \times D_n$ is a bijection between $(Co\text{-PPER})_n$ and binary relations over D_n satisfying (i) and (ii) of Lemma 6.2, of which there exists only a finite number because of the finiteness of D_n. Therefore $(Co\text{-PPER})_n$ is a finite set. □

LEMMA 6.7: *Let* $Q^* \in (\text{Co-}PPER)_n$. *Then* Q^* *is a finite element in* $(\text{Co-}PPER, \subseteqq)$.

Proof: Let $Q^* \subseteqq \bigsqcup_{i \in I} R_i^*$, where $\{R_i^* \mid i \in I\}$ is a directed set of *Co*-PPER's.

Let $Q = Q^* \cap (D_n \times D_n)$. The number of pairs (x, y) such that $(x, y) \in Q$ is finite. Thus there exists $j \in I$ such that $Q \subseteqq R_j^*$ and hence $Q^* \subseteqq (R_j^*)_n$ because Q^* is a n-minimal *Co*-PPER. Now, by Lemma 6.5 (ii), we have that $(R_j^*)_n \subseteqq R_j^*$. □

THEOREM 6.8: $(\text{Co-}PPER, \subseteqq)$ *is a Scott domain.*

Proof: By Lemma 6.3 it follows that *Co*-PPER is a consistently complete cpo. Besides, by Lemmas 6.5 (iii) and 6.7 *Co*-PPER is algebraic. Lemma 6.6 guarantees that the basis of finite elements is countable. □

In order to be able to give the full proof of Theorem 4.6 we have still to show that $(\text{PPER}, \subseteqq)$ is a Scott domain, and to this a further technical lemma is needed.

LEMMA 6.9: *Let* $\boldsymbol{R} = \{R_i \mid i \in I\}$ *be a direct family of PPER's. Then the lub of* $\boldsymbol{R}$, $\sqcup\boldsymbol{R}$ *is the PPER defined by the following condition:*

$(x, y) \in \sqcup\boldsymbol{R} \Leftrightarrow \forall n \in \omega \quad \exists i \in I$ *such that* $x_n \, R_i \, y_n$.

Proof: Clearly if S is a PPER such that $\forall i \in I \; R_i \subseteqq S$, then $\sqcup\boldsymbol{R} \subseteqq S$. Thus what we need to prove is only that $\sqcup\boldsymbol{R}$ is a PPER, i.e. that conditions 1-4 of Definition 4.1 are satisfied. We shall check only condition 3 since it is the only one not trivial.

Let $\{x^k\}_{k \in \omega}$ and $\{y^k\}_{k \in \omega}$ be two increasing chains in D such that $\forall k \in \omega\,(x^k, y^k) \in \sqcup\boldsymbol{R}$.

We must prove then that $(\bigsqcup_{k \in \omega} x^k, \bigsqcup_{k \in \omega} y^k) \in \sqcup\boldsymbol{R}$, *i.e.*

$$(\star) \qquad \forall n \in \omega \quad \exists i \in I.((\bigsqcup_{k \in \omega} x^k)_n, (\bigsqcup_{k \in \omega} y^k)_n) \in R_i.$$

Let $k \in \omega$. From the definition of $\sqcup \boldsymbol{R}$ it follows:

$$\forall n \in \omega \quad \exists i_k \in I \text{ such that } ((x^k)_n,\ (y^k)_n) \in R_{i_{k,n}}.$$

Because of the finite cardinality of D_n there must exist k such that

$$(\bigsqcup_{k\in\omega} x^k)_n = (x^k)_n \qquad \text{and} \qquad (\bigsqcup_{k\in\omega} y^k)_n = (y^k)_n.$$

Hence condition $(\star)$ is satisfied. $\square$

THEOREM 6.10: *$(PPER,\ \subseteqq)$ is a Scott domain with $\bigcup_{n\in\omega} (PPER)_n$ (where $(PPER)_n = \{P \cap (D_n \times D_n) \mid P \in PPER\}$) as set of its finite elements.*

Proof: By Lemma 6.9 PPER is a cpo. Its completeness follows by the fact that PPER is closed for arbitrary intersection, by defining, for each $R,\ S \in$ PPER, $R \sqcup S = \cap\{P \in \text{PPER} \mid R \cup S \subseteqq P\}$.

To prove its ω-algebraicity first notice that an element of (PPER)$_n$ is of the form $P_n = P \cap (D_n \times D_n)$ with $P \in$ PPER and hence a finite subset of $D \times D$. (PPER)$_n$ is a set of finite elements of PPER. It remains now to show that, for each $P \in$ PPER, $P = \bigsqcup_{n\in\omega} P_n$; this is done by the following equivalences

$$(x,\ y) \in P \Leftrightarrow$$
$$\forall n \in \omega.\quad (x_n,\ y_n) \in P \Leftrightarrow \forall n \in \omega.\quad (x_n,\ y_n) \in P_n \Leftrightarrow (x,\ y) \in \bigsqcup_{n\in\omega} P_n.$$

Notice that because of the finiteness of D_n, the elements P_n are countable. $\square$

Theorem 4.6 can now be proved starting from theorems 6.8 and 6.10 and by proving that PPER is isomorphic to $(X,\ \subseteqq_X)$, defined below, which, by Proposition 6.12, is a Scott domain.

DEFINITION 6.11: $(X,\ \subseteqq_X)$ is the posed defined in the following way:
$-X = \{(R,\ S^*) \mid R \in \text{PPER},\ S^* \in \text{Co-PPER and } R \cap S^* = \varnothing\}$.
$-(R,\ S^*) \subseteqq_X (Q,\ T^*)$ <u>*iff*</u> $R \subseteqq Q$ & $S^* \subseteqq T^*$.

PROPOSITION 6.12: *$(X,\ \subseteqq_X)$ is a Scott domain.*

Proof: We first prove that $(X,\ \subseteqq_X)$ is a cpo.

Let us consider an ascending chain $\{(R_i, S_i^*)\}_{i\in\omega}$. In order to prove that $(\bigsqcup_{i\in\omega} R_i, \bigcup_{i\in\omega} S_i)$ is the lub of the chain we must check that $(\bigsqcup_{i\in\omega} R_i) \cap (\bigcup_{i\in\omega} S_i) = \varnothing$. By contradiction, let $(x, y) \in (\bigsqcup_{i\in\omega} R_i) \cap (\bigcup_{i\in\omega} S_i)$, then there exists $n \in \omega$ such that $(x_n, y_n) \in \bigcup_{i\in\omega} S_i^*$. Hence $(x_n, y_n) \in S_k^*$ for a certain $k \in \omega$. If $(x, y) \in \bigsqcup_{i\in\omega} R_i$ then $(x_n, y_n) \in \bigsqcup_{i\in\omega} R_i$. Hence $(x_n, y_n) \in R_h$ for a certain $h \in \omega$ [since $\bigsqcup_{i\in\omega} R_i \cap (D_n \times D_n) = \bigcup_{i\in\omega} R_i \cap (D_n \times D_n)$]. Now, if $m > h, k$, then $(x_n, y_n) \in R_m \cap S_m^*$. Contradiction with the definition of X.

Once we know that $(X, \subseteqq_X)$ is indeed a cpo, Lemmas 6.7, 6.8 and Theorem 6.10 allow us to infer that it is also ω-algebraic, with the pairs (P_n, Q_n^*) as finite elements, where $P_n \in (\text{PPER})_n$ and $Q_n^* \in (Co\text{-PPER})_n$.

To be a Scott domain X has now only to be proved to be bounded complete. Then let (R_1, S_1^*) and (R_2, S_2^*) be such that there exists a bound $(R, S^*) \in X$ for them. This means that $R_1, R_2 \subseteqq R$ and $S_1^*, S_2^* \subseteqq S$. Since $(R, S^*) \in X$ we have that $R \cap S^* = \varnothing$ and hence $(R_1 \sqcup R_2) \cap (S_1^* \cup S_2^*) = \varnothing$. We can now conclude that $(R_1 \sqcup R_2, S_1^* \cup S_2^*) \in X$. It is immediate to check that such an element is the lub of (R_1, S_1^*) and (R_2, S_2^*). □

Proof of Theorem 4.6: The thesis of Theorem 4.6 follows now by showing that $(X, \subseteqq_X)$ is isomorphic to (PPER-int, $\subseteqq_C$).

Let us consider the map α : PPER-int$\rightarrow X$ defined as follows:

$\alpha([r, R]) = (r, R^*)$ (R^* being the complement of R in $D \times D$, and hence a *Co*-PPER).

It is immediate to check that α is an isomorphism. □

Remark: Notice that a directed set $\{[r_i, R_i]\}_{i\in\omega}$ in PPER-int is such that $\{r_i\}_{i\in\omega}$ is a directed set in PPER and $\{R_i^*\}_{i\in\omega}$ is a directed set in *Co*-PPER, that is $\{R_i\}_{i\in\omega}$ is a filtered set in PPER. In particular $\sqcup\{[r_i, R_i]\}_{i\in\omega} = [\sqcup\{r_i\}_{i\in\omega}, \cap\{R_i\}_{i\in\omega}]$.

7. CONTINUITY OF THE TYPE CONSTRUCTORS

In this fairly technical section we shall prove that the constructors $\rightarrow$ and $\forall_{\leqq}$ are continuous over PPER-int. The proof for the type constructor

μ will not be given since it follows immediately by the interpretation of μ as least fixed point operator.

Continuity of the type constructor $\rightarrow$.

Let us begin with a technical lemma showing that the operator $\rightarrow$ is well-behaved with respect to lub's of directed sets and intersections of filtered sets of PPER's.

LEMMA 7.1: *Let* $\boldsymbol{R}= \{R_i \mid i \in I\}$ *be a direct set of PPER's,* $\boldsymbol{Q}= \{Q_j \mid j \in J\}$ *be a filtered set of PPER's and* S *be a PPER. Then the following properties hold:*

(i) $S \rightarrow (\sqcup\boldsymbol{R}) = \bigsqcup_{i\in I} (S \rightarrow R_i)$

(ii) $S \rightarrow (\cap\boldsymbol{Q}) = \bigcap_{j\in J} (S \rightarrow Q_i)$

(iii) $(\sqcup\boldsymbol{R}) \rightarrow S = \bigcap_{i\in I} (R_i \rightarrow S)$

(iv) $(\cap\boldsymbol{Q}) \rightarrow S = \bigsqcup_{j\in J} (Q_j \rightarrow S)$.

Proof: We only prove (iii) since the proofs of the other properties are quite similar.

The proof that $\sqcup\boldsymbol{R}\rightarrow S \subseteqq \bigcap_{i\in I} (R_i \rightarrow S)$ is immediate since $R_i \subseteqq \sqcup\boldsymbol{R} \Rightarrow \sqcup\boldsymbol{R}\rightarrow S \subseteqq R_i \rightarrow S$. To prove the opposite inclusion it is sufficient to show that, due to the ω-algebraicity of PPER

$$\forall n \in \omega \quad \bigcap_{i\in I} (R_i \rightarrow S) \cap (D_n \times D_n) \subseteqq (\sqcup\boldsymbol{R}\rightarrow S) \cap (D_n \times D_n).$$

Let $(x, y) \in \bigcap_{i\in I} (R_i \rightarrow S) \cap (D_n \times D_n)$. To prove that $(x, y) \in \sqcup\boldsymbol{R}\rightarrow S$ let $(d, e) \in \sqcup\boldsymbol{R}$.

Then $\forall m \in \omega\, \exists i \in I.\ (d_m, e_m) \in R_i$. In particular there exists $j \in I$ such that $(d_{n-1}, e_{n-1}) \in R_j$. Since $(x, y) \in (R_j \rightarrow S)$, by the assumption that $(x, y) \in \bigcap_{i\in I} (R_i \rightarrow S)$, we get $(x \bullet d_{n-1}, y \bullet e_{n-1}) \in S$. By the properties of the domain D, $(x \bullet d, y \bullet e) = (x \bullet d_{n-1}, y \bullet e_{n-1})$ and hence $(x \bullet d, y \bullet e) \in S$. $\square$

THEOREM 7.2: $[\rightarrow] : (PPER\text{-}int) \times (PPER\text{-}int) \rightarrow (PPER\text{-}int)$ *is continuous.*

Proof: Let $\{[r_i, R_i] \mid i \in I\}$ be a direct set of PPER-int and $[s, S]$ a PER-int. Then

$$\begin{aligned}
&\bigsqcup_{i\in I}([r_i, R_i] \to [s, S]) \\
&\quad = \bigsqcup_{i\in I}([R_i \to s, r_i \to S]) \\
&\quad = [\bigsqcup_{i\in I} R_i \to s, \bigcap_{i\in I} r_i \to S] \\
&\qquad \text{by the remark after the proof of Theorem 4.6} \\
&\quad = [(\bigcap_{i\in I} R_i) \to s, (\bigsqcup_{i\in I} r_i) \to S] \text{ by (iii) and (iv) of the previous lemma} \\
&\quad = [\bigsqcup_{i\in I} r_i, \bigcap_{i\in I} R_i] \to [s, S] \\
&\quad = (\bigsqcup_{i\in I}[r_i, R_i]) \to [s, S].
\end{aligned}$$

In a similar way, by using (i) and (ii) of the Lemma 7.1, it is possible to prove that

$$\bigsqcup_{i\in I}([s, S] \to [r_i, R_i]) = [s, S] \to \bigsqcup_{i\in I}[r_i, R_i]. \quad \square$$

Continuity of the type constructor $\forall_{\leqq}$.

We turn now our attention to the $\forall_{\leqq}$ constructor.

To prove the continuity of such a constructor we need some extra mathematical apparatus (which is linked to Lawson topology of Scott domains: *see* e.g. [11], [18]).

Let T denote the infinite oriented complete binary tree.

Let $\{e_n \mid n \in \omega\}$ be a numbering of the finite elements of PPER-int.

A path in T is intended starting always from the root.

We label now the edges of T in the following way: at the level n, we label with e_n each edge which descends rightwards and with $\neg e_n$ each edge which descends leftwards.

To each element x of PPER-int it is associated a path in T: corresponding to a level n, the direction is fixed according to if $e_n \subseteqq_C x$ (rightwards) or $e_n \nsubseteqq_C x$ (leftwards).

It is worth pointing out that, although for each element of PPER-int there exists a corresponding path in T, the vice versa fails: for example if e_n and e_m are two incompatible finite elements (*i.e.* $e_n \sqcup e_m$ is not defined) a path descending rightwards at levels n and m cannot correspond to any element.

We say that a subtree T' of T is *uniform* *iff* every internal node has two sons.

Notation: For a given path π in T, we write $e_n \in \pi$ descends rightwards at level n. Otherwise we write $\neg e_n \in \pi$ (or $e_n \notin \pi$).

DEFINITION 7.3: *a*) A path π is *consistent* iff there exists $x \in$ PPER-int which satisfies it, *i.e.* $e_n \in \pi \Leftrightarrow e_n \subseteq_C x$.

Let π be a consistent path in T.

b) $\sqcup\pi =_{\text{def}} \sqcup \{e \mid e \in \pi\}$.

c) π is *maximally-consistent* *iff* there exists a maximal element $x \in$ PPER-int which satisfies it.

d) Let $d \in$ PPER-int. π is *d-maximally-consistent* *iff* there exists a maximal element $z \in$ PPER-int such that $d \subseteq_C z$ and z satisfies π.

e) Given a PPER R a path is *R-maximally consistent* *iff* it is $[\{(\bot, \bot)\}, R]$-maximally consistent.

LEMMA 7.4: *Let* $(d, e) \in D_n \times D_n$. *Then the set of the* n*-minimal* Co-*PPER's* Q_n^* *such that* $(d, e) \in Q_n^*$ *is finite.*

Proof: Immediate consequence of Lemma 6.6. □

Notation: Given a path π in T, π_n will denote the sub-path of π going from the root to the level n.

LEMMA 7.5: *Let* π *be a path in* T *associated to a non-maximal element* $x = [r, R] \in$ *PPER-int. Then there exists a finite sub-path* ρ *of* π *which is not maximally consistent.*

Proof: If $[r, R]$ is non-maximal, i.e. r is strictly contained in R, then there exists an element (x, y) in $D \times D$ such that $(x, y) \in R$ and $(x, y) \notin r$. The latter fact implies that there must exists an n such that $(x_n, y_n) \notin r$, while $(x_n, y_n) \in R$.

Let P be the PPER generated by (x_n, y_n) and let $\{Q_i^* \mid i \in I\}$ be the set of n-minimal *Co*-PPER's containing (x_n, y_n), which is clearly a finite

set. Besides, let us denote by Q_i the complement of Q_i^* and consider the following PPER-intervals:

$$[\{(\perp, \perp)\}, Q_i] \quad (i \in I);$$
$$[P, D \times D].$$

These intervals are finite elements in PPER-int. Moreover we have that:

$$[\{(\perp, \perp)\}, Q_i] \not\subseteq_C [r, R] \quad [R \subseteq Q_i \text{ fails because } (x_n, y_n) \notin Q_i].$$

$$[P, D \times D] \not\subseteq_C [r, R] \quad [(x_n, y_n) \in P \text{ but } (x_n, y_n) \notin r].$$

Then it holds $(\star)$ $[\{(\perp, \perp)\}, Q_i]$ and $[P, D \times D]$ are not in π.

Let us consider now a maximal interval $[S, S]$.

We have two possibilities: $(x_n, y_n) \in S$ or $(x_n, y_n) \notin S$.

In the first case we have $P \subseteq S \subseteq D \times D$ and hence $[P, D \times D] \subseteq_C [S, S]$.

In the second case $(x_n, y_n) \in D \times D \backslash S \, (= S^*)$. Let us consider S_n^*, this a n-minimal *Co*-PPER such that $(x_n, y_n) \in S_n^*$. Hence there exists $i_0 \in I$ such that $Q_{i0}^* = S_n^*$. For such i_0 we get that

$$[\{(\perp, \perp)\}, Q_{i0}] \subseteq_C [S, S].$$

What we have shown is then that for each maximal element $[S, S]$ the following holds:

$$(\star\star) \qquad [P, D \times D] \subseteq_C [S, S] \text{ or } \exists\, i \in I\, [\{(\perp, \perp)\}, Q_i] \subseteq_C [S, S].$$

Let us consider now the numbering of finite elements of PPER-int.

There exist n_0, $n_i \, (i \in I)$ such that $e_{n_0} = [P, D \times D]$, $e_{n_i} = [\{(\perp, \perp)\}, Q_i]$.

We fix $m \in \omega$ such that $n_0 < m$, $n_i < m \, (i \in I)$, whose existence is guaranteed by the finiteness of I, and consider the sub-path $\rho = \pi_m$. From $(\star)$ and $(\star\star)$ it follows that π_m cannot be maximally consistent. □

DEFINITION 7.6: Let e be a finite element of D and g : PPER-int $\rightarrow D$ a continuous function (D is a Scott domain). Let $x \in$ PPER-int, R be a PPER and T' a sub-tree of T.

(i) T' is a *x-g-witness* tree for e <u>*iff*</u>

1. T' is finite, uniform and an initial subtree of T.

2. For every path in T' which ends with a leaf and is x-maximally consistent it holds $e \sqsubseteq g(\sqcup\pi)$.

(ii) T' is R-maximally consistent *iff* it is a $[\{(\bot, \bot)\}, R]$-g-witness tree.

LEMMA 7.7: *Let g : PPER-int$\rightarrow$ D be continuous, $e \in D^0$* (i.e. *e is a finite element of D) and $x \in$* PPER-*int. Besides, let us assume that the following holds: $e \sqsubseteq \sqcap\{g(z) \mid x \subseteq_C z \;\&\; z$ is a maximal.$\}$*

Then there exists a x-g-witness tree T' for e.

Proof: We prove this lemma by contradiction.

Let us assume that such a tree does not exist and consider the subtree T' of T defined in the following way: for each node N we erase all the node below N if the path π to N satisfies at least one of the following conditions:

1. π is not x-maximally consistent.
2. $e \sqsubseteq g(\sqcup\pi)$.

The so obtained tree T' is clearly uniform. Moreover it must be infinite (otherwise it would be a x-g-witness tree for e).

Let us now take any infinite path π in T' (which exists since otherwise T' would be finite and hence a witness tree). We claim

(i) $\sqcup\pi$ exists in PPER-int.

(ii) $\sqcup\pi$ is maximal

(iii) $x \subseteq_C \sqcup\pi$.

To prove (i) it is enough to see that $\{\sqcup\pi_n \mid n \in \omega\}$ is a direct set. For (ii), if $\sqcup\pi$ were not maximal, then it would exist, by Lemma 7.5, a finite sub-path π' not maximally consistent, and then, by the definition of T' all nodes below π' should be erased, contradiction.

To prove (iii) let us assume $x \not\subseteq_C \sqcup\pi$. In such a case it should exist a finite element $d \in$ PPER-int such that $d \subseteq_C x \,\&\, d \not\subseteq_C \sqcup\pi$. Let us consider now the level k such that (in the numbering of finite elements of PPER-int) $e_k = d$.

Let ρ be a path in T descending down to k-th level: clearly x cannot satisfy the path $\rho \cup (\neg e_k)$.

Therefore every path that, up to level $n+1$, coincides with $\rho \cup (\neg e_k)$ is truncated in T' (and therefore finite). Thus if $e_k = d \not\subseteq_C \sqcup\pi$ it follows $\neg e_k \in \pi$, a fact which implies that π is finite, contradiction. Then $x \subseteq_C \sqcup\pi$ and $\sqcup\pi$ is maximal.

By the definition of T' we get $\forall n \in \omega\ e \sqsubseteq g(\sqcup \pi_n)$. By the compactness of e it follows $e \sqsubseteq g(\sqcup \pi)$, a thing that contradicts the hypothesis $e \sqsubseteq \sqcap\{g(z) \mid x \subseteq_C z \ \&\ z \text{ is maximal}\}$ by (i), (ii) and (iii).

Therefore it must exists a x-g-witness tree for e. □

COROLLARY 7.8: *Let g : PPER-int$\rightarrow$ D be continuous, $e \in D^0$, $S \in PPER$ and $e \sqsubseteq \sqcap\{g([R, R]) \mid R \subseteq S\}$. Then there exists a S-g-witness tree.*

Proof: Immediate from the previous lemma (by considering $x = [\{(\bot, \bot)\}, S]$). □

LEMMA 7.9: *Let π be a finite path in T and let $x \in$ PPER-int with $x = \sqcup X$, where $X = \{x_k \mid k \in \omega\}$ is a direct set. Moreover let us suppose that it does not exist any maximal element z such that:*

1. *z satisfies π.*
2. *$x \subseteq_C z$.*

Then there exists $n \in \omega$ such that no maximal element z satisfies 1. *and the condition*

$2._n\ x_n \subseteq_C z$.

Proof: It is not restrictive to assume X being an ascending chain.

Let us suppose that $\forall k \in \omega$ there exists a maximal element z_k such that z_k satisfies (1) & $(2)_k$.

We construct an element ζ in the following way: suppose that we have already build a prosecution π^{i-1} of π up to a level i. We now build a prosecution π^i by descending rightwards from the end of π^{i-1} if there exist infinite indexes j_k such that z_{jk} satisfies $\pi^{i-1} \cup \{e_i\}$. If such condition is not fulfilled we set $\pi^i = \pi^{i-1} \cup \{\neg e_i\}$. The union of all the paths π^i is an infinite path π'. Clearly there exists $\sqcup \pi'$ because the set $\{\sqcup \pi_i \mid i \in \omega\}$ is ascending. We set $\zeta = \sqcup \pi'$. We claim that ζ is maximal. In fact, if it was not the case, by Lemma 7.5 it should exist a finite path not maximally-consistent. This contradicts the fact (true by definition of ζ) that for each level i there exist infinite maximal elements consistent with π_i.

Moreover we have that $x \subseteq_C \zeta$. In fact let e be a finite element such that $e \subseteq_C x$. Then there exists $h \in \omega$ such that $e \subseteq_C x_k$, if $k > h$. Now let us consider in T the i-th level for which $e = e_i$. There are infinite $k \in \omega$ such that $e \subseteq_C x_k$, so that $e_i \in \pi$. Therefore $e \subseteq_C \sqcup \pi' = \zeta$. Finally, it is easy to check that ζ satisfies π. So we have got a contradiction with the hypothesis. □

COROLLARY 7.10: *Let π be a finite path in T, R a PPER and $\{R_k \mid k \in \omega\}$ a filtered set of PPER's such that $\bigcap_{k\in\omega} R_k = R$. Moreover, let us assume that there is no maximal element $z = [S, S]$ such that:*

1. *z satisfies π.*
2. *$S \subseteqq R$.*

Then there exists $n \in \omega$ such that no maximal element satisfies condition 1. and the condition

2.$_n$ $S \subseteqq R_n$.

Proof: Immediate from the previous lemma by considering the intervals $[\{(\bot, \bot)\}, R]$ and $[\{(\bot, \bot)\}, R_k]$. $\square$

PROPOSITION 7.11: *Let g : PPER-int$\rightarrow$ D be a continuous function, $e \in D^0$ and $x \in$ PPER-int with $x = \sqcup\{x_k \mid k \in \omega\}$.*

Moreover let us assume that $e \sqsubseteqq \sqcap\{g(z) \mid x \subseteqq_C z$ and z maximal$\}$.

Then there exists $n \in \omega$ such that $e \sqsubseteqq \sqcap\{g(z) \mid x_n \subseteqq z$ & z is maximal$\}$.

Proof: From Lemma 7.7 it descends that there exists a x-g-witness tree T' for e. By definition of witness tree it follows that each path π that reaches a leaf is such that one of the following two conditions is satisfied:

(i) π is not x-maximally consistent

(ii) $e \sqsubseteqq g(\sqcup\pi)$.

Let us consider now the set $\{\rho_j \mid j \in J\}$ of all the paths satisfying (i). J is obviously finite because T' is a finite tree. From Lemma 7.9 it follows then

$\forall j \in J \quad \exists k_j$ such that ρ_j is not x_{k_j} maximally consistent.

Let n be such that $\forall j \in J\, k_{k_j} \subseteqq_C x_n$.

ρ_j is not x_n-maximally consistent and hence, given a maximal element z such that $x_n \subseteqq_C z$, the path associated to z must coincide, in its initial part, with some path π_z in T' such that π_z is x-maximally consistent.

Therefore, from definition of witness tree, it follows $e \sqsubseteqq g(\sqcup\pi_z)$.

But $\sqcup\pi_z \subseteqq_C z$, so $e \sqsubseteqq g(z)$ and therefore

$e \sqsubseteqq \sqcap\{g(z) \mid x_n \subseteqq_C z$ and z is maximal$\}$. $\square$

PROPOSITION 7.12: *Let* $\{f_n : PPER\text{-}int \to D \mid n \in \omega\}$ *be a direct set of continuous function such that* $\bigsqcup_{n\in\omega} f_n = f$. *Then, for any PPER S,*

$$\bigsqcup_{n\in\omega} \bigsqcap_{r \subseteq S} f_n([r, r]) = \bigsqcap_{r \subseteq S} f([r, r]).$$

Proof: $(\sqsubseteq)$ is immediate.

To prove the opposite direction let e be a finite element such that $e \sqsubseteq \bigsqcap_{r \subseteq S} f([r, r])$. Then there exists a S-f-witness tree T' for e. Let $H = \{\rho_j \mid j \in J\}$ be the set of all the paths which are S-maximally consistent. Then, from the definition of witness tree $\rho \in H \Rightarrow e \sqsubseteq f(\sqcup\rho)$. Let h : PPER-int $\to D$ be the function defined in the following way:

$$h(x) = \begin{cases} e & \text{if} \quad \exists \rho \in H.\ \sqcup \rho \subseteq_C x \\ \bot & \text{otherwise} \end{cases}$$

Clearly h is finite and $h \sqsubseteq f$. Therefore there exists $n \in \omega$ such that $h \sqsubseteq f_n$.

For every $[r, r]$ maximal such that $r \subseteq S$ let us consider the path $\rho_r \in T'$ such that $\rho_r = T' \cap \pi_r$, where π_r is the infinite path of $[r, r]$. We get now

$$e \sqsubseteq f_n(\sqcup\rho_r) \sqsubseteq f_n(\sqcup\pi_r) = f_n([r, r]).$$

Therefore $e \sqsubseteq \bigsqcap_{r \subseteq S} f([r, r]) \Rightarrow e \sqsubseteq \bigsqcup_{n\in\omega} \bigsqcap_{r \subseteq S} f_n([r, r])$. $\square$

PROPOSITION 7.13: *Let* $f : PPER\text{-}int \to D$ *be a continuous function and* $\boldsymbol{R} = \{R_n \mid n \in \omega\}$ *a filtered set of PPER's such that* $\cap\boldsymbol{R} = R$.

Then $\bigsqcup_{n\in\omega} \bigsqcap_{r \subseteq R_n} f([r, r]) = \bigsqcap_{r \subseteq R} f([r, r])$.

Proof: $(\sqsubseteq)$ is trivial.

For the opposite direction let e be a finite element in D such that $e \sqsubseteq \bigsqcap_{r \subseteq R} f([r, r])$. Since, by assumptions, $[\{\bot, \bot\}, R] = \bigsqcup_{n\in\omega} [\{\bot, \bot\}, R_n]$, there exists $n \in \omega$ such that $e \sqsubseteq \bigsqcap_{r \subseteq R_n} f([r, r])$ (by Lemma 7.11). Therefore $e \sqsubseteq \bigsqcup_{n\in\omega} \bigsqcap_{r \subseteq R_n} f([r, r])$. $\square$

THEOREM 7.14: $[\forall_{\leqq}]$: *PPER-int*$\rightarrow$*[PPER-int*$\rightarrow$*PPER-int]*$\rightarrow$*PPER-int is well defined and continuous.*

Proof: First of all let us prove that $[\forall_{\leqq}]$ is well defined.

Given $y = [s, S] \in$ PPER-int and $f \in$ [PPER-int$\rightarrow$ PPER-int] we have that $\forall_{\leqq}(y)(f) = [\bigsqcap_{r \subseteqq S} f^+([r, r]), \bigsqcap_{r \subseteqq S} f^-([r, r])]$. It is straightforward to check that $\forall_{\leqq}(y)(f)$ is an interval.

Let us now first verify that $\forall_{\leqq}(y)$ is monotonous and continuous in [PPER-int$\rightarrow$ PPER-int].

For the monotonicity let $f, g \in$ [PPER-int$\rightarrow$ PPER-int] with $f \sqsubseteqq g$, *i.e.* $\forall z \in$ PPER-int $f^+(z) \sqsubseteqq g^+(z) \sqsubseteqq g^-(z) \sqsubseteqq f^-(z)$. Then $\bigsqcap_{r \subseteqq S} f^+([r, r]) \sqsubseteqq \bigsqcap_{r \subseteqq S} g^+([r, r]) \sqsubseteqq \bigsqcap_{r \subseteqq S} g^-([r, r]) \sqsubseteqq \bigsqcap_{r \subseteqq S} f^-[r, r])$, *i.e.* the monotonicity of $\forall_{\leqq}(y)$ in [PPER-int$\rightarrow$PPER-int] holds.

For the continuity let $\{f_n\}_{n \in \omega}$ be an ascending chain in [PPER-int$\rightarrow$PPER-int] such that $\bigsqcup_{n \in \omega} f_n = f$.

We need to prove that $\bigsqcup_{n \in \omega} \forall_{\leqq}(y)(f_n) = \forall_{\leqq}(y)(f)$ *i.e.*

$$\bigsqcup_{n \in \omega} \bigsqcap_{r \subseteqq S} f_n^+([r, r]) = \bigsqcap_{r \subseteqq S} f^+([r, r]) \tag{1}$$

$$\bigsqcap_{n \in \omega} \bigsqcap_{r \subseteqq s} f_n^-([r, r]) = \bigsqcap_{r \subseteqq s} f^-([r, r]). \tag{2}$$

(1) follows immediately from Proposition 7.12.

To prove (2) it is easy to check that

$$\bigsqcap_{n \in \omega} \bigsqcap_{r \subseteqq s} f_n^-([r, r]) = \bigsqcap_{r \subseteqq s} \bigsqcap_{n \in \omega} f_n^-([r, r]) = \bigsqcap_{r \subseteqq s} f^-([r, r]).$$

We are then now left with the proof of the continuity of $\forall_{\leqq}$, since the monotonicity can be trivially verified.

Let $\{y_n\}_{n \in \omega}$ with $y_n = [s_n, S_n]$ be an ascending chain in PPER-int such that $\bigsqcup_{n \in \omega} y_n = y = [s, S]$, *i.e.* $\bigsqcup_{n \in \omega} s_n = s$ and $\bigsqcap_{n \in \omega} S_n = S$.

What we have to prove is then that, given $f \in$ [PPER-int$\rightarrow$PPER-int], $\bigsqcup_{n\in\omega} \forall_{\leqq}(y_n)(f) = \forall_{\leqq}(y)(f)$, *i.e.*

$$\bigsqcup_{n\in\omega} \bigsqcap_{r\subseteqq S_n} f^+([r, r]) = \bigsqcap_{r\subseteqq S} f^+([r, r]) \tag{1}$$

and

$$\bigsqcap_{n\in\omega} \bigsqcap_{r\subseteqq s_n} f^-([r, r]) = \bigsqcap_{r\subseteqq s} f^-([r, r]). \tag{2}$$

(1) follows immediately from Proposition 7.13. To prove (2) we have only to check that $\bigsqcap_{n\in\omega} \bigsqcap_{r\subseteqq s_n} f^-([r, r]) \subseteqq \bigsqcap_{r\subseteqq s} f^-([r, r])$, since the other inclusion trivially holds. Let $p \in D \times D$ and be a finite element such that $p \notin \bigsqcap_{r\subseteqq s} f^-([r, r])$. Then there exists $r' \subseteqq s$ such that $p \notin f^-([r', r']) \Rightarrow p \in f^{*-}([r', r'])$, where $f^{*-}([r', r']) = D \times D \backslash f^-([r', r'])$ (f^{*-} is continuous on *Co*-PPER). Hence, since p is finite, we have that there exists a finite element $[c, C]$ in PPER-int, with $c \subseteqq r' \subseteqq C$, such that $p \in f^{*-}([c, C])$, hence $p \notin f^-([c, C])$. Since c is a finite PPER there exists n' such that $c \subseteqq s_{n'}$ and then $[c, C] \subseteqq_C [s_{n'} \cap r', s_{n'} \cap r']$.

Therefore $p \notin f^-([s_{n'} \cap r', s_{n'} \cap r']) \Rightarrow p \notin \bigsqcap_{n\in\omega} \bigsqcap_{r\subseteqq s_n} f^-([r, r])$. □

THEOREM 7.15: $[\mu]$: [*PPER-int*$\rightarrow$*PPER-int*]$\rightarrow$PPER-int *is a continuous function.*

Proof: Trivial. □

8. THE PPER-int KIND FRAME IS A KIND MODEL AND RULE (AMBER) IS SOUND

Let us firstly deal with the proof that the PPER-int kind frame is indeed a kind model, *i.e.* rule (Max-μ) is sound.

LEMMA 8.1: *Let f : PPER-int$\rightarrow$PPER-int be a function defined by a maximal and contractive type,* i.e. $f = \lambda\, d \in$ *PPER-int.* $\sigma_{\eta(t/d)}$ *where it holds Max*(σ) *and σ is contractive in t.*

Then

(i) $\forall [r, R] \in PPER\text{-int}\ (f([r, R]))_{n+1} = (f([r_n, R_n]))_{n+1}$

(ii) μf *is a maximal element of PPER-int.*

Proof: (i) See [10].

(ii) Let $[r, R]$ be the least fix point of f. Then $[r, R] = [f^+([r, R]), f^-([r, R])]$, *i.e.* we have

1. $r = f^+([r, R])$
2. $R = f^-([r, R])$.

Let us first prove by induction that $\forall n \in \omega\, r_n = R_n$.

The base case $n = 0$ is trivial.

For the induction step $r_{n+1} = (f^+([r_n, R_n]))_{n+1}$ and $R_{n+1} = (f^-([r_n, R_n]))_{n+1}$ by (i).

By the induction hypothesis $r_n = R_n$. Since f maps maximal elements in maximal elements it follows that $(f^+([r_n, R_n]))_{n+1} = (f^-([r_n, R_n]))_{n+1}$ i.e. $r_{n+1} = R_{n+1}$. □

Proof of Lemma 4.13: Immediate from the above lemma. □

Let us turn now our attention to the proof that the rule (Amber) is sound in our model.

LEMMA 8.2: *Let f : PPER-int→PPER-int be a function defined by*

$$f = \lambda d \in PPER\text{-}int.\ \sigma_{\eta(t/d)} \quad \textit{where } Max(\sigma) \textit{ and } \sigma \textit{ is contractive in } t.$$

Then $\forall n \in \omega\, (f^n(I_1))_n = (f^n(I_2))_n$, *where* $I_1 = [\{(\bot, \bot)\}, \{(\bot, \bot)\}]$ *and* $I_2 = [\{(\bot, \bot)\}, D \times D]$.

Proof: By induction on n.

The base case $n = 0$ is easy, since each approximation at level 0 is $\{(\bot, \bot)\}$.

Let us assume the thesis for n holds.

$$\begin{aligned}
\text{Then } (f^{n+1}(I_1))_{n+1} &= (f(f^n(I_1)))_{n+1} \text{ by definition} \\
&= (f((f^n(I_1))_n))_{n+1} \text{ by Lemma 8.1} \\
&= (f((f^n(I_2))_n))_{n+1} \\
&\qquad \text{by the induction hypothesis} \\
&= (f(f^n(I_2)))_{n+1} \\
&= (f^{n+1}(I_2))_{n+1}. \quad \square
\end{aligned}$$

THEOREM 8.3: (i) *Let* f, g : *PPER-int*$\rightarrow$*PPER-int be as the Lemma above and such that* $f \leqq_M g$, i.e. *for each maximal element* z *we have* $f(z) \leqq_M g(z)$).

Then $\mu f \leqq_M \mu g$.

(ii) *The rule (Amber) is sound.*

Proof: (i) We have that, by defining $I_1 = [\{(\perp, \perp)\}, \{(\perp, \perp)\}]$ and $I_2 = [\{(\perp, \perp)\}, D \times D]$,

$$\begin{aligned}
\mu f &= \bigsqcup_{n \in \omega} f^n(I_2) \\
&= \bigsqcup_{n \in \omega} (f^n(I_2))_n \quad \text{(by algebricity of PPER)} \\
&= \bigsqcup_{n \in \omega} (f^n(I_1))_n \quad \text{by Lemma 8.2} \\
&= \bigsqcup_{n \in \omega} [((f^+)^n(I_1))_n, ((f^-)^n(I_1))_n].
\end{aligned}$$

Analogously $\mu g = \bigsqcup_{n \in \omega} [((g^+)^n(I_1))_n, ((g^-)^n(I_1))_n]$.

From the condition $f \leqq_M g$ it then follows

$$\forall n \in \omega\, ((f^+)^n(I_1))_n \subseteqq ((g^+)^n(I_1))_n \text{ and hence } \mu f \leqq_M \mu g.$$

(ii) Immediate from (i). □

9. LIMITATIONS OF A NATURAL EXTENSION OF THE PPER INTERVALS APPROACH

Using the Interval of PPER's we have managed to perform a step forward in the direction of modeling the full QUEST system. This at the cost of extending the system with a "maximality" predicate. This extension of the syntax however does not introduce, as we have pointed out, any essential limitation to the expressive power of QUEST.

What μ-FunK lacks with respect to the QUEST language is higher order quantification, *i.e.* given a constructor $F : K \Rightarrow$ Type where K : Kind it is not possible to use the function F to build an infinite product type (denoted in the following by $\forall_K F$).

It might be thought that we could introduce higher order quantification in our system since the model we proposed seems to easily support such an extension by means of a natural generalization of the semantics of $\forall_{\leqq}$:

given a function $F : \mathbf{K} \to$ PPER-int, where $\mathbf{K}$ is the interpretation of K, a natural way of modeling $\forall_K F$ is the following:

$$\forall_K F = [\bigsqcap_{z \in \mathrm{Max}(\mathbf{K})} F^+(z), \bigsqcap_{z \in \mathrm{Max}(\mathbf{K})} F-(z)].$$

Unfortunately such an extension, does not work since, in general, the constructor $\forall_K$ is not continuous.

The aim of what follows is to present a counter example to the continuity of $\forall_K$ if it is defined as above.

Counter example to the continuity of $\forall_K$

To prove that the semantics of PPER interval cannot be extended in the natural way we prove now that

$$\forall_{[\mathrm{Type} \to \mathrm{Type}]} : ([\mathrm{PPER\text{-}int} \to \mathrm{PPER\text{-}int}] \to \mathrm{PPER\text{-}int}) \to \mathrm{PPER\text{-}int}$$

is not continuous.

To present this counter example with a sufficient level of detail some preliminary notions and Lemmas are necessary.

Notation: $\forall_2$ will denote $\forall_{[\mathrm{Type} \to \mathrm{Type}]}$.

Let $\{p_i \mid i \in \omega\}$ be an enumeration of the finite elements of $D \times D$ and let P_n be the smallest PPER containing $\{p_1, \cdots, p_n\}$. For each $k \in \omega$ we define:

$$g_k(x) = \begin{cases} [\{\bot, \bot\}, D \times D] & \text{if } x \subseteq_C [P_k, D \times D] \\ \alpha & \text{otherwise} \end{cases}$$

where α is a fixed maximal interval.

Let us prove some properties of the functions g_k.

It is easy to check that, by definition of g_k, for each maximal element $z \in$ PPER-int $g_k(z) = \alpha$. It then follows that $g_k \in \mathrm{Max}([\mathrm{PPER\text{-}int} \Rightarrow \mathrm{PPER\text{-}int}])$

LEMMA 9.1: *For each $k \in \omega$ g_k is a continuous function.*

Proof: Let $x = \bigsqcup_{i \in \omega} x_i$.

If $x \subseteq_C [P_k, D \times D]$ then there is nothing to do. Otherwise it has to exist a finite element $e \in (\mathrm{PPER\text{-}int})^\circ$ such that $e \subseteq_C x$ & $e \not\subseteq_C [P_k, D \times D]$. We have that $e \subseteq_C x \Rightarrow \exists j \in \omega.\ e \subseteq_C x_j$. Then $e \subseteq_C x_j \Rightarrow x_j \not\subseteq_C [P_k, D \times D] \Rightarrow g_k(x_j) = \alpha$. $\square$

LEMMA 9.2: *Let e be a finite element of PPER-int such that $e = [r, D \times D] \subseteq_C [D \times D, D \times D]$. Then there exists $k^* \in \omega$ such that, if $k \geqq k^*$ then $g_k(e) = [\{\perp, \perp\}, D \times D]$.*

Proof: Since r has to be a finite subset of $D \times D$, necessarily there exists $k^* \in \omega$ such that $r \subseteq P_{k^*}$. Hence if $k \geqq k^*$ we have that $e \subseteq_C [P_k, D \times D]$ and then $g_k(e) = [\{\perp, \perp\}, D \times D]$. $\square$

Let $E = \{e_i | i \in \omega\} \subseteq (\text{PPER-int})^0$ be an ascending chain such that $\sqcup E = [D \times D, D \times D]$.

Let us consider the functionnals F_n : [PPER-int $\rightarrow$ PPER-int] $\rightarrow$ PPER-int defined, for each $n \in \omega$, as follows:

$$F_n(f) = \begin{cases} f(e_n) & \text{if } \forall x \in \text{PPER-int } f(x) \subseteq_C \alpha \\ \alpha & \text{otherwise} \end{cases}$$

(α is the maximal element in the definition of the g_k's).

LEMMA 9.3: *$\forall n \in \omega$ F_n is continuous.*

Proof: Let $f = \bigsqcup_{j \in \omega} f_j$ with $\{f_j \mid j \in \omega\}$ directed family of continuous functions.

If $\forall x \in$ PPER-int $f(x) \subseteq_C \alpha$ then

$$F_n(f) = f(e_n) = \bigsqcup_{j \in \omega} f_j(e_n) = \bigsqcup_{j \in \omega} F_n(f_j).$$

Besides, if $\exists x \in$ PPER-int such that $f(x) \not\subseteq_C \alpha$ then $\bigsqcup_{j \in \omega} f_j(x) \not\subseteq_C \alpha$ and hence there exists $j \in \omega$ such that $f_j(x) \subseteq_C \alpha$. Therefore

$$\bigsqcup_{j \in \omega} F_n(f_j) = \alpha. \quad \square$$

Using the above lemmas we can now prove that $\forall_2$ is not continuous.

It is easy to check that the family $F = \{F_n \mid n \in \omega\}$ is increasing in $n \in \omega$ and therefore there exists $\sqcup F$ and besides, $\forall f \in$ [PPER-int $\rightarrow$ PPER-int]

$$\begin{aligned} \sqcup F(f) &= \bigsqcup_{n \in \omega} F_n(f) \\ &= \begin{cases} f([D \times D, D \times D]) & \text{if } \forall x \in \text{PPER-int } f(x) \subseteq_C \alpha \\ \alpha & \text{otherwise} \end{cases} \end{aligned}$$

If f is a maximal element of [PPER-int $\rightarrow$ PPER-int] and $\forall x \in$ PPER-int $f(x) \subseteq_C \alpha$ then it necessarily has to be $f([D \times D, D \times D]) = \alpha$. Hence $\forall f \in \mathrm{Max}([\text{PPER-int} \rightarrow \text{PPER-int}]) \sqcup F(f) = \alpha$.

In particular, if $\alpha = [S, S]$ we have that

$$(\forall_2 (\sqcup F))^+ = \bigsqcap_{f \in \mathrm{Max}([\text{PPER-int} \rightarrow \text{PPER-int}])} \bigsqcup_{n \in \omega} (F_n)^+ (f) = S.$$

Nonetheless $\bigsqcup_{n \in \omega} \bigsqcap_{f \in \mathrm{Max}([\mathsf{Type} \rightarrow \mathsf{Type}])} (F_n)^+ (f) = [\{\bot, \bot\}, D \times D]$.

To prove this let us consider the previously introduced function g_k.

We have that $\forall x \in$ PPER-int $g_k(x) \subseteq_C \alpha$ and hence $F_n(g_k) = g_k(e_n)$.

Besides, for a given n by Lemma 9.2 there exists k^* such that $g_{k^*}(e_n) = [\{\bot, \bot\}, D \times D]$.

Therefore $F_n(g_{k^*}) = [\{\bot, \bot\}, D \times D]$ and then $\forall_2$ is not continuous.

We have then shown that if one whishes to use the PPER intervals as a basis for a model for an extension of μ-FunK something more subtle that the natural generalization of our semantics for $\forall_K$ has to be given.

ACKNOWLEDGMENTS

We are very grateful to Mariangiola Dezani for her gentle guidance, to Mario Coppo and Felice Cardone for useful suggestions and discussions and to two anonymous referees for helpful comments and suggestions.
The second author whishes to thank also Manuela Adolfi and Maria Carla Mantuano for their invaluable support.

REFERENCES

1. M. Ababi, G. D. Plotkin, A PER Model of Polymorphism and Recursive types, *Proc. Symposium on Logic in Computer Science,* IEEE, 1990, 355-364.
2. F. Alessi, F. Barbanera, Toward a Semantics for the QUEST Language, to appear in *Sixth Annual IEES Symposium on Logic in Computer Science,* Amsterdam, July 1991.
3. R. Amadio, Recursion over Realizability Structures, *Information and Computation,* 1991.
4. H. P. Barendregt, The Lambda Calculus: Its Syntax and Semantics, *Studies in Logic, 103,* North-Holland, 1981, Revised Edition, 1984.
5. V. Breazu-Tannen, T. Coquand, C. Gunter, G. Scedrov, Inheritance and Explicit Coercion, in *Fourth Annual IEEE Symposium on Logic in Computer Science,* 1989, 112-129.

6. K. Bruce, G. Longo, A Modest Model for Records, Inheritance and Bounded Quantification, *Information and Computations* 1990, *87,* 196-240.
7. L. Cardelli, *Typeful Programming,* SRC report 45, 1989.
8. L. Cardelli, G. Longo, *A Semantic Basis for Quest,* SRC report 55, 1990.
9. L. Cardelli, P. Wegner, On Understanding Types, Data Abstraction and Polymorphism, *Computing Surveys,* 1985, *17,* 471-522.
10. F. Cardone, Recursive Types for Fun, *Theoretical Computer Science,* 1991.
11. R. Cartwright, Types as Intervals, *Proc. Symposium on Principle of Programming Languages, ACM,* 1984, 22-36.
12. M. Coppo, A Completeness Theorem for Recursively defined Types, *Proc. 12th International Colloquium on automata, Languages and programming, LNCS 194,* 1985, 120-129.
13. M. Coppo, M. Zacchi, Type Inference and Logical Relations, *Proc. Symposium on Logic in Computer Science, IEEE,* 1986, 218-226.
14. M. Dezani Ciancaglini, B. Venneri, Partial Types and Intervals, *SIAM Journal on Computing,* 1990, *19,* 644-658.
15. C. Gunter, Universal Profinite Domains, *Information and Computation,* 1987, *72,* 1-30.
16. C. Gunter, D. Scott, Semantic Domains, *Handbook of Theoretical Computer Science,* North Holland, 1990, 633-674.
17. D. MacQueen, G. Plotkin, R. Sethi, An Ideal Model for Recursive Polymorphic Types, *Information and Control,* 1986, *71,* 95-130.
18. S. Martini, Bounded Quantifiers have Interval Models, *ACM Conference on LISP and Functional Programming Languages,*, 1988, 164-173.
19. J. C. Mitchell, A Type Inference Approach to Reduction Properties and Semantics of Polymorphic Expressions, *Proceedings of ACM Conference on LISP and Functional Programming,* 1986, 308-319.